# ASTM E57 3D Imaging Systems Data Exchange Format Reference Implementation

*Beta Test Program Instructions*

## 1 Overview

The ASTM E57.04 Data Interoperability subcommittee is developing a standard data exchange format for 3D imaging systems. The E57 file format specified by the standard provides a compact, vendor-neutral format for storing point clouds, images, and metadata produced by 3D imaging systems, such as laser scanners. The subcommittee has also developed an open source implementation of the standard that provides an interface and tools for reading, writing, and validating E57 files. The purpose of this beta test program is to ensure the usability of the “E57 Foundation API” and to obtain feedback on the format’s practical use that can be incorporated into future versions of the standard.

The reference implementation source code does not belong to ASTM. It has been developed by authors of the standard in a voluntary effort and as such shall not put the authors under an obligation to deliver anything.

## 2 Beta Test Process

The beta test program is aimed primarily at vendors of 3D imaging systems or 3D imaging software. Most beta testers will already have some established, vendor-specific file format and/or software that processes or visualizes 3D imaging system data. Beta testers will be asked to evaluate the process of writing their existing data into E57 files using the “E57 Foundation API”, and to evaluate the process of reading E57 files into their existing data structures or file format, again using the “E57 Foundation API”.

## 3 Registering as a Beta Tester

To register for beta testing, send email to Roland Schwarz (roland_schwarz@users.sourceforge.net).

Please indicate:

- The name of your company (if applicable).
- The name and email address for the beta test contact person at your company. Correspondence about the beta test program will be sent to this person.
- A brief summary of the software and programming environment that you intend to use in the beta testing program (e.g., which compiler, vendor software, programming language, etc.). Note that the “E57 Foundation API” is compatible with C++, but support for additional languages is planned for the future.

Beta testing is a voluntary effort, and as such beta testers are not required to deliver anything. Beta testers are however required to put contributions to the source code under the same open source license as the original code. Any Information exchanged during the testing process may be made available to the public.

## 4 Accessing the Reference API

During beta testing, the reference implementation of the “E57 Foundation API” will be available via the sourceforge project site at:

http://sourceforge.net/downloads/e57-3d-imgfmt/

The software may be downloaded as a zip file, or it can be accessed using subversion. Instructions for using subversion can be found on the project site. If you are not planning to make contributions to the source, the use of the zip files is recommended.

## 5 Communications During Beta Testing

Beta testers will be automatically added to a public mailing list when they register. News and announcements about the beta testing will be sent to all beta testers using this list. Archives of the list and a list management interface can be accessed at:

http://lists.sourceforge.net/lists/listinfo/e57-3d-imgfmt-devel

Beta testers are expected to communicate by sending emails to the list address:

e57-3d-imgfmt-devel@lists.sourceforge.net

New versions of the "E57 Foundation API" will be announced on the mailing list. We will use version numbers in the format "major.minor.build," where the major number will be 0 during the beta test phase, the minor number will be incremented with each new version for testing, and the build number will be used for internal purposes by the developers. This is all you need to know about the numbering scheme during beta testing. Extended information is available at the end of the document.

Bug reports may be sent to the list or can be entered directly via the tracker interface at the project site at sourceforge. The latter method requires obtaining a sourceforge id first.

Software modifications, such as bug fixes, may be sent to the mailing list. The preferred format is in diff format against a known subversion revision number. If you do not know what this means, simply contact via the list where the developers will try to help.

A list of known bugs, and their status will be kept at the "Bug Reports Tracker" on the sourceforge project site.

## 6 Phases

This section gives detailed information about the development cycles. The information can be safely skipped on the first reading and is not required for understanding beta testing. The information is of interest if you want to contribute to the code base and to understand the version numbering system once the code is in the production-stable state.

### *6.1 Alpha Phase (Writing)*

This is the phase of writing of the initial piece of the reference implementation. The authors shall provide a document of the minimal features that must have been implemented to begin with formal testing, i.e. entering the beta phase. Examples for features are certain aspects of the API such as the availability of compressors or ability to support custom namespaces, the ability to check the XML syntax and general file validity, to name a few examples. The support of certain platform types such as 64 bit or Linux as well as the support of compilers that are expected to work shall be defined.

No explicit version numbering is required at this stage, as no files will be released in the download section accessible to the non developing user. Versioning is implicit by making use of a source code management system, specifically *subversion*[1].

An integral part of creating the software is the writing of an appropriate user level documentation.

In parallel to software writing interested parties may register for participation in the beta phase testing plan.

### 6.2 Beta Phase (Beta Test Program)

The purpose of the beta phase is to assert the correctness of the software implementation with respect to the standard and to provide feedback to the developers about usability of the programming interface. The correctness will be governed by the "*API test plan and test strategy"* document. The usability will be judged by the beta test participants. The interface and implementation may be changed during this process. In particular there are no requirements for API[2] stability during the beta phase.

During beta phase a version numbering system will be established. Version identifiers (m.m.b) are made of three parts separated by dots:

```
major.minor.build
```

The major number is 0 during the beta phase. The minor number shall be incremented for each step of the beta phase that needs testing by the beta testers group. The build identifier will identify a unique version within the source code management system. A logical setting for this identifier is to use the string that can be obtained with the *svnversion* command. It is recommended to invoke this command as part of the software build process.

During beta phase the reference implementation will be made available as a library. This may be in any form the authors deem useful. Examples are prebuilt binaries, header files and associated documentation or tools. The library files will be made available as a packed and compressed file in the download area of the project site. Part of the name of such a file must be the complete 3 part (m.m.b) identifier.

The end of beta phase is indicated by advancing the major number to 1.

### 6.3 Release Phase (Maintenance)

The release phase is active during the rest of the lifetime of the reference implementation software. The version numbering during this phase is governed by the following rules:

- The build identifier is the value of the subversion version id. It's purpose is to uniquely identify the source code belonging to a certain library version.
- The minor number is advanced for any substantial changes beyond fixing of simple bugs, as long as the API stays backward compatible.

Changes that imply changes to the semantic of the API or otherwise will break current applications need to increment the major number. Plain extensions to the API, e.g. adding new functions will not necessarily break compatibility and as such might not require incrementing the major number.

[1] Subversion: a source code management system. A recommended tool for the windows operating system is TortoiseSVN from http://tortoisesvn.tigris.org/

[2] API: Application Programming Interface