JOURNAL OF OBJECT TECHNOLOGY
Online at www.jot.fm. Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2002**

Vol. 1, No. 2, July-August 2002

The start of an Eiffel standard

Bertrand Meyer, ETH/ISE

The launching of the JOT provides a welcome opportunity to resume the Eiffel column that ran for many years in JOOP from 1998 (edited first by Rock Howard from whom I took over in 1998). It will run pretty much like its predecessor, with a mix of contributions by guest columnists and by me, and a mix of topics — from language matters to methodological issues of object-oriented software construction to reports on industry projects — that I hope will be of interest not only to Eiffel users but also to many others.

A NEW TECHNICAL GROUP AT ECMA

On June 6 and 7 of 2002 the first meeting of the new technical group TC39/TG4 of ECMA — Eiffel language standardization — took place in Zürich. This is an important development for everyone interested in Eiffel and a fitting subject for this first JOT column. I will explain what the ECMA effort is about, how it will proceed, and what it means for the Eiffel community.

First, why a standard? Part of the answer is obvious: with the growth of Eiffel usage in corporate environments, the existence of an international standard is a strong signal to the management of companies benefiting from Eiffel that the technology is here to stay and that the various suppliers of compilers and other tools provide compatible offerings.

This is the political and commercial advantage. In the Eiffel case we also see a technical advantage: stabilizing the technology and resolving any remaining inconsistency or ambiguity. By bringing together the best experts in the use of Eiffel technology, who have applied it to many successful projects and gained deep insights into what matters for large-scale development, we hope to get the language to a state where all users will be comfortable with all of its features.

The effort is expected to last from eighteen months to two years, with about four meetings a year. The meetings will alternate between Europe and the US, with probably at least one in Australia. The convener of the group is Christine Mingins from Monash University in Melbourne (no connection with the last remark, of course) and the organizations represented at the first meeting included Axa Rosenberg (USA), CALFP Bank (France/UK/ USA, currently as observer), Enea Data (Sweden), ETH (Switzerland), ISE (USA), Loria (France), Monash (Australia). We expect a number of other companies to join in future meetings.

Cite this article as follows: Bertrand Meyer: *The Start of an Eiffel Standard*, in Journal of Object Technology, Vol. 1, No. 2, July-August 2002, pages 95-99. http://www.jot.fm/issues/issue_2002_07/column8

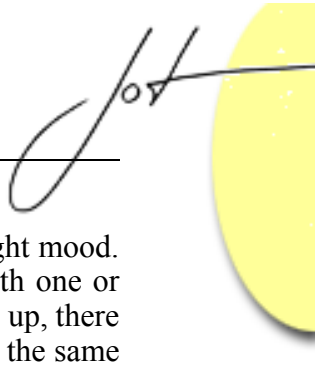
ECMA

Why choose ECMA for this effort? Based in Geneva, ECMA is a standards organization with a more than 40 years' history and many prestigious computer hardware and software standards to its credit. The name used to be an acronym, but ECMA doesn't expand it any more since it is no longer just European; it's still definitely about the Computer industry, but certainly not just for Manufacturers. So it's just ECMA. What distinguishes it from many other standard bodies is that it remains an Association of companies, collaborating to define effective standards. In contrast, the national standard bodies — ANSI in the US, DIN in Germany, AFNOR in France and many others — have an open policy which enables any interested party to participate in a standard. ECMA is different: while it welcomes comments from the public, it's a membership organization, intended to enable members to produce working standards quickly and effectively, with a minimum of bureaucratic hassle. Both models have their merits, but we like ECMA because its standards efforts are run in a goal-oriented, business-like manner. This seems just right for the Eiffel community, which needs to avoid political disputes and produce an excellent standard in as short a time as is realistic.

The idea of an ECMA group came from Emmanuel Stapf from ISE who has been active since early 2001 on the ECMA Technical Group that standardized the Common Language Infrastructure (CLI), that is to say, the basic specification of the Microsoft .NET technology. Emmanuel and I enjoyed the quality of the work performed by that group, and it seemed that the setup would be ideal for an Eiffel group. We were gratified that in a meeting in Redmond in March 2002 ECMA accepted our proposal to form a new committee for that purpose, TG4, part of Technical Committee 39 (originally "scripting languages", although this is just for historical reasons), and that the Secretary-General of ECMA, Jan van den Beld, was highly supportive from the start. Jan indeed attended the Zürich meeting where he was instrumental in getting the group going by briefing it on ECMA practices, procedures and advice — not to mention hosting a memorable dinner in a unique "Whisky Museum" not far from the city.

ECMA standards enjoy wide respect, as evidenced by the existence of a "fast-track" status enabling them to become International Standards Organization (ISO) standards without going through the usual channel of national bodies.

At this time, and in this industry, it would seem that electronic communication suffices to achieve a standard. The ECMA process applies a different view: while it extensively relies on the Internet — each Technical Group has a discussion group and a Web site, all documents are kept in FTP archives, and considerable work proceeds electronically between meetings — it insists on members physically getting together at regular intervals. The .NET (CLI) effort, run briskly with the intent of completing the process in a year, had one meeting every two months and a phone conference in-between; for Eiffel the pace will be a little less frantic, but not much. The general expectation is that to have a serious effect on the process you must attend most meetings. Clearly, this implies a taxing travel schedule, too taxing for some people. But the result is that the members are truly committed and that meetings, well prepared by electronic exchanges, can be very productive since people have met each other many times before and can get right down to solving the toughest issues.



On that count the Eiffel group seemed from the first meeting to be in just the right mood. All the members are experienced Eiffel developers with years of practice, some with one or more Eiffel compiler implementations behind them. Whenever a language issue came up, there wasn't much need for background explanation; everyone was almost immediately on the same wavelength. This doesn't mean everything will run smoothly; there will be more than enough opportunities for heated debates. But at least we won't waste much time getting everyone to a good level of understanding, because everyone is already there. Most importantly, the compiler writers share a desire to remove any remaining incompatibility between their implementations. All this bodes well for the success of the process.

For more information about ECMA see their Web site at http://www.ecma.ch, which also indicates how your company can become a member to participate in the Eiffel standard.

TIME TO BE SMART

It is widely recognized that a standard definition process is an opportunity to stabilize and codify proven practices, not the time to engage in new designs. TG4 will follow this rule, but we feel we can afford to be a bit bolder than the usual standards committee. Because the group members are so keenly aware of Eiffel-related issues, we feel that we shouldn't waste opportunities to improve the existing language design if we see some. This confidence is buttressed by several characteristics of the group and of Eiffel:

- The Eiffel community has always been particularly conservative in its approach to language evolution. While many languages originally designed at about the same time as Eiffel have changed dramatically, Eiffel still remains close to the original variant. In particular Eiffelists have always hated "featurism": there is still just one kind of loop. The changes and extensions that did appear over the years have always been justified by a high **signal-to-noise** ratio: significant increase of expressive power (the signal) at minimal addition to the overall complexity of the edifice (noise). These ideas are imprinted in the minds of the group's participants and we feel — I hope rightly — that we are immune to the usual lures of design-by-committee.
- Both compiler writers and application developers are strongly represented in the group. No language feature will be adopted for good until it has been successfully integrated in a compiler and used in applications.
- The Eiffel community, and the members of the standards group in particular, have a strong sense of quality. If we see a better way to do something, we won't release a standard with an inferior solution just because it's what was used in the past. This is the other side of the Eiffeslists' conservatism: they are not religious about compatibility; in the past they have accepted changes as long as they were justified, announced with sufficient lead time, and supported by conversion tools.
- Finally, the ECMA process provides insurance against changes getting out of control. In a completely public process where anyone can propose an extension, the only choice, to avoid meltdown, is to apply a drastic no-innovation policy. With a small group of experts who share a basic outlook, we can afford to be more open, while never lowering our guard against the creeping specter of featurism.

At this stage, almost no issue has been barred from consideration. In particular, there is a strong sense that we can find a good way to reconcile covariance with full static typing, a problem that has been a source of criticism for years. Some of the other open issues include multiple generic constraints, merging manifest arrays and manifest tuples, generalizing free operators. Some correspond to practices that have already been implemented by compilers, often because they were present in recent publications [2] [3]; others result from proposals by the group members, often following from discussions that have been pursued for years on Eiffel newsgroups and forums.

A STANDARD AND A BOOK

The current language reference, the book "Eiffel: The Language" [1], known as ETL-1, is more precise than most programming language descriptions. In particular, its "validity rules", which explain under what conditions a construct is legal — stating, for example, type constraints —, are all of the "**if and only if**" kind. It is easy to tell programmers some of what they *may not* do, for example assigning a real value to an integer variable. All language specifications do that. It is far more delicate to say "if your construct satisfies the following rules, then you *may* use it and I promise that any conforming compiler will accept it". That's the "if" part of "if and only if" and it requires a thorough job on the part of the language definition, to cover all possible cases. Here is an example of a validity rule, defining when the Inheritance part of a class is valid:

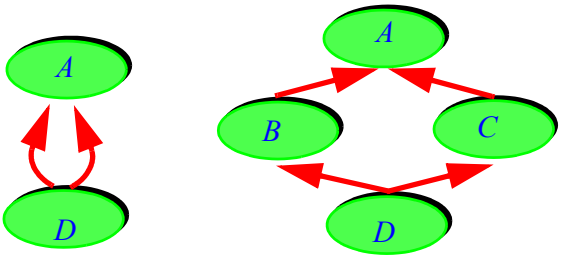
Parent rule *CHPR*

The Inheritance clause of a class *D* is valid if and only if it meets the following two conditions:

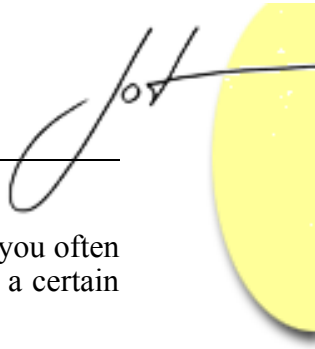
1 • In every Parent clause for a class *B*, *B* is not a descendant of *D*.

2 • If two or more Parent clauses are for classes which have a common ancestor *A*, *D* meets the conditions of the Repeated Inheritance Consistency constraint for *A*.

Names in green, such as Inheritance, denote syntactic constructs, here the inheritance clause of a class. The first clause means that you cannot specify a parent for a certain class if this would introduce a cycle in the inheritance hierarchy. (An equally rigorous definition preceding this rule states that a "descendant" of a class means a direct or indirect heir — formally, the class itself or, recursively, a descendant of one of its heirs.)The second clause refers to another rule of the same style, which governs the case of repeated inheritance, the so-called "diamond structure":



Repeated inheritance: **(direct)** **(indirect)**



This enables you to know exactly what you may and may not do; in other languages you often have to having to chase the manual for various interdictions, never being quite sure a certain usage is right until you have tried it with a compiler.

Eiffel's syntax is defined in a similarly rigorous way; the semantics is also precise although less formal for obvious technical reasons. This style, which has been refined and improved for the ongoing work on the next edition ETL-3 [3], paves the way for the standard.

A book, of course, is not a standard; ETL tries to be several things at once — tutorial, reference, programmer's guide — and the standard should only include a very small subset of this information, the dry description of the language's universally agreed properties. Since so much work is going into ETL-3, however, it would be unpleasant and inefficient to proceed separately on the two efforts.

So the general idea is to start from the book as the repository of basic information, and rely on automatic text processing tools to produce the first version of the standard. This is already the case in ETL-3, where a whole set of appendices (syntax reference, full language reference) are produced automatically through extraction of specially marked "formal" paragraphs from the main text. In the first parts of the book these paragraphs — syntax, validity constraints such as the one shown above, semantics — appear in the midst of explanations and examples; in the appendices they are repeated alone. (The designers of FrameMaker must be acknowledged here for providing sophisticated tools to support this automatic extraction and update process.)

The big push, in this approach, is to avoid redundancy — to achieve **reuse**. The same arguments that lie behind reuse of software in the Eiffel method suggest the reuse of formal parts in the language description. As we continue to discuss, document and tune the language in the next months, it would be a pity to have to update two or more texts; as in software, redundancy doesn't just mean more work, it also raises the prospect of divergence and contradictions.

Standards have their own formating rules and publishing constraints. At some point the reference appendices will have to take a life of their own and wean themselves from ETL-3 to become an ECMA standard. This should happen as late as possible in the process, when all the technical issues have been settled. Until then, we hope to have many illuminating discussions and use the opportunity to produce the best possible language standard, and the best possible language.

REFERENCES

[1] Bertrand Meyer: *Eiffel: The Language*, Prentice Hall, 1991 (first printing), 1992 (second printing with corrections).

[2] Bertrand Meyer: *Object-Oriented Software Construction, 2nd edition*, Prentice Hall, 1997.

[3] Bertrand Meyer: *Eiffel: The Language, 3rd edition*, work in progress at http://www.inf.ethz.ch/personal/meyer/ongoing/etl/, user name Talkitover, password etl3.

About the author

Bertrand Meyer is Professor of Software Engineering at ETH Zürich and scientific advisor of ISE (Santa Barbara).