ECMA/TC39/2000/17
ECMA/TC39-TG2/2000/6

Standardizing Information and Communication Systems

Minutes of the 1st meeting of TC39-TG2 held in Mountain View (USA) on 6th-7th November 2000

Convenor: Ms C. Thompson (HP)

Secretary: Mr. C. Clinick (Microsoft), Mr. J. van den Beld (SG ECMA)

Attending: Mr. Beard (Netscape), Mr. Deese (Microsoft), Mr. Eidt (HP), Mr. Gibbons (Pixo), Mr. Golde (Microsoft), Mr. Hejlsberg (Microsoft), Mr. Horwat (Netscape), Mr. Iarrobino (Sun), Mr. Jaeschke (Microsoft), Mr. Marcey (Intel), Mr. McCabe (Netscape), Mr. Neumann (Microsoft), Mr. Ruby (IBM)

Guests: Mr. Benito (JTC 1/SC22), Mr. Dyer (Mountain View Compiler), Mr. Plum (Plum Hall)

1 Opening and adoption of the agenda

Ms. Thompson, Convenor of TG2, opened the 1st meeting with a special welcome.

A roll call was held.

All participants will be added to the TG2 reflector (e-tc39-tg2@ecma.ch), and get a user-id and password for the ftp folder for TG2. A separate email on its usage will be sent around by the ECMA Webmaster.

Experts from other organizations than ECMA members can attend TG2 meetings but need an invitation by the ECMA Secretary General.

2 Approval of the agenda (00/1)

The agenda was approved with the following modifications:

- item 7 becomes: Actions to be taken
- item 9 becomes: Date and place of next meetings

In future the following permanent items will appear on the agenda (identical for all ECMA meetings):

1. Opening
2. Adoption of the agenda (incl. posting of new documents)
3. Approval of the minutes of the previous meeting of TG2
4. Matters arising from the minutes not covered elsewhere

.
.
.

n-2 Any other business

n-1 Date and place of next meetings

n Adjournment

3 Project Editor's report (00/2)

Mr. Jaeschke, editor for C#, presented his report, highlighting several topics. From this presentation and the discussion the following was noted:

- The initial draft for C# (00/3) was submitted by Microsoft, with the following standard disclaimer read out by **Mr. Deese:**

  - This contribution is being provided "AS IS", and the SPONSORS EXPRESSLY DISCLAIM ANY AND ALL WARRANTIES REGARDING THIS CONTRIBUTION, INCLUDING ANY WARRANTY THAT THIS CONTRIBUTION DOES NOT VIOLATE THE RIGHTS OF OTHERS OR IS FIT FOR A PARTICULAR PURPOSE.

  This disclaimer will appear on all documents contributed by Microsoft as a company. Contributions by individuals / editors will not carry such a disclaimer.

  Working documents in ECMA are called drafts. Each new version of a draft standard, e.g. the C# draft standard, will be a new draft, and will carry a new TG2 document number. There are **no** sub-versions of drafts, e.g. 2.1. The number of drafts is not limited.

  Mr. Jaeschke will prepare the first draft based on changes to the initial draft approved by TG2.

  The **final** draft is the document to be submitted for approval to the ECMA General Assembly. Submission has to be made no later than two months before an ordinary ECMA General Assembly which is held each half a year, in June and in December.

  Drafts are available from the TG2 ftp folder both in Word and in pdf format.

  Uploading of documents preferably has to be done in Word format. The ECMA Secretariat will then assign an ECMA document number, and produce the pdf version. Documents without such an ECMA number are not considered to be ECMA documents.

- Successive drafts are usually accompanied by a separate 'delta' document marking all changes that have been made between the last draft and the last but one draft. Other, additional, documents and ways of marking changes agreed by TG2 are left to the Editor and the TG. There is no further detailed and prescribed working method for the TG.

  The Editor, having considerable experience in ISO standardization, proposed several recommendations on how to deal with the drafts, e.g., usage of line numbers, separate (i.e. preferably not in TG meetings) reporting of typographical errors, etc.

  Examples of ISO standards which are useful for comparative purposes are the C and C++ standards (see 00/4 and 00/5): they shall be used for internal ECMA work only !

- The 'standard' **initial** clauses of an ECMA (and also ISO) standard shall be used:

  1. Scope
  2. Conformance
  3. References (in ISO standards only normative references are allowed), if any
  4. Definitions, if any
  5. Conventions and notations, if any
  6. Acronyms, if any

  It should be noted that this standard practice should be applied as soon as possible because one of the objectives is to fast-track the ECMA Standard into ISO/IEC JTC 1.

- The conformance is a very important clause describing how conformance to the standard can be claimed for an implementation. There may be a need to introduce some concepts, like:

- 'MAIN',
- 'hosted' versus 'freestanding' or 'embedded' implementation,
- 'options' (an option is something that has been completely specified in the standard) can be present or absent in an implementation, e.g. allowed extensions,
- (strictly) conforming program (a non-conforming program would be a program that, for example, relies on certain libraries for extensions although these libraries do not exist on a certain platform),
- internationalization (I18N),
- diagnostic levels, etc.

It is recommended to (re)consider the text for the conformance clause at a later stage in the standard development process.

– TG2 has considerable interest in a 'rationale' as proposed by **Mr. Jaeschke**. Preferably it is a living document somewhere in the ftp folder. It is often useful as an FAQ as well. However, it is not meant to be a tutorial. The document is open, and generally available. The document can probably best be (hot) linked to the standard, e.g. it could be included in the electronic publication of the future standard, but probably not in a paper version. Another possibility might be to publish the rationale as a TR (which can also be fast-tracked in JTC 1). A distinction has to be made between the rationale for the design of the language and the rationale for what is going into the standard. In the case of C# the language design rationale may not necessarily be complete because certain language aspects may not be discussed but taken as a given, e.g., because they occur also in C or C++.

  Developing, editing and maintaining a rationale is an activity for the whole group. The editor would probably mainly administer the document.

  The amount of (extra) work to develop a rationale is very likely not negligible, and needs to be considered as well.

  An interesting model for a rationale is Strustrop's book "Annotated C++".

  TG2 decided not to take a decision now, but to put the subject on next meeting's agenda, for further consideration.

– Cross-reference index: very useful.

– Normative and narrative text have to be clearly distinguished, e.g. narrative texts could go into informative annexes. (Informative annexes can be included in the standard behind the possibly present normative annexes).

  During the development of the standard it is very helpful to have the informative, explanatory text 'on line' at hand, e.g. via dynamic links. The ultimate standard contains only normative text, with the exception of possible informative annexes and some notes (which can be interspersed with the normative text; notes (and also figures) are never normative). Some redundancy in the normative text can be useful, although it can cause errors as well (especially during 'maintenance' years later), but should not be avoided for formalistic purity reasons only.

– (In)Dependence of C# on CLI. Contributions on this subject are welcome. **Mr. Iarrobino** would like to explore the possibilities of a completely selfstanding C# (standard). It is questionable whether an implementation would provide a useful compiler. Also TG3 discussed the subject (see TG3/2000/9), and the minimal set of base class libraries needed for C# .

– A suggestion was made to prepare an informative annex listing differences between C# and C, C++ and Java.

– Usage of double underscores for extended keywords: to be explored.

– The question whether testing is possible in advance was answered positively.

4 Work schedule

Monthly meetings are planned till August 2001, i.e. 10 meetings in total, to allow for approval of the final draft by TC39 in September 2001. Also next SC22 meeting may be relevant: it is scheduled for 17-20 September 2001. The first activity consists of a general overview, to be followed by detailed chapter-by-chapter review. Input is solicited, e.g. via email discussions.

If useful a (big) issues list will be prepared. Big issues may be assigned to individual owners, small issues will be resolved immediately. In the December 2000 meeting chapters 1 to 8 of the C# specification will be reviewed, in the January 2001 meeting chapters 9 to 17. **Input for the review as well as editorial changes should preferably be distributed no later than one week before the meeting, i.e., no later than 30 November 2000.**

5 Technical topics

TG2's general objective is to standardize C#, as used in practice: the intention is not to redesign the language but to consolidate it. A second main goal is to clearly specify the relation between C# and CLI.

Mr. Hejlsberg summarized the targets for the C# language as follows:

- General purpose rather than any specific domain.
- Make development faster and more efficient.
- Component oriented.
- Portable devices.
- Internationalization - full Unicode support.
- Robustness - type safety.
- Decimal support - more business application focused than predecessors.

Mr. Golde added two 'non-goals':

- Not going after the "assembler replacement" that C is.
- Not going for the mathematic pureness.

A round along the TG2 members in the meeting led to the following list of interesting topics (T), most of which were shortly clarified (C) and responded to by **Mr. Hejlsberg:**

1) T: Try to be as type-safe as possible. Generating values beyond a range is error-prone. It would be nice to be able to declare enumeration types that are limited to the given values.

   C: There is no range checking after certain operations, e.g., AND. Runtime checks are avoided systematically.

   TG2 has also to check whether all 'necessary' types can be written in the language, and whether there is a sufficient base level of functionality (such as console access), required to define the language.

2) T: Array co-variance

   C: TG2 to explore the possibility of arrays of types T and U where T and U are not similar and do not have the same type.

3) T: Byte versus sbyte. Why is the default unsigned ? Is sbyte ever used ?

   C: There does not seem to be a real agreement on what is the best way. There is no ubyte in C# . Having sbyte and ubyte in C# would also be an interesting combination.

   The inventors of the language get 'the benefit of the doubt'.

4) T: Implicit interface implementations

   C: TG2 to explore the interaction between interfaces and class inheritance.

5) T: Use of = = operator

   C: No magic solution known to overcome this 'pet peeve' problem of overloading the operator. Separate operators would be nice. Defining the border between C# and CLI is going to be a tough issue, according to **Mr. Golde**.

6) T: Something 'in between' structs and objects

   C: TG2 to explore how to pass structures by reference, and automatic (one-way) evacuation to the heap.

7) T: Why are events (duplicating the functionality of delegates) a core element of the language ?

   C: Events could not just be types of properties because of the need to multicast events. Events should not be part of the core language.

8) T: Delegates

   C: ?

9) T: (Weak) References

   C: CLI is dealing with weak references, C# is not.

10) T: Aliases and declarations (order dependency)

    C: 'Using statements come before declarations' is a technique preferred by Microsoft. Another approach could have been to put using statements and declarations as closely together as possible. **Mr. Horwat** explained that the Microsoft technique opens the source file up to abuse, e.g., by importing source code.

11) T: How closely are C# and CLI related ?

    C: See items 3 and 5 20).

12) T: 'For each' statement

    C: To a large extent this is a pragmatic solution to cover the 'for' statement. It is appreciated by programmers. It also opens opportunities for optimizations. For example, if an object supports current and movement methods then 'for each' will work on the object. 'For each' is an area where CLI and C# are (too) closely tied.

13) T: Semantics of boxing

    C: ?

14) T: Generics

    C: To enable 'generics' properly they would have to be part of the runtime. Generics are not in C#.

15) T: Length of multi-dimension arrays

    C: All arrays derive from system.array and the length property does not actually make much sense when used on multi-dimensional arrays. Ideally, the property would only be on single-dimension arrays.

16) T: Is it not possible for interfaces to declare constants and enums ?

    C: See 17). Microsoft has not implemented inherited enumerations. It is a version issue. Enums are constants. As soon as you add additional enums in the base class then it would shift existing enumerations forcing a recompilation of any derived classes.

17) T: Qualification of name spaces

    C: It is virtually impossible to fully qualify namespaces without resorting to 'tricks' in the code.

18) T: Measure portability, e.g. by considering the number of times something in the standard is specified as 'implementation dependent', or by considering on how many platforms C# runs.

    It is difficult to predict, e.g., new devices. The more C# is portable the easier it will be to implement new things.

19) T: Do not be too lenient in allowing too many new features and choices. Focus on practical and good things, to be specified within our relatively short time frame.

20) T: What do I have to know to write a C# compiler ?

    C: The base class library is the most important part of CLI that is needed for C#. C# itself has no library. TG2 may carefully consider what is the minimum library requirement for C#, e.g. by means of creating a

WG. It has also to be sorted out what of the base class library goes into the C# standard, into the CLI standard, and possibly into both (the core minimum library ?).

21) T: What is platform independent in the current C# implementation ?

C: To be determined in next meetings. The amount of interdependencies between the language specification and the runtime specification is likely not huge.

22) T: Is multi-threading allowed ? Why asking for any guarantee with respect to atomicity ?

C: The memory model is not defined in the language standard. Therefore, multi-threading is not guaranteed.

23) T: Integer portability

C: This is difficult but everything should be done to achieve this.

One question is whether there should be a platform-independent int, e.g., called sysint. This would then require the mapping and conversion of int to sysint. It is usually required, if only for efficiency reasons, to use the most efficient int on every particular platform hardware.

CLI has a pointer of type int called intptr.

One runs easily into complexity issues, e.g., when providing extensions beyond basic value ranges, e.g., to 32-bit (or the still futuristic 64-bit) integers: this compounds the whole design of a system. Will we end up with a C# 32 and a C# 64 (for the support of long sizes) ? What should be the aim of the language, make it easy for the majority of developers to port code or make it simple for large arrays on 64-bit architectures. Also should the array have length and long-length or just have a length that is 64-bit wide.

Do we make the a choice of 32/64 bit an explicit decision of the developer or an implicit part of the platform ?

A practical approach should be taken by TG2. Contributions on long-length are invited. **Mr. Golde** and **Mr. Horwat** will prepare an initial contribution.

6 Liaisons

Mr. Iarrobino is appointed by TG2 as their liaison officer to TG3. Future liaison needs, if any, will be dealt with in due time.

7 Actions

1. **Mr. Jaeschke** will write or co-ordinate the requirements for running a conforming program in specific environments.
2. **Mr. Jaeschke** will provide a simple example on how to display the rationale in the drafts.
3. **Mr. Jaeschke** will add the initial 'standard' clauses.
4. **Mr. Jaeschke** will develop an index for the C# standard.
5. Document 00/3 will be used as the initial draft for comments, and review at the next meeting.
6. **Mr. Horwat** will summarize the issues raised by him and mentioned under item 5. Other TG2 members are invited to make contributions on their issues, mentioned under item 5 as well.
7. **Mr. Hejlsberg** and **Mr. Golde** will prepare a contribution on long-length.
8. Microsoft will propose a minimum class library for C# (**Mr. Hejlsberg** and **Mr. Golde**).
9. **Mr. Golde** will provide a rationale for byte-versus-sbyte.
10. **Mr. Jaeschke** will prepare an initial, tentative delineation between normative and other (informative, non-normative) text.
11. **Mr. Horwat** will provide a summary on threading/atomicity/memory model.

8 Any other business

8.1 GA and CC minutes

These minutes, 4 to 6 per year, are available from the ftp site (ftp.ecmadoc.ch) in the folder COMMON.

9 Date and place of next meetings of TC39-TG2

No. 2 Cupertino/CA, 6-7 December 2000, on kind invitation by HP. Venue: see 00/

No. 3 Redmond/WA, 16-17 January 2001, on kind invitation by Microsoft

No. 4 Phoenix/AZ, 14-15 February 2001, on kind invitation by Intel

No. 5 Kona/HI, 15-16 March 2001, on kind invitation by Microsoft

No. 6 Raleigh/NC, 26-27 April 2001, on kind invitation by IBM

No. 7 Montreux/CH, 24-25 May 2001, on kind invitation by ECMA. Venue: see 00/

10 Adjournment

The Convenor closed the meeting.