Open letter to the members of the ASPRS LAS committee

"Please do not approve the current proposal of the LAS 1.4 specification. It is broken."

The proposed LAS 1.4 specification breaks the forward / backward compatibility of the LAS 1.X family and unnecessarily increases file sizes. I bring this to your attention only after countless discussions with Lewis Graham and other members of the ASPRS committee that were more or less ignored in the end. The LAS format has de-facto become an international standard without ever undergoing a slow (and painful) standardization process. The ASPRS committee and all those involved in designing the LAS specification have done a great job and are respected for their work. The proposed LAS 1.4 specification will blemish these accomplishments, undermine the credibility of LAS as a standard, and diminish trust in ASPRS' ability to act in such a role.

All previously approved LAS 1.X specifications were both forward and backwards compatible. For example, a LAS 1.1 reader was able to read a LAS 1.3 file as long as it only contained points of type 1. And similarly, a LAS 1.2 reader was able to read a LAS 1.0 file. The most current LAS specification was always a complete specification of the entire LAS 1.X family. LAS 1.4 breaks with this tradition. This is completely unnecessarily and could easily be avoided by rearranging a few header entries (as outlined later). In its current form the LAS 1.4 specification is a nightmare for programmers and users of LAS alike, because …

- removing support for point types 0 and 5 in LAS 1.4 would do a great disservice to the community. People want to store more than 4 billion LIDAR points in LAS format. Many are hoping that LAS 1.4 will allow them to merge collections of LAS or ASCII content into a single file. However, the proposed LAS 1.4 specification would force them to use the point type 6 (29 bytes) instead of the point type 0 (20 bytes). This adds a lot of zeros that increase the size (of already Gigabyte-sized files) by almost 50 percent.

- no existing LAS reader will be able to read a LAS 1.4 file - even if it contains less than 4 billion points of only point types 0 through 5. The proposed LAS 1.4 specification breaks the forward / backward compatibility of the LAS 1.X family due to an incompatible header.

- every LAS importer/exporter will from now on be required to implement at least two different LAS specifications in order to support current and future LAS content. For example, after reading a LAS 1.2 file the software will have to "remember" that the original version of this file was 1.2 and then chose the appropriate LAS 1.2 writer on output. Such "special handling" was not necessary in previous LAS specifications.

  An example: Hans Meier has 1 TB of LAS 1.1 files and wants to classify all ground points with a brand-new software that uses the LAS 1.4 specification to read and write LAS files. The classified files - although they only contain LAS 1.1 content - will therefore be in LAS 1.4 format and Hans will not be able to read them with his LAS 1.2 software.

  This is a real-world scenario! Why? Because …

  - when LP360 from Qcoherent processes a LAS 1.0 file, the result is stored as a LAS 1.3 file with LAS 1.0 content.
  - when LAStools converts xyz coordinates from ASCII to LAS it really only creates LAS 1.0 content but the result is stored as a LAS 1.2 file.

  This had never been a problem as an increment in version number had not made the content unreadable for earlier versions. But with the proposed changes in LAS 1.4 this become a major headache. Programmers that want their software to cater clients with LAS 1.0 to LAS 1.4 will have to read and implement at least two LAS specifications.

What needs to be fixed?

- The LAS 1.4 header needs to remain backward / forward compatible.
    - All new 64 bit fields should be appended to the current LAS 1.3 header. This means some fields - like the “number of point records" field - would be replicated and exist in 32 bit as part of the LAS 1.0 – 1.3 header and in 64 bit as part of the LAS 1.4 header (see tables below for details). We then use three simple rules:

        A. If the value of the 32 bit “offset to point data” is set to zero we use the 64 bit "extended offset to point data" field.

        B. If the value of the 32 bit "number of point records" field is set to zero we use the 64 bit "extended number of point records" field and also the first five (5) "extended number of points by return" fields.

        C. If in addition the "point data format" is set to a value larger than 5 then we use all fifteen (15) "extended number of points by return" fields.
    - The LAS 1.4 file has exclusive LAS 1.4 features and requires a version 1.4 or higher LAS reader only when one or more of A , B, or C are true.
    - A “lazy” LAS 1.4 writer could simply always set the 32 bit “offset to point data” and the 32 bit “offset to point data" fields to zero and only use the extended 64 bit versions of these fields.
    - A more “clever” LAS 1.4 writer would first check whether the total number of points is less than 4 billion and whether the point type is between 0 and 5 in which case the extended 64 bit fields are not needed. Now the old 32 bit fields can be used and the content will be readable by older LAS readers as well.
- The VLR header needs to remain backward / forward compatible.
    - Simple. Add a “number of extended VLRs” to the LAS 1.4 header and have the new extended VLRs follow the regular VLRs. A LAS 1.4 writer would only use extended VLRs for very large payloads like pyramid or indexing schemes that would then only be available to a LAS 1.4 (or higher) reader and use regular VLRs for things like projection VLRs, … that any LAS reader can access. The extended VLR concept is already part of the LAS 1.3 specification. Now they would also be allowed to occur before the point data block.
- Minor alignment/performance issue (is it minor?): Increasing the basic point structure by 1 byte from 20 to 21 bytes plus the mandatory 8 byte double-precision gpstime results in a 29 byte structure. Will this result in performance penalties and during read and write access due to poor byte alignment? Not a deal-breaker but to keep in mind for LAS 1.5.

The exact technical details on how to fix the LAS 1.4 header.

I first report the format of the header for LAS version 1.0, 1.1, and 1.2 and the format of the header for LAS version 1.3 as they are in use today. Then I report the format of the header as it is proposed for LAS version 1.4 that breaks with the LAS 1.X family. Finally I describe a minor modification of the header that - simply by replicating 7 unsigned 32 bit integers - achieves the goals of LAS 1.4 without breaking the specification.

Format of header for LAS version 1.0, 1.1, and 1.2 (227 bytes)

Item	**Format**	**Size**	**Required**
File Signature (“LASF”)	char[4]	4 bytes	*
File Source ID	unsigned short	2 bytes	*
Global Encoding	unsigned short	2 bytes	*
Project ID - GUID data 1	unsigned long	4 bytes	
Project ID - GUID data 2	unsigned short	2 byte	
Project ID - GUID data 3	unsigned short	2 byte	
Project ID - GUID data 4	unsigned char[8]	8 bytes	
Version Major	unsigned char	1 byte	*
Version Minor	unsigned char	1 byte	*
System Identifier	char[32]	32 bytes	*
Generating Software	char[32]	32 bytes	*
File Creation Day of Year	unsigned short	2 bytes	*
File Creation Year	unsigned short	2 bytes	*
Header Size (must be at least 227)	unsigned short	2 bytes	*
Offset to point data	unsigned long	4 bytes	*
Number of Variable Length Records	unsigned long	4 bytes	*
Point Data Format ID (0-99 for spec)	unsigned char	1 byte	*
Point Data Record Length	unsigned short	2 bytes	*
Number of point records	unsigned long	4 bytes	*
Number of points by return	unsigned long[5]	20 bytes	*
X scale factor	double	8 bytes	*
Y scale factor	double	8 bytes	*
Z scale factor	double	8 bytes	*
X offset	double	8 bytes	*
Y offset	double	8 bytes	*
Z offset	double	8 bytes	*
Max X	double	8 bytes	*
Min X	double	8 bytes	*
Max Y	double	8 bytes	*
Min Y	double	8 bytes	*
Max Z	double	8 bytes	*
Min Z	double	8 bytes	*

Format of header for LAS version 1.3 (235 bytes, first 227 are compatible to LAS 1.0 - 1.2)			
Item	**Format**	**Size**	**Required**
File Signature (“LASF”)	char[4]	4 bytes	*
File Source ID	unsigned short	2 bytes	*
Global Encoding	unsigned short	2 bytes	*
Project ID - GUID data 1	unsigned long	4 bytes	
Project ID - GUID data 2	unsigned short	2 byte	
Project ID - GUID data 3	unsigned short	2 byte	
Project ID - GUID data 4	unsigned char[8]	8 bytes	
Version Major	unsigned char	1 byte	*
Version Minor	unsigned char	1 byte	*
System Identifier	char[32]	32 bytes	*
Generating Software	char[32]	32 bytes	*
File Creation Day of Year	unsigned short	2 bytes	*
File Creation Year	unsigned short	2 bytes	*
Header Size (must be at least 235)	unsigned short	2 bytes	*
Offset to point data	unsigned long	4 bytes	*
Number of Variable Length Records	unsigned long	4 bytes	*
Point Data Format ID (0-99 for spec)	unsigned char	1 byte	*
Point Data Record Length	unsigned short	2 bytes	*
Number of point records	unsigned long	4 bytes	*
Number of points by return	unsigned long[5]	20 bytes	*
X scale factor	double	8 bytes	*
Y scale factor	double	8 bytes	*
Z scale factor	double	8 bytes	*
X offset	double	8 bytes	*
Y offset	double	8 bytes	*
Z offset	double	8 bytes	*
Max X	double	8 bytes	*
Min X	double	8 bytes	*
Max Y	double	8 bytes	*
Min Y	double	8 bytes	*
Max Z	double	8 bytes	*
Min Z	double	8 bytes	*
Start of Waveform Data Packet Record	unsigned long long	8 bytes	*

Proposed (broken) header format for LAS 1.4 (343 bytes, not compatible to LAS 1.0 - 1.3)
The fields that break the compatibility to the LAS 1.X family are marked in red.

Item	**Format**	**Size**	**Required**
File Signature (“LASF”)	char[4]	4 bytes	*
File Source ID	unsigned short	2 bytes	*
Global Encoding	unsigned short	2 bytes	*
Project ID - GUID data 1	unsigned long	4 bytes	
Project ID - GUID data 2	unsigned short	2 byte	
Project ID - GUID data 3	unsigned short	2 byte	
Project ID - GUID data 4	unsigned char[8]	8 bytes	
Version Major	unsigned char	1 byte	*
Version Minor	unsigned char	1 byte	*
System Identifier	char[32]	32 bytes	*
Generating Software	char[32]	32 bytes	*
File Creation Day of Year	unsigned short	2 bytes	*
File Creation Year	unsigned short	2 bytes	*
Header Size	unsigned short	2 bytes	*
Offset to point data	unsigned long long	8 bytes	*
Number of Variable Length Records	unsigned long	4 bytes	*
Point Data Format ID (0-99 for spec)	unsigned char	1 byte	*
Point Data Record Length	unsigned short	2 bytes	*
Number of point records	unsigned long long	8 bytes	*
Number of points by return	unsigned long long[15]	120 bytes	*
X scale factor	double	8 bytes	*
Y scale factor	double	8 bytes	*
Z scale factor	double	8 bytes	*
X offset	double	8 bytes	*
Y offset	double	8 bytes	*
Z offset	double	8 bytes	*
Max X	double	8 bytes	*
Min X	double	8 bytes	*
Max Y	double	8 bytes	*
Min Y	double	8 bytes	*
Max Z	double	8 bytes	*
Min Z	double	8 bytes	*
Start of Waveform Data Packet Record	unsigned long long	8 bytes	*

Fixed header format for LAS 1.4 (375 bytes, first 227/235 are compatible to LAS 1.0 - 1.3)
The replicated fields that assure the compatibility with the LAS 1.X family are marked in green.

Item	Format	Size	Required
File Signature (“LASF”)	char[4]	4 bytes	*
File Source ID	unsigned short	2 bytes	*
Global Encoding	unsigned short	2 bytes	*
Project ID - GUID data 1	unsigned long	4 bytes	
Project ID - GUID data 2	unsigned short	2 byte	
Project ID - GUID data 3	unsigned short	2 byte	
Project ID - GUID data 4	unsigned char[8]	8 bytes	
Version Major	unsigned char	1 byte	*
Version Minor	unsigned char	1 byte	*
System Identifier	char[32]	32 bytes	*
Generating Software	char[32]	32 bytes	*
File Creation Day of Year	unsigned short	2 bytes	*
File Creation Year	unsigned short	2 bytes	*
Header Size (must be at least 375)	unsigned short	2 bytes	*
Offset to point data	unsigned long	4 bytes	*
Number of Variable Length Records	unsigned long	4 bytes	*
Point Data Format ID (0-99 for spec)	unsigned char	1 byte	*
Point Data Record Length	unsigned short	2 bytes	*
Number of point records	unsigned long	4 bytes	*
Number of points by return	unsigned long[5]	20 bytes	*
X scale factor	double	8 bytes	*
Y scale factor	double	8 bytes	*
Z scale factor	double	8 bytes	*
X offset	double	8 bytes	*
Y offset	double	8 bytes	*
Z offset	double	8 bytes	*
Max X	double	8 bytes	*
Min X	double	8 bytes	*
Max Y	double	8 bytes	*
Min Y	double	8 bytes	*
Max Z	double	8 bytes	*
Min Z	double	8 bytes	*
Start of Waveform Data Packet Record	unsigned long long	8 bytes	*
Extended Offset to point data	unsigned long long	8 bytes	*
Extended Number of point records	unsigned long long	8 bytes	*
Extended Number of points by return	unsigned long long[15]	120 bytes	*
Number of Extended Variable Length Records	unsigned long	4 bytes	*

Thank you for your attention in this matter.

Regards,

[illegible]

Sommerhausen, June 26th 2011

Martin @lastools

martin.isenburg@gmail.com
http://www.cs.unc.edu/~isenburg/lastools/

creator of LAStools, LASzip, and TIN streaming