L2/03-079

Unicode: Getting Through the Next Ten Years

Authors: *Michael Kaplan* (Trigeminal Software/Microsoft), *Cathy Wissink* (Microsoft), *Sandra Martin O'Donnell* (Hewlett Packard)

Abstract

In examining the current direction of the Unicode Standard, the potential future direction of the standard, and the program of work of the Unicode Technical Committee (UTC), the authors have identified procedural issues that we believe must be addressed. If these issues (and their root causes) are ignored, the problems that have started to emerge for implementers over the last few years will become more severe and frequent. These problems will hurt the credibility (and potentially, the usability) of the standard. This document will outline the procedural issues we found, some of the root causes, and suggest some potential solutions.

Introduction

At the heart of the Unicode Standard is the character repertoire. In the words of ***What is Unicode?***[1],

> "Unicode provides a unique number for every character, no matter what the platform, no matter what the program, no matter what the language"

Everything independent of the actual character repertoire—the non-characters, the Unicode Standard Annexes, or the Unicode Technical Standards—should be considered supplemental materials meant to assist in the usage of characters.   Use of the term "supplemental" is not intended to minimize the importance of or need for the work. The wide array of characters and their interaction requires such information for implementers to *use* Unicode. In short, characters just sit there; these additional or supplementary rules are what make characters useful to implementers (and ultimately to users). However, the UTC's overall workload has shifted in a direction that we feel is to the detriment of the standard.

The problems we discuss fall into two basic categories:

- the ability to complete the outstanding character proposals;
- the need to standardize all of the other (non-character) proposals.

The potential solutions we discuss are numerous. The practicality of each solution will have to be decided by the UTC.

Problem #1: Are We There Yet? Will We Make It?

The characters must be available first prior to developing implementation guidelines of said characters; the resources to actually encode these characters are finite, and dwindling quickly. Our biggest concern at this time needs to be whether or not all of the characters will be encoded before "the Canon is closed", which for most implementers will be in the Unicode 5.0 timeframe.

To be more specific, the world's living scripts used by at least 90% of the population have either been encoded or will be a part of Unicode 4.0.. At the same time, the work to handle the East Asian ideograms that must be encoded proceeds apace and the IRG is working to finish that work. The majority of the non-living scripts are currently on the Unicode Roadmap, and work to encode those

[1] http://unicode.org/standard/WhatIsUnicode.html

historical scripts is proceeding. And unfortunately it is in that last bit where we may stumble. Everson reports[2]

> ....there are at present no less than 92 scripts yet to be encoded! These scripts range from large, complex and famous dead scripts like Egyptian hieroglyphs, to small, little-known but simple scripts like Old Permic. But, importantly, about a third of the scripts are living scripts which are intended to go on the BMP....

Although Everson is quite optimistic about the potential of the Scripts Encoding Initiative, others are not quite so confident about the work's anticipated completion date. As more and more sponsoring companies in Unicode feel that "the work is done" for their purposes, corporate involvement is expected to lessen (e.g., NCR) or in some cases (such as Progress and Unisys) stop. Other companies may act in the same manner purely for financial reasons in the current economy, either decreasing their financial commitment or reducing their participation (and subsequent travel costs). There is genuine worry that the Unicode "Canon" will be de facto closed due to lack of industrial involvement and resources before all of the work is done, essentially keeping certain scripts from ever being represented in Unicode. In fact, some feel that the deadline the UTC faces may well be the Unicode 5.0 release.

Summary of problem: The UTC must find a way to do as much of the actual character addition that it can before a "closing of the Unicode canon" occurs, to maximize the usefulness of Unicode for all potential implementers.

Problem #2: Consistency vs. Correctness (When Bad Things Happen to Good Characters)

In theory consistency and correctness should be the same -- if one makes a correct decision in any part of the standard, then one can consistently refer to the decision. In practice, however, it proves to be more complicated. Just as a lawyer is often forced to make a decision that is ethically required yet may not be morally sustainable (such as vigorously defending a client one knows to be guilty), in the UTC we have sometimes made mistakes, and are often forced to choose between whether we want give the most accurate answer so that implementers can count on the standard being "right" or whether we want to give the most consistent answer so that implementers can count on the answer not changing.. It is one of those interesting questions where the advocates of each school of thought are both right -- and wrong.

This problem of mistakes in the standard and how to handle them is exacerbated by the fact that the Unicode Standard has grown beyond the ability of any one member of the UTC or indeed the EC to fully understand. (Several members of the UTC and the EC maintain expert knowledge of the bulk of the standard, but even they as imperfect human beings are forced to rely on the work of others in some cases. This in no way reflects poorly on these members; it is simply recognition of the standard's complexity.) Because no one person can understand all nuances of the standard, the UTC (and presumably the EC) is often forced to accept the accuracy of the proposals they are given without being able to fully review all of the technical issues therein. With fewer eyes reviewing and understanding the full content of proposals (as the standard grows more complex and as members drop out), the likelihood of such mistakes increases.

In looking at the nature of the mistakes that have happened, two interesting facts emerge:

A. The requirements and procedures for character proposals are codified and understood.[3] The needs are easy to describe and easy for others to understand with a bit of

[2] Unicode Technical Note #4, http://www.unicode.org/notes/tn4/

[3] Submitting New Characters or Scripts, http://unicode.org/pending/proposals.html

background knowledge. While mistakes can be made, they are comparably infrequent. Also, mistakes are both more often caught and less often forgotten -- they serve as excellent object lessons.

B. Nothing outside of the character proposals has a process that guarantees a basic standard of quality of the proposed new functionality. Decisions are sometimes made without full committee understanding of the technical consequences, and it may be years before implementations either from member companies or outside of the Standard actually hit problems. Although the review process of the *document* proposing the functionality is codified, a thorough review of the actual functionality is not.

While there is still room for improvement in the "A" category[4], the majority of errors in the standard that are not caught prior to releases fall into the "B" category, through no intentional fault of the UTC, the EC, or the presenters of proposals.

At the point where a problem is found, the promises that have been made to IETF, W3C, and others are reviewed. Potential solutions are often hotly debated between members of the UTC who span the entire spectrum between the two extremes of consistency and correctness. Every time such a debate occurs, the deeper problem is exposed yet again -- no matter what the UTC decision is, the mistake will cost the UTC some credibility, and all that can be done is damage control.

The truth is that the track record of the UTC has not been stellar on avoiding such mistakes. And the "default" answer of those who do not understand or have not fully reviewed a proposal (that is, a YES vote on a proposal) does not serve the committee well. One way or another, this must improve if we are not to simply give up either our consistency or our correctness (or both) forever.

Summary of problem: The UTC must find a way to bring more rigor to the proposals it manages outside of character proposals. The below section outlines some suggestions.

Proposed Solutions

None of the suggestions here are believed by the authors to be complete solutions. They are, however, steps toward a solution most likely to solve the dual problems we face:

- A lack of time and resources to finish the work to encode characters;
- The inability to guarantee much of the accuracy of the non-character work in progress or already in the standard..

These proposals are enumerated below.

Giving Difficult Character Proposals the Highest Priority

Conventional wisdom has been to get as much into the standard as possible, which obviously leads to the "low hanging fruit" of easy scripts being encoded first. Although this may get more characters encoded, the "Canon closing" deadline might be better served by encoding the difficult scripts that may require innovation to encode first. Doing so will make sure that all of the tools are in place so that future scripts additions may be able to be as simple as having the fonts and reading the character properties (something that companies will be able to do for far longer then they will be able to create shaping engines or other innovative development work).

[4] Freytag, Asmus. Suggested changes in UTC / L2 procedures, http://www.unicode.org/L2/L2003/03029-procedures.txt

Give Character Additions a Higher Priority than Everything Else

Once the character repertoire is at a steady state, the UTC could easily spend the next 10 years improving the processing of characters and rules for dealing with said processes. Since there is no real time limit on new "everything else" work, but there is a time limit on the characters, we **must** consider focusing our efforts on character additions for the near future. Authors doing extensive work on non-character proposals, especially new work, should consider shelving the proposals for a bit to focus on and successfully finish the character repertoire as it is outlined in the Roadmap. This of course does not limit implementers from continuing to innovate technically; it just means it will not become part of the standard for the near future. In fact, in our present situation that lacks a formal conformance model, no one is truly hurt by the inability to add functionality in the near future.

Slow Down the "Everything Else" Proposals

If we give implementers time to examine where their products expose holes in the proposed changes to the standard, rather than trying to proactively solve the issues for said implementers, then the likelihood of correct solutions will be much greater. The solutions will be based on a broader development experience rather than narrow or theoretical implementations. In short, let's get the characters in first, and minimize the "mucking about" with the characters for the time being.

Avoid Pre-emptive Proposals or Proposals Not Fully Understood

The truth is that both committees (UTC/EC) are already overburdened and it is likely to be years before anyone needs to seek out additional work for the standard.. The change in the economy and events in the industry dictate that many participants will have even less time to contribute to the standard. Proposals that do not have customers, users, or implementers officially requesting assistance or clarification are an unfortunate side effect of the fact that Unicode has two such committees that contain a lot of smart, technical people.[5] Such "orphan" or pre-emptive proposals should be shelved until there is a real-world request that requires UTC action in the form of a proposal.

More Abstentions from Members

There are many companies voting for or against non-character proposals that they either are years away from implementing, or will never implement. The problem is that there is too much trust in the UTC -- trust that the people presenting the proposals have done their due diligence. However, if a member company truly does not feel the issue is fully understood, it is better to abstain than to vote for a proposal that should not yet be approved.

Fewer Consensus Decisions, if Any

Looking at meetings over the past several years, a clear trend has been found that we find to be disturbing. There has been a vast increase in the number of decisions that were passed by consensus rather than by voting of member companies (see the table, below)[6].

Additionally, while looking at the nature of the decisions, before UTC-88 the majority of consensus decisions were not about technical issues that required UTC feedback (e.g. [86-C1] Consensus: Remand the Glossary discussion back to the Editorial Committee for a recommendation to the UTC).

[5] The best example that the authors can recall is that of the Combining Grapheme Joiner (U+034F), originally conceived online on the Unicode public mailing list.

[6] Source: Unicode Technical Committee Minutes, http://unicode.org/consortium/utc-minutes.html

For UTC-88 and later, the notion of having technical decisions passed by consensus came in vogue (e.g., [92-C8] Consensus: Synchronize the version and repertoire of the Unicode Collation Algorithm (UCA) with versions of the Unicode Standard starting with Version 4.0).

Meeting	When?	Consensuses	Motions
UTC-93	Fall 2002	36	4
UTC-92	Summer 2002	36	2
UTC-91	Spring 2002	32	5
UTC-90	Winter 2002	30	11
UTC-89	Fall 2001	30	11
UTC-88	Summer 2001	9	18
UTC-87	Spring 2001	10	22
UTC-86	Winter 2001	2	32
UTC-85	Fall 2000	6	14
UTC-84	Summer 2000	3	17
UTC-83	Spring 2000	10	24

The authors question the validity of a practice that encourages the "yes by default" method of getting work done in UTC meetings. Although consensus decisions streamline work within the UTC, they perpetuate the illusion of "no objections means acceptance". The end result is that a member not paying attention at the moment a consensus is requested or a person with misgivings not clear enough to be voiced could easily result in equivalence to a unanimous decision.

The suggested change is to no longer allow consensus decisions on such a wide array of such issues, especially those with a long-term technical impact to the standard; the only alternative is for member(s) to start openly objecting to consensus decisions on an issue by issue basis. To be blunt, it is in some cases our "streamlined" process that has in part gotten us into this mess. It is the authors' opinion that only less streamlining will help get us out of it.

Change the Model for Review of Proposals

The current model used for review is similarly geared toward getting things done:

- For review within the UTC, PDUTR becomes DUTR becomes UTR/UTS/UAX and for the most part it is based on review feedback and compromising on issues that are debated. It is a train that almost always moves forward, with the only delaying factors being the number of suggested changes made during review (so that the end result of few review comments is a faster item)
- For public review, issues are given an expiration date for giving feedback, after which a decision is made based on that feedback.

While this model clearly moves work along (a good thing), both review processes have a common, fatal flaw: if review is not done, or if review is incomplete, then the proposal still moves forward (in many cases lacking either expert advice or an implementation).. Once member companies and others get around to implementing solutions, they run into problems, including but not limited to:

- imprecise/incorrect language
- inaccurate technical content (mistakes)

- important issues missing from discussion or content

At the point where a problem is discovered, then if the change can be made to the standard, it is. If it cannot (such as when there are stability guarantees that prevent it), then one or more people will point out that the review period for the material was clear and unambiguous -- and so the fault lies in the reviewers. This argument really cannot be realistically made any longer, if the UTC is not to lose all credibility with either those who need consistent results for the sake of their own stability or those who depend on correct results to make proper use of the standard.. Additionally, until an implementation of a proposal has shipped to and been tested by customers (e.g., implementers outside the close circle of the UTC and their technical staff), the proposed solution cannot be considered well-tested.

The most plausible solution becomes obvious when one looks at how the Unicode Technical Committee was able to eventually discover problems -- in some cases within weeks and in others up to several years -- when people try to implement issues:

- Mark Davis recently commented on the Unicore alias: "Nobody can produce compatible implementations of Mongolian currently because it is so badly underspecified. And trying to introduce piecemeal chunks of description for shaping over successive versions of Unicode does not give the UTC any way to assess the way all the pieces fit together.... Until we have a complete picture that we can assess, it would be premature to make any changes in the standard."
- Steve Atkin of IBM found language and technical flaws in the Bidirectional Algorithm based on thorough review and a test implementation that found issues in Unicode's reference implementations (some of which had been there for years).
- Paul Nelson discovered problems in canonical mapping of various Hebrew marks (prior review by people interested in the Hebrew script did not point out the issues -- but those people admitted that they did not use the marks, so their opinion should likely have been discounted).
- Variation Selectors were added to Unicode without a good understanding from many members. Although need for them was specified, the number of amount of confusion reported later with text and usage suggests that they needed more work prior to being approved.
- Mark Davis found problems in the 3.2 release timeframe in proposals that he himself submitted within weeks after their approval, based on his real world experiences with ICU in trying to implement the proposals.
- Recent "Unicore" mailing list discussion on the scope of enclosing marks have brought the problem into sharp relief that no matter how well the common cases are enumerated, the unclear edge cases can negatively complicate the decision after the fact.

Any member of the UTC who has been at a meeting in the last few years or has spent time on the Unicore or Unicode mailing lists could doubtlessly come up with many more.

And if the review process would both (a) require real world reference implementations prior to finalizing proposals, and (b) refuse to count absent or insufficient reviews as acceptable, we would be able to help stop these sorts of problems from happening.

Be Willing To Leave Proposals in Queue Until They Are Ready

As previously mentioned, character proposals tend to have a degree of stability and accuracy to which all other proposals should aspire. One problem that has occurred with some specific property assignments in the proposals, however, is that the assignments are made before usage is fully understood. Because of this, the UTC is forced into the same correctness vs. consistency problems due to normative requirements. The recommendation of the authors is to require more time and assurance for the assignment normative properties that cannot be changed later -- such as decomposition, combining class, and compatibility mappings.

Most importantly, if there is not enough information then the lack of feedback should not be equated with approval by default. It should be a required prerequisite to receive feedback prior to approving the proposal. Past experience has proven that the most useful feedback comes from implementers unable to use the information in its current form; holding up proposals until the information exists will allow the UTC to wait until we can be confident about the decision.

If there are no specific, known target users at the time of the addition, then one truly needs to question the timing of the addition itself -- after all, there is no harm in having one or five or a dozen almost-finished proposals waiting for approval. Having so many quality proposals that await only the final information on normative properties puts the UTC in a much better place than we would have been by rushing potentially incorrect assignments through.

The end result would be a longer, but safer, journey on the road to normativity.

UTC Should More Carefully Consider Subcommittee Requests

The Unicode Editorial Committee has on it some of the smartest and most hard working people involved with the Unicode standard, and in truth everything that they can handle, they do. When there is a policy issue that is beyond their scope and they ask the UTC for clarification, members of the UTC should work harder to understand what is being asked of them. Too often such issues later lead to problems (e.g. issues that came up after the Spring 2002 UTC, immediately prior to the 3.2 release), and the answer of "UTC was asked" is not sufficient. The EC should continue to be as hardcore about pushing back on decisions as in the past (when appropriate); the UTC must be better about paying heed to these issues.

On a similar vein, when other committees such as the Bidi Committee are giving feedback to the UTC, both sides need to be better about making sure that issues are understood well enough to make a decision and that all issues are explored as best as they possibly can.

"Not Enough Information" should mean "Not Yet Implemented"

This idea has been suggested in other solutions above but bears explicit mention since we feel it is a primary contributing factor to the single most important problem the UTC faces.

The equivalent of "motion fails for lack of a second" needs to be added to documents or ideas under review that do not receive enough participation. The principle of N.E.I. (not enough information) should cause the UTC to default to N.Y.I. (not yet implemented). The fact is that if implementers of Unicode can survive without a formal conformance model as long as they know that one is coming, then the UTC should be willing to handle less weighty matters in the same way. Note that this does not stop implementers from developing innovative ways to apply the standard prior to normative or formalized references. It should, however, help keep the UTC from developing implementation standards before it fully understands customer requirements.

The UTC needs to be much more conservative in this area than in the past. We expect that fear of proposals not advancing or even being refused may even cause the authors of future proposals to increase the quality of the work they present to the committee.

Summary

There is a lot in Unicode that is good, but if Unicode is to succeed over the next ten years there will need to be some fundamental changes in the way new work is done. This document has attempted to capture what the authors feel to be some of the largest issues that must be considered. It is hoped that this document can spur healthy discussion in this UTC meeting and the next one on what changes can be made to ensure Unicode's success for the next decade and beyond.