




[CCB-264] [Make the Line Feed \(LF\) character an allowed record delimiter](#) Created: 08/Jul/19 Updated: 04/Feb/21 Resolved: 04/Feb/21

Status:	Resolved
Project:	PDS4 Standards Change Control Board
Component/s:	Information Model , PDS Tools
Affects Version/s:	1.D.0.0
Fix Version/s:	1.G.0.0

Type:	Enhancement / Improvement	Priority:	Urgent
Reporter:	Steven Hughes	Assignee:	Mike Drum
Resolution:	Done		
Labels:	Queued_for_Implementation, Queued_for_NextBuild		
Remaining Estimate:	Not Specified		
Time Spent:	Not Specified		
Original Estimate:	Not Specified		

Attachments:	 CRLF community feedback.pdf  Line_endings_memo_orex.pdf  minimal_test_product_line_endings.zip
Problem Statement:	In the PDS4 Information Model only the Windows Carriage Return - Line Feed (CRLF) sequence is allowed as a record delimiter. However Linux/UNIX operating systems use Line Feed (LF). Any data provider using a Linux/UNIX system to prepare their data is required to convert the record delimiter in order to submit the data to the PDS. This conversion seems to be an unreasonable burden to place on data providers since Linux/UNIX operating systems are in common use and a relatively simple fix to PDS4 software would handle both record delimiters.
Proposed Solution:	Add "Line-Feed" as a permissible value to the attribute <code><*record_delimiter></code> and with value_meaning, "Records in the * are delimited by ASCII line-feed (0x0A)".
Requested Changes:	<p>Update the IM: Add "Line-Feed" as a permissible Value to the attribute <code><record_delimiter></code> and with value_meaning, "Records in the * are delimited by ASCII line-feed (0x0A)". This should apply in the following contexts:</p> <p>Checksum_Manifest Stream_Text Table_Character (and its subclasses) Table_Delimited (and its subclasses)</p> <p>Update the following documents:</p> <p>PDS Standards Reference:</p> <p>4C.1 - Amend the first sentence of item 2 to read as follows: "Each record, including the last, is followed by a record delimiter - either the ASCII carriage-return line-feed pair (denoted <code><CR><LF></code>) or just the ASCII line-feed <code><LF></code>." Add a sentence to the end of 4C.1.2: "Whichever record delimiter is used (<code><CR><LF></code> or <code><LF></code>), it must be used consistently for all records." For the examples in this section, please change a few of them to use <code><LF></code> instead of <code><CR><LF></code></p> <p>8A.3 - In the fifth paragraph, amend the first sentence to read as follows: "Records in these tables are delimited by either line-feed characters or carriage-return line-feed pairs"</p> <p>9C.1 - Amend the first sentence to read as follows: "Records in the Inventory table are delimited by either the line-feed character or carriage-return line-feed pairs; each record has two fields, delimited by a single comma.</p> <p>PDS Data Provider's Handbook:</p> <p>Amend the sentence in 13.1.2 to read as follows: "Each entry consists of [...] followed by an ASCII line-ending marker in the form of a line-feed character or carriage-return line-feed pair. The line-ending marker used must be consistent for the entire file." Amend the sentence in 13.1.3 to read as follows: "Each record is delimited by an ASCII line-ending marker in the form of a line-feed character or carriage-return line-feed pair. The line-ending marker used must be consistent for the entire file." Amend Appendix B.4.c to read as follows: "If (4) but neither (a) nor (b) and it is ASCII_Short_String_Collapsed with <code><LF></code> or <code><CR><LF></code> record delimiters, then use Product_File_Text." Amend the end of K.2.1.5 to read as follows: "The record_delimiter attribute gives the delimiter that separate lines in the text. Currently the only acceptable values are "Carriage-Return Line-Feed" and "Line-Feed".</p> <p>PDS Concepts</p> <p>9.1 - Amend the last sentence to read as follows: "PDS also allows the non-printing Tab as a field delimiter in the Table_Delimited class and requires either the Line-Feed character or Carriage-Return Line-Feed pair as the record delimiter in both Table_Delimited and</p>

	<p>Table_Character."</p> <p>11.1 - Amend the second sentence to read as follows: "Each record in a character table must end with a record delimiter consisting of either a one-byte ASCII line feed character or a two-byte pair of both the ASCII carriage return character and the ASCII line feed character."</p> <p>12.1 - Amend the second bullet to read as follows: "record_delimiter — set to either the line-feed character or the carriage-return line-feed pair"</p> <p>12.3 - Amend the end of the second paragraph to read as follows: "Fields are separated by [...] and records are separated by record delimiters (ASCII line-feed or carriage-return line-feed pairs)."</p> <p>Appendix A 'table' - Amend the definition to read as follows: "A two-dimensional data structure [...] and a record delimiter (an ASCII line-feed character or carriage-return line-feed pair)."</p>
Impact Statement:	<p>Impact:</p> <ul style="list-style-type: none"> -- Information Model -- The requested change has a minor impact on the IM. -- Standards Reference -- The requested change has a minor impact on the Standards Reference. -- DPH -- The requested change has a minor impact on the DPH. -- Concepts Document -- The requested change has a minor impact on the Concepts Document. -- PDS Tools -- The requested change has a moderate impact on the PDS Tools. The change has a moderate impact on users. <p>No Impact:</p> <ul style="list-style-type: none"> -- APG -- External Agencies -- ISO Standards -- PDS Website -- PAG
Technical Assessment:	<p>The request is to add "Line-Feed" as a permissible value to the attributes <*record_delimiter>. This change allows the use of the commonly used Linux/UNIX record-delimiter Line Feed (LF) and eliminates the need to reprocess large volumes of digital data being submitted to the PDS4 archive. This change has minimal impact on PDS4 software. Even though this change is backward compatible with regard to the system, it is not necessarily backward compatible for all users. This change request is reasonable since as argued, the benefits of this change significantly offset any negative impact.</p>
System Impact:	backwards compatible
DDWG Notes:	<p>DDWG vote item:</p> <ul style="list-style-type: none"> -- Item Passed: 10 Yes (atm, cis, en, geo, ipda, naif, ppi, rms, rs, sbn)
CCB Processing Summary:	<p>CCB Vote item 264:</p> <ul style="list-style-type: none"> -- CCB-264 PASSED: 7 Yes (ATM, CIS, GEO, IPDA, PPI, RMS, SBN);

Description

In the PDS4 Information Model only the Carriage Return - Line Feed (CRLF) sequence is allowed as a record delimiter. This sequence is used as a record delimiter in Windows operating systems and in the HTTP protocol. However the Linux/UNIX operating systems use Line Feed (LF) and these systems are in common use by many data providers. Having to convert the Linux/UNIX record delimiter to the Windows record delimiter is considered by many to be almost make-work. Allowing both record delimiters eliminates this burden and with minimal impact to PDS4 software.

Comments

Comment by [Mark S. Bentley](#) [18/Jul/19]

Yes please!! This is a constant thorn in everyone's side - on the mission I'm working out of 11 instruments (actually many more pipelines), one team uses Windows and every other pipeline is on some *nix derivative and uses LF only. In addition, forcing CRLF has only caused me the following issues:

- to archive ASCII FITS tables, forcing <CR><LF> into the data files is unsupported by most FITS libraries, and requires produced files to be hacked afterwards
- we (PSA/BepiColombo) would like to archive raw ground-station tracking files for radio science - many of these have data that can be described by a PDS4 label, but would require changing the original file to force Windows line endings, which we would prefer not to do.

Since the IM has an attribute to specify the line ending, and tools could easily be adapted to support validation etc. I would strongly support this.

Comment by [mitch gordon](#) [18/Jul/19]

The RMS Node remains opposed to this. The arguments against this proposal remain unchanged since the previous proposal, ~~CCB-90~~ was rejected by the MC.

Briefly, our concerns are:

This idea has untold potential to break pre-existing end user software. It also violates the PDS4 design goal of defining simple, predictable standards. During the initial design phase, we considered allowing multiple options for Record_Delimiter and made the decision to restrict PDS4 to

one value in order to simplify the data design as much as possible.

The proposal indicates that "Any data provider using a Linux/UNIX system to prepare their data is required to convert the record delimiter in order to submit the data to the PDS. This conversion seems to be an unreasonable burden to place on data providers ...".

In essence this is a proposal to pass that "unreasonable burden" to end users.

Here is Mark's comment from [CCB-90](#).

PDS4 is all about translating files into a user's favorite format, and line termination can be a part of that. However, assuming that our OS will take care of this for us opens up several cans of worms. For one, it breaks checksums. For another, it invalidates any record-length information in the header. Finally, as many users have seen, whether or not line terminators are translated depends on the individual software being used--some tools translate, others do not. It even varies from one web browser to another, and what a browser does is often undocumented. A PDS4 translation tool can handle all of that correctly, and such a tool really should exist. It would update the terminators, record length and checksum in a consistent way, so the user could always receive valid products with native termination for his/her own platform. However, the question on the table is how we archive, not how we deliver products to users. Every inconsistency in the archive increases long-term costs to the PDS. Archiving files with two different kinds of line termination would just make the aforementioned translation tool twice as complicated, because it would have to work in both directions. It also makes product validation more complicated. Finally, an inconsistent archive does not really solve the stated problem for Unix and Mac users, because there would still be many other products in the archive with non-native termination.

Comment by [Lyle Huber](#) [18/Jul/19]

1) "This idea has untold potential to break pre-existing end user software." This is an assertion made previously by Mark and now by you that I now have to ask for supporting evidence. I would think that any such software would only be looking at the record length and should ignore the record delimiter characters whether there are one or two of them. Whereas we do know it places a burden on providers, the only evidence of a burden on users is your sayso.

2) "During the initial design phase, we considered allowing multiple options for Record_Delimiter and made the decision to restrict PDS4 to one value in order to simplify the data design as much as possible." Not really, it was to mostly to defer this argument until a later time. There would be no point in including a Record_Delimiter attribute if we knew that there could only ever be one value for the entire existence of PDS4.

Comment by [Mark S. Bentley](#) [06/Aug/19]

I agree with Lyle - I am right now receiving a lot of development products from teams that use the "wrong" line endings, but are completely self-consistent - i.e. the record length is set right, regardless of line endings. All of the PDS software that I have tested handles this (as it should); to any reader an extra character or two after the field definitions, but inside the record length, should simply be ignored. The **only** problem is that validate plays it by the book and complains about the "wrong" line endings.

As an example, find attached two minimal observational products, one using LF line endings and one using CRLF. Both open correctly in every tool I have tried - the python reader, the java tools, transform etc. The *only* difference is that one fails validation due to the line endings

[minimal_test_product_line_endings.zip](#)

Comment by [mitch gordon](#) [30/Oct/19]

Regarding Lyle's question whether there is software that would be broken by this change:

PDS publishes a standard so that anybody in the world can program to it. Our stated design philosophy is to be "as simple as possible," among other things. Mark (Showalter) has written software assuming PDS data would conform to this point in the standards. That software will break with this change. It is likely there are others who have done the same thing, but there is no way to know who or how many until we make the change and get the complaints.

Regarding Lyle's comment regarding the previous decisions:

"Not really, it was to mostly to defer this argument until a later time. There would be no point in including a Record_Delimiter attribute if we knew that there could only ever be one value for the entire existence of PDS4."

Not really. There was no intent to defer. We included Record_Delimiter because we did not want to have "default" values - everything necessary to read and understand the data would be provided explicitly. It's the same reason we have axis_index_order, which also has only one permitted value. Similarly, "offset" and "object_length" are required to specify "unit=byte" even though no other value is permitted.

Regarding Mark's BepiColombo example:

"we (PSA/BepiColombo) would like to archive raw ground-station tracking files for radio science - many of these have data that can be described by a PDS4 label, but would require changing the original file to force Windows line endings, which we would prefer not to do."

This standard applies only to ASCII files. Are these BepiColombo tracking files in ASCII format? Historically, raw radio files have been horrifically complex and designed to be extremely compact, using bit-fields of arbitrary size all over the place. Maybe modern files are simpler, but it would still surprise me a lot if they are ASCII.

Even if the data streams are very simple now, any file containing a mixture of ASCII and binary would have to be treated as a binary product, in which case line termination would cease to be an issue.

Regarding Mark's FITS example:

"to archive ASCII FITS tables, forcing <CR><LF> into the data files is unsupported by most FITS libraries, and requires produced files to be hacked afterwards"

The FITS standard does not specify line terminators, and it has no requirement for line terminators in any context. It would seem to follow that any usage of either LF or CRLF is a "hack", and there is no reason to prefer one over another.

Is there a FITS software library that supports LF but not CRLF? If so, this would be asking us to change the PDS standards in order to accommodate faulty programming logic in a FITS implementation.

Finally, I will point out that a COTS tool to modify line terminators, `dos2unix`, which also includes `unix2dos` already exists. It comes in most Linux distributions, and is available for download here: <http://dos2unix.sourceforge.net/>.

Yes, this would require an extra step in a provider's pipeline, but surely it is not that egregious.

Comment by [mitch gordon](#) [30/Oct/19]

Sorry, I hit the add button prematurely.

I acknowledge that since FITS records are required to be fixed-length 2880 bytes, the use of `unix2dos` and `dos2unix` is problematic in the context of FITS ASCII table files which haven't been padded to reach 2880.

Comment by [Lyle Huber](#) [30/Oct/19]

Please see <https://devblogs.microsoft.com/commandline/extended-eol-in-notepad/> to note that even Microsoft recognizes what already exists.

Comment by [mitch gordon](#) [05/Nov/19]

Thanks Lyle, but I don't see Microsoft as an archiving agency. Simple and consistent standards are a virtue in and of themselves.

Just an observation that has never occurred to me before. The tech assessment considers the impact of this proposal on PDS, but completely ignores its potential larger scale impact. The term "backwards compatible" may be formally correct from the standpoint of the PDS, because all pre-existing PDS4 products will continue to be PDS4-compliant. However, this change is not backwards compatible to users. Perhaps future technical assessments should also include a clear statement about the potential impacts to users.

Comment by [Mark S. Bentley](#) [21/Nov/19]

Thanks @mgorden for your reply to my comments - yes, regarding FITS you are quite right - apologies for my mistake. Regarding the groundstation files, if I recall correctly the ESTRACK TTCP closed-loop files are ASCII whilst the open-loop are indeed binary.

Comment by [Edward Guinness](#) [13/Dec/19]

The previous change request on this issue ([CCB-99](#)) requested that Line Feed be allowed as the record delimiter for `Stream_Text`, `Table_Character`, and `Table_Delimited`. However, `record_delimiter` is used in several other classes based on searching the 1.D.0.0 Information Model html file. The most common of which is the `Inventory` class used in collection products. Allowing Line Feed in the `Inventory` class would have an impact on the `validate` tool (I don't know how much). `Record_delimiter` is also used in the following classes - `Checksum_Manifest`, `Manifest_SIP_Deep_Archive`, and `Transfer_Manifest`. I don't know for sure, but these may have an impact on the NSSDCA deep archive transfer process.

Comment by [Richard Simpson](#) [18/Dec/19]

To be clear, Ed has found that `record_delimiter` is specified in `Stream_Text`, `Table_Character`, `Table_Delimited`, `Inventory`, `Checksum_Manifest`, `Manifest_SIP_Archive`, and `Transfer_Manifest`. In each case, we presently require that `record_delimiter` be Carriage-Return Line-Feed. If we change it to something else (or allow an alternate delimiter) in some cases, it's going to be really confusing if we don't change it in all cases.

Comment by [Richard Simpson](#) [08/Jan/20]

1. If [CCB-264](#) goes forward, Standards Reference 4C.1 should probably be changed, since it is the DSV standard and requires `<CR><LF>`.
2. I concede that `<LF>` has become a widely accepted standard for record delimiters. But literally, `<LF>` only moves the cursor to the next line. The `<CR>` returns the cursor to the start of line; so the combination is what is important, and that is one reason the pair was adopted by systems which are no longer widely used. If we kept `<CR><LF>` we ensure that there is no ambiguity.
3. I'm not convinced that converting `<LF>` to `<CR><LF>` is a lot of work; I do it all the time, and it's usually a single line of code added to a piece of archiving software.

Comment by [Lyle Huber](#) [15/Jan/20]

As part of my action item, I have identified the following areas in the documents that need to be edited.

PDS SR 4C.1 - Change item 2 to read as follows: "Each record, including the last, is followed by a record delimiter - either the ASCII carriage-return line-feed pair (denoted `<CR><LF>`) or just the ASCII line-feed `<LF>`."

PDS DPH - No changes.

PDS Concepts - Changes need to be made to sections 9.1, 11.1, 12, 12.1 and 12.3 as well as Appendix A - table. These should be self-evident to the editor.

Comment by [Edward Guinness](#) [16/Jan/20]

Lyle - There is mention of record delimiter in the DPH. Sections 13.1.2 and 13.1.3 imply `<CR><LF>` pair is needed. Also, the pair is noted in Appendix B item 4 and in Appendix K, section K.2.1.5 at the end of the section.

Comment by [Lyle Huber](#) [16/Jan/20]

Ed is correct - Ron and I both missed those. Change the PDS DPH modifications to those listed by Ed.

Comment by [Trent Hare](#) [30/Jan/20]

While I am not horribly opposed to this change (meaning to allowing <LF>), this update will most likely break our few current readers. They will need to be updated to query the <record_delimiter> attribute before proceeding.

If I am following the code correctly it does look like [CCB-264](#) will break PDS4_Viewer since <CR><LF> appears to be hardwired:

https://github.com/Small-Bodies-Node/pds4_tools/blob/89b652168e864e4f7d29925415a8f2c40afd2172/pds4_tools/reader/read_tables.py#L858

GDAL has some checks for <record_delimiter> but would need updates too.

(updated): <https://github.com/OSGeo/gdal/blob/109214da6f4964557ac88d4cf9e199d6eeacc30e/gdal/frmts/pds/pds4vector.cpp#L747>

Recall this also impacts any record length calculations:

<https://github.com/OSGeo/gdal/blob/109214da6f4964557ac88d4cf9e199d6eeacc30e/gdal/frmts/pds/pds4vector.cpp#L1506>

Looks like pds4-jparser will need updates too:

https://github.com/NASA-PDS-Incubator/pds4-jparser/search?q=CARRIAGE_RETURN_LINE_FEED&unscoped_q=CARRIAGE_RETURN_LINE_FEED

food for thought...

Comment by [Mark S. Bentley](#) [30/Jan/20]

That's interesting @Trent - I tried a few tests with a modified product and most tools seemed to work fine, assuming that the record length was updated to account for the one-character difference. In fact validate was the only case that tripped my test up. Perhaps I got lucky! 😊

Comment by [Matt Tiscareno](#) [30/Jan/20]

We acknowledge the reality that data providers are annoyed by this requirement. The correct response is to do a better job of managing expectations, rather than to loosen the standard.

The passage in the DPH that describes this requirement should include a summary of the reasons for it. Dick has already articulated this in his comment above. I have slightly modified his language in the following proposed addition to the DPH:

Although <LF> has become a widely accepted standard for record delimiters, what it literally does is only move the cursor to the next line. The <CR> additionally returns the cursor to the start of line. Although most contemporary systems require only the LF, many legacy systems and tools require the combination. We maintain this standard in order to ensure that there is no ambiguity.

The global replacement of <LF> with <CR><LF> in a file generally involves a single line of code. Here are some example code fragments:

```
<unix> <python> <IDL> <C++>
```

Trent's exercise with the PDS4 Viewer demonstrates that even our own tools may have the CR-LF standard embedded within them, and that changing the standard would require us to search out possible conflicts and fix them. Doesn't it seem very likely that our users have similar tools that would be similarly impacted?

Although I have some sympathy with data providers who would rather not bother with a global replacement of <LF> with <CR><LF>, maintaining that small burden upon them seems to have much less negative impact than loosening the standard.

Comment by [Jordan Padams](#) [10/Feb/20]

I am late to this conversation, but as a response to :

The global replacement of <LF> with <CR><LF> in a file generally involves a single line of code. Here are some example code fragments:

```
<unix> <python> <IDL> <C++>
```

This argument holds true for updating existing software as well.

If software tools are identified that require update, EN has 2 weeks dedicated in the schedule to make necessary updates to those tools. These can include discipline node tools if they do not have the resources to make the one-line update to their code.

Comment by [Jordan Padams](#) [10/Feb/20]

Additionally, per this comment:

Trent's exercise with the PDS4 Viewer demonstrates that even our own tools may have the CR-LF standard embedded within them, and that changing the standard would require us to search out possible conflicts and fix them. Doesn't it seem very likely that our users have similar tools that would be similarly impacted?

PDS4 Viewer, PDS View, and pds4_tools have all been identified as PDS tools that need to be updated, and our schedule includes a timeframe to make these applicable updates.

Comment by [Lyle Huber](#) [11/Feb/20]

This statement "Although <LF> has become a widely accepted standard for record delimiters, what it literally does is only move the cursor to the next line. The <CR> additionally returns the cursor to the start of line. Although most contemporary systems require only the LF, many legacy systems and tools require the combination. We maintain this standard in order to ensure that there is no ambiguity." is not really accurate and has not been so for a couple of decades. The <LF> character that is typically represented by \n does not "literally only move the cursor to the next line". The \n in fact is taken to mean "newline" which both moves the cursor to the next line and to the start of the line.

Comment by [Steven Hughes](#) [11/Feb/20]

"During the period of 1963 to 1968, the ISO draft standards supported the use of either CR+LF or LF alone as a newline, while the ASA drafts supported only CR+LF." <https://en.wikipedia.org/wiki/Newline>

Comment by [Richard Simpson](#) [11/Feb/20]

Steve is right. The original interpretation came from the teletype era in which <LF> simply moved the printer to the next line and continued printing from the same column position. A single <CR> moved the carriage to the beginning of the line where it could overprint. The teletype needed both <LF> and <CR> to move the carriage to the left margin *and* step to the next line. Early electronic displays adopted the same conventions; only later did <CR> become implicit.

Comment by [Steven Hughes](#) [13/Feb/20]

The ISO draft standard helps refocus the issue. The object being modeled is newline. It has several aliases including the terms "record delimiter", "CRLF", and "LF". It has two commonly accepted values, namely the character sequences <CR><LF> and <LF>.

The definition of newline is widely accepted and is modeled in the IM as "record_delimiter". However, the model is only half correct. The most common value of newline has been omitted due to software engineering concerns. Allowing software engineering concerns to impact the model contradicts our original goal of a model-driven architecture.

A foundational principle for the PDS4 Information Model (IM) is that it remains independent from its implementation. Its purpose is to provide a consistent set of informational requirements that drive the design and development of PDS software and services. This suggests that the model designers should be implementing technology agnostic, as much as possible, and leave the details to the software engineers. This then further suggests that software and services should be written to respond to the information model. This principle helps disentangle the impact of changes to the model that are due to community evolution from software changes that are due to implementing technology evolution.

Comment by [Trent Hare](#) [12/Mar/20]

From: Steven Joy <sjoy@igpp.ucla.edu>
Sent: Wednesday, February 26, 2020 4:24 PM
Subject: comments on [CCB-264](#)

The CCB met on 2/6/2020 to discuss the merits of and issues with SCR-264. A majority agreed that there is probably a need for PDS to evolve in this direction, however, there was no compelling need case made for time criticality. As such, we would like clarification on several key points before considering this change again.

If there is a compelling or time-critical need for this change, that should be fully spelled out in the change request. CCB members demonstrated that several existing PDS tools would need to be updated in order to implement this change. An unknown number of user tools would potentially be impacted as well. Therefore, the impact assessment included with the change request was inadequate. The CCB felt that there should be a thorough assessment of the impact on tools, internal and external, provided with the request when it is sent to the CCB the next time. This assessment should include an estimate of how much effort would be required to update the PDS tools and state when the updated tools would be available to the community.

Several CCB members felt that this change might require updating the PDS4 standard to V2 and not just pushed out in a point release as it represents a major change in what is allowed by PDS as a whole. A discussion of the issue of PDS standards versioning should be included when this SCR is presented again to the CCB. Furthermore, the DDWG might consider holding this change in reserve, along with other major changes, for the eventual update to V2.

The CCB sympathizes with data providers who are inconvenienced by the current requirement. In order to better manage expectations and communicate the reason for the requirement, we agreed that it would be good to update the DPH. By way of suggestion, the DDWG might consider the JIRA comment entered at 10:01am Pacific, 30/Jan/20. In addition to that draft language, perhaps the DPH should mention that it is advisable for tools to be made agnostic regarding LF and CR/LF, as the standard might be changed in the future.

Comment by [Mike Drum](#) [05/Aug/20]

OSIRIS-REx mission would like us to expedite the re-submission of this ticket to the CCB, because they have data that is ready to be archived with LF terminators. The special sauce here is that the data files are being generated by DSN and NAIF without CR/LF; the Orex team is not modifying them, and I believe rightfully thinks it's a little silly that data from NASA can't be archived with NASA.

I've attached a memo from that team that gives a bit more detail, and formalizes the request. It's on the agenda to be discussed in DDWG on 8/6: [Line_endings_memo_orex.pdf](#)

Comment by [Richard Simpson](#) [05/Aug/20]

~~CCB-264~~ was discussed briefly on July 2; the plan continues to be for DNs to survey their user communities and report on software impacts some time in the November time frame.

Kate Crombie's memo was written in late July. An OSIRIS-REx radio science review was held on August 3, and the issues behind ~~CCB-264~~ came up. The lien report from that review includes the following:

"32. Current PDS standards require that ASCII tables have <CR><LF> delimiters, but TRO, ION, and SFF products have <LF> delimiters. An SCR to allow <LF> delimiters is currently being proposed. Resolve with the following plan:

"- If, by the end of 2020, this SCR has passed, the data with <LF> delimiters will be appropriately labelled in the new standard. Note that the new IM will not be released until March of 2021. (If possible, the entire O-Rex raw RS bundle will be labelled in the same version of the IM, or if this presents problems, only the TRO, ION, and SFF collections can be labelled in the new version of the IM.)

"- If the SCR has not passed by the end of 2020, SBN will take on the task of converting the TRO, ION, and SFF data files to <CR><LF> delimiters, and making the necessary changes to the labels."

Thus, O-REx and SBN have agreed to wait until the end of the year, which fits nicely with the DDWG schedule for collecting user impact information and discussing ~~CCB-264~~. ~~CCB-264~~ is on track, and O-REx does not need special attention.

Comment by [Richard Simpson](#) [06/Dec/20]

I conducted a survey of the radio science community using lists provided by two other people. I didn't have access to either list; but my own list is completely out of date, and I decided not to try to create a new one for this survey.

Each list probably covered 80-120 people; many were likely duplicates. Fourteen people responded to the survey; twelve said there would be no impact if ~~CCB-264~~ were approved, and two said the impact would be low (because they might change delimiters in their archiving software). A couple felt the change would be advantageous.

Comment by [Richard Simpson](#) [06/Jan/21]

I have the following suggestions after the revisions earlier today.

PDS SR 4C.1 (2): Add examples with only the <LF> delimiter, perhaps in a second column.

PDS SR 4C.1 (2): The added sentence should read "... it must be used ..." The word "should" is too permissive in this context.

Concepts 11.1: The amended sentence should read "Each record in a character table must end with a record delimiter consisting of either a one-byte ASCII line feed character or a two-byte pair of both the ASCII carriage return character and the ASCII line feed character." The original was missing three articles "a".

Comment by [Mike Drum](#) [07/Jan/21]

PDS SR 4C.1 (2): The added sentence should read "... it must be used ..." The word "should" is too permissive in this context.

Concepts 11.1: The amended sentence should read "Each record in a character table must end with a record delimiter consisting of either a one-byte ASCII line feed character or a two-byte pair of both the ASCII carriage return character and the ASCII line feed character." The original was missing three articles "a".

I've incorporated these suggestions, thanks!

PDS SR 4C.1 (2): Add examples with only the <LF> delimiter, perhaps in a second column.

I agree those would add clarity. Not sure the best way to go about it, or if it's completely necessary though, because those same examples only use a comma-delimiter rather than showing ever permutation of field delimiters. Perhaps we could suggest mixing the delimiters in across all the various examples?

Comment by [Richard Simpson](#) [07/Jan/21]

These are details, but details are important when setting standards.

In SR 4C.1 the new wording says "Whichever record delimiter is used (<CR><LF> or <LF>), it must be used consistently for the entire product." But 4C.1 is for generic spreadsheets, which become products only within the context of PDS4. The last four words should be dropped. But the latest wording "for all records" is equally good.

There is a related issue with the proposed new wording for DPH 13.1.2 and 13.1.3 which says that the record delimiter must be consistent throughout that file; that's OK because we're talking about data objects that are files. But, if there's another discussion of consistency for character or delimited tables, we need to be careful. If there are two or more tables in a single file, they can have different record delimiters.

Comment by [Trent Hare](#) [08/Jan/21]

Personally I have not heard anyone strongly against this change from the community. It will require updates to the existing GDAL library (read/write), which I will make sure happens. Obviously, the reader will be updated to accept either style. The current writer is locked to <CR><LF> and I am on the fence if the user should be provided an option for which flavor they want during table/vector conversion.

Comment by [mitch gordon](#) [04/Feb/21]

Okay this is now officially closed, but someone needs to let the user and data provider communities know that it is coming in the spring build - with an approximate date (June 1?) and appropriate IM version). Who is responsible for doing this?

Comment by [Mike Drum](#) [04/Feb/21]

I plan to update the [initial FAQ page](#) we're hosting at SBN PSI with details about how users might need to update their tools, as well as a date and version number. We can then send this out through the PEN and OpenPlanetary as well.

We also need to update our own tools within the PDS though. Not sure how much auditing of that there's been.

Comment by [Tom Stein](#) [04/Feb/21]

In addition to announcements like Mike mentions, "we" might consider:

- Call out this ([CCB-264](#)) change explicitly in the release notes with a paragraph of text in addition to the normal entries that go into the release notes.
- Add some text to the data standards page that reminds tool providers to check the release notes for breaking changes.
- Create a new mailing list for tool providers and other interested parties to receive release notifications including release highlights and breaking changes. (There might already be such a list and I simply am not aware.)
- Provide a single page list of release notes for all releases so a developer does not have to hunt around through scads of release notes pages.
- The release notes are terse. They should include links to details of the changes to the IM to help developers understand any limits or qualifications.

Generated at Thu Feb 04 16:10:16 PST 2021 by Jordan Padams using Jira 8.14.0#814001-sha1:ab08d3d2b500818bdad5c85d636b509d4cade801.