



The Virtual Data Center Tagged-Format Tool— Introduction and Executive Summary

By John R. Evans¹, Melinda Squibb², Christopher D. Stephens¹, W.U. Savage¹, Hamid Haddadi³, Charles A. Kircher⁴, and Mahmoud M. Hachem⁵

Open-File Report 2008-1327

U.S. Department of the Interior
U.S. Geological Survey

¹U.S. Geological Survey, Menlo Park, Calif.

²COSMOS Virtual Data Center, University of California, Santa Barbara (now at: NEESit, NEES Cyberinfrastructure Center, University of California, San Diego, La Jolla, Calif.).

³California Geological Survey, Office of Strong Motion Studies, Sacramento, Calif.

⁴Kircher and Associates, Palo Alto, Calif.

⁵Wiss, Janney, Elstner Assoc., Inc., Emeryville, Calif.

U.S. Department of the Interior
DIRK KEMPTHORNE, Secretary

U.S. Geological Survey
Mark D. Myers, Director

U.S. Geological Survey, Reston, Virginia: 2008

For product and ordering information:
World Wide Web: <http://www.usgs.gov/pubprod>
Telephone: 1-888-ASK-USGS

For more information on the USGS—the Federal source for science about the Earth, its natural and living resources, natural hazards, and the environment:
World Wide Web: <http://www.usgs.gov>
Telephone: 1-888-ASK-USGS

Suggested citation:
Evans, J.R., Squibb, Melinda, Stephens, C.D., Savage, W.U., Haddadi, Hamid, Kircher, C.A., and Hachem, M.M., 2008, The Virtual Data Center Tagged-Format Tool; introduction and executive summary: U.S. Geological Survey Open-File Report 2008-1327, 171 p. [<http://pubs.usgs.gov/of/2008/1327/>].

Any use of trade, product, or firm names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

Although this report is in the public domain, permission must be secured from the individual copyright owners to reproduce any copyrighted material contained within this report.

The Virtual Data Center Tagged-Format Tool: Introduction and Executive Summary

By John R. Evans, Melinda Squibb, Christopher D. Stephens, W.U. Savage, Hamid Haddadi, Charles A. Kircher, and Mahmoud M. Hachem

Executive Summary

This Report introduces and summarizes the new Virtual Data Center (VDC) Tagged Format (VTF) Tool, which was developed by a diverse group of seismologists, earthquake engineers, and information technology professionals for internal use by the COSMOS VDC and other interested parties for the exchange, archiving, and analysis of earthquake strong-ground-motion data.

Background

For many decades, specialized formats have been defined and used to encode and store digitized acceleration time-series data from strong-motion sensors, as well as information derived from these data (for example, the “response spectra” widely used by structural engineers for design and retrofit of structures subject to seismic hazards). While many strong-motion formats currently in use derive in some way from the California Institute of Technology Blue Book format (circa 1970), there is little accepted standardization. Additionally, as compared to other seismic disciplines, the strong-motion and engineering communities require much more extensive supporting and derivative data to adequately qualify a record; thus, the use of more modern formats, such as SEED and SAC, is precluded.

In many ways current strong-motion formats reflect an early period of digital computation and storage when such resources were limited by today's standards. These early facilities also relied on the column-formatted “flat files” used by Fortran and lacked modern information-technology (IT) notions, such as “structured” storage, “object-oriented” and “platform-independent” programming, markup languages like xml, and relational databases.

The COSMOS v1.20 fixed-field format successfully amalgamates the various derivatives of the “Blue Book” fixed-field format and other compatible formats and could be used as a standard mechanism for exchanging data among various institutions and individuals worldwide; indeed, it is being used in the exchange of data between several institutions within California (the California Integrated Seismic Network), for example.

There is an opportunity and a need for mechanisms of strong-motion formatting, data exchange, and processing that exploit modern resources and techniques. There is also an opportunity to create a viable international exchange and format-conversion medium—strong-motion seismology and earthquake engineering depend upon international exchange of data and results. For example, the data provided to the world by Taiwan's Central Weather Bureau for the Chi-Chi earthquake mainshock more than doubled the worldwide strong-motion database and addressed issues that were previously

untenable. In particular, the global scope of VDC activities necessitated a medium of conversion between the numerous extant, historical, and likely future formats for strong-motion data and products. This VTF tool is intended to fill this need. By analogy to telephony switching centers, the VTF tool will be used as a crossbar switch, interconnected and compatible with all sources and users of strong-motion data. The VTF, and specifically its xml mirror, currently serve as a primary data-storage, data-exchange, and operational tool within the VDC IT structure.

The limitations inherent to fixed-field formats and legacy IT techniques motivated our indepth review of strong-motion formatting, data exchange, operations, and management, resulting in this proposed VTF along with its exactly equivalent xml representation and provision of key user tools for both the VDC and other interested parties.

Approach

The VTF tool is designed with the following key goals and constraints:

1. To be readily and unambiguously readable throughout by both humans and computers;
2. To be expressly international in its viability, including support of non-English scripts and the use of English—the current international standard for science and engineering—as its basis;
3. To be inherently both “backward compatible” with older versions of the VTF and prospectively extensible with minimal effort by the VDC and all potential users (that is, no tag will ever be dropped from subsequent versions, so that software that can read a later version of the VTF will also be able to read legacy VTF files);
4. To be inherently self-documenting and autonomous (that is, not requiring reference to *any* external information for the meaning of all parts of a record to be clear to future users even when *all* supporting information, such as this document, is lost; and
5. To be easily translated to and from both fixed-field-formats and formally structured formats, including xml, Excel™, relational databases, and other media now in use or likely to evolve.

To a lesser degree, the VTF tool also is designed to minimize redundancy and duplication within each file—that is, to express a given piece of information in exactly one location. Because of widely reported difficulty with parsing multiple-record files, we also limit VTF records to a single component of data per file. Finally, the notion of separate text, real, and integer header sections is supplanted by a universal form of data-type identifier, allowing either alphabetical or logical arrangements of the information for ease of use and for making evident the relationships between tags. The notions of data types remains, but only as guidance incorporated directly in the name of each variable (in the formal IT sense, *all* VTF data are text, including the text representations of numbers).

To accomplish these goals, we have chosen a simple tagged format exemplified by ***Descriptive.Tag_type = value units***; Within this tagged format we also implement a modern “structured” or “class” design that is easily understood by a reader but readily translated by computers to and from other structured forms. Structuring the tag names also makes relationships between them more evident. Thus, we have made every tag name, its value, and its units of measure explicitly descriptive (and have provided for extensive comments). This design is the essence of dual human- and computer-readability, as well as of extensibility and autonomy.

An example of one minor set of entries for a VTF file is:

```
DataSeries.Sa(1).Period_dbl = 0.3 s;           ||| Period
DataSeries.Sa(1).StructureDamping_dbl = 0.05;  ||| Damping, 0-1
DataSeries.Sa(1).Value_dbl = 32.0 cm/s/s;      ||| Sa value
```

This example defines the response spectral acceleration (32 cm/s^2) at a period of 0.3 s for 5 percent structure damping (0.05 fractional damping). The comments at the right or in the numerous *Comments* tags are anything the users believe will be helpful to themselves or others. The index (*I*) illustrates that arrays of tags—arrays of any length—can be created.

Three departures from Blue-Book-style formats are:

1. There is no longer either a fixed number or any concatenation of real, integer, or text variables (in order to allow greater efficiency and extensibility as well as more convenient or logical organization);
2. There are no coded entries, that is, no look-up tables requiring the survival of those tables for the entry to be meaningful. Instead, all table values are clearly understandable words, phrases, or values (\pm units) whose meaning will be the same in thirty years as they are today without reference to the tables from which they come; and
3. We have corrected the few typographical errors and omissions in the v1.20 fixed-field format while the opportunity for complete, detailed descriptions also is greatly expanded.

Where look-up tables are used, in addition to the listed self-evident values most such tables are extensible. That is, VTF users may define new entries if they do not find a suitable value already listed, so long as it is terse and clear to all users. Similarly, users may invoke *Private* tags when no extant tag suffices. While it is highly desirable to use standardized values and tags, and we have attempted to be comprehensive, these facilities accommodate items we have not anticipated and provide a means for suggesting additions to the VTF, which will be reviewed and acted upon.

Utility

Applications of the VTF include providing a generalized storage and translation medium for use by the VDC to simplify and expedite translations between all sources and users of strong-motion data internationally. The VTF tool is available for download and use by any interested party and is complemented by a series of portable software applications designed to make implementation straightforward.

Because we have kept the essence of the COSMOS v1.20 fixed-field format in the VTF tool, all of the Blue Book data dictionary is reproduced here, though sometimes rearranged or generalized. Thus, it is straightforward to translate unambiguously between VTF and the v1.20 fixed-field format and, by extension, all other formats.

The support tools currently available from the VDC are self-updating Java™ routines for access, display, and input/output; these subroutines can become the front and back ends of existing processing systems as desired. Initially, we are supporting Fortran, C, MatLab™, and Excel™, which are used widely by these communities. We also ask the strong-motion engineering and seismology community for consensus on other software worthy of direct support. By design, the translation routines are easily extended to other formats, thus are the basis of a universal translator at the VDC.

Appendix

Attached at the following page is the current governing document for the VTF. Although this document is detailed and in some ways difficult to read, it is the sole formal definition of the format and is kept up-to-date and distributed only by the VDC and its assigns. The most current version can be downloaded at <http://db.cosmos-eq.org/FormatConversion.html> if they postdate this Open-file Report.

The COSMOS Virtual Data Center Tagged-Format

Tool: Formal Definition [Version VTF.1.0]

(16 October 2008)

*John R. Evans¹, Melinda Squibb², Christopher D. Stephens¹, W.U. Savage¹,
Hamid Haddadi³, Charles A. Kircher⁴, and Mahmoud M. Hachem⁵*

¹*U.S. Geological Survey, Menlo Park, Calif.*

²*SOPAC, University of California, San Diego, La Jolla, Calif.*

³*California Geological Survey, Office of Strong Motion Studies, Sacramento, Calif.*

⁴*Kircher and Associates, Palo Alto, Calif.*

⁵*Wiss, Janney, Elstner Assoc. Inc., Emeryville, Calif.*

(There is a Table of Contents at the front and an Index of Tags at the end of this document. This is the sole, formal, governing document for this format, and as such is authoritative but difficult reading. The Executive Summary and other supporting items are supplied only for the user's convenience.)

Table of Contents

INTRODUCTORY MATERIAL.....	11
OVERVIEW.....	11
FORMAL DEFINITION OF THE COSMOS VIRTUAL DATA CENTER TAGGED FORMAT.....	13
Formal Syntax.....	14
Authority.....	15
Language.....	16
A Note on Fourier Spectra.....	16
Tag Arrays.....	17
Primary Classes.....	18
TABLE OF PRIMARY TAG CLASSES AND DEFINITIONS OF TERMS.....	19
TAG-VALUE EXAMPLE.....	21
A NOTE ON COMMENTS.....	22
ADDITIONAL MATTERS OF FORMAT.....	23
File Naming.....	23
REQUIRED AND LESSER TAGS.....	26
NOTATION.....	27
ORGANIZATION OF THIS DOCUMENT.....	27
ISO DATE-AND-TIME FORMAT FOR DATETIME TAGS.....	28
ON PRECISION: THIS IS IMPORTANT!.....	29
NULL VALUES.....	30
THE COSMOS VDC TAGGED FORMAT TABLES (VTF.1.0).....	31
TABLE 1. VARIABLE-TYPES SUFFIX LIST.....	31
TABLE 2. RECOGNIZED UNITS.....	32
TABLE 3A. CERTAIN GENERAL AND SPECIAL HEADER TAGS.....	36

TABLE 3B(A). TYPE OF DATASERIES	41
TABLE 3B(B). CODES FOR TYPES OF DATASERIES	42
TABLE 3C(A). RECORD TYPES	45
TABLE 3C(B). RECORD-TYPE AND INPUT-TYPE CODES	46
TABLE 3D. GEOGRAPHIC NAMING INFORMATION.....	47
ON SENSOR GEOLOCATION NAMING.....	50
SNCL Naming Details.....	52
TABLE 3E(A). ARRAY VARIABLES, INCLUDING NETWORK/OWNER/AGENCY CODES	53
TABLE 3E(B). COSMOS AND FDSN ARRAY/NETWORK CODES.....	56
GEOGRAPHIC-POSITION ISSUES	58
On the Accuracy of Geographic Positions.....	59
TABLE 3F. OTHER GEOGRAPHIC AND ARRAY TAGS—COORDINATES	59
TABLE 3G. GEODETIC ELEVATION DATUMS.....	64
TABLE 3H(A). SITE TYPE AND THE BUILT ENVIRONMENT	66
TABLE 3H(B). SITING CONDITIONS, REFERENCE, AND FREE FIELD	78
TABLE 3H(C). BUILT ENVIRONMENT VALUES (CSMIP).....	79
TABLE 3H(D). SITING CONDITIONS, BRIDGES	81
TABLE 3H(E). SITE TYPE AND BUILT ENVIRONMENT VALUES—HAZUS CODES	82
TABLE 3H(F). OCCUPANCY CLASSES—HAZUS CODES	84
TABLE 3I. EVENT (EARTHQUAKE) INFORMATION	85
TABLE 3J(A). DATA-ACQUISITION UNIT (DAU; RECORDER; DATA LOGGER) INFORMATION	96
TABLE 3J(B). DAU MODELS AND MANUFACTURERS	102

TABLE 3K(A). SENSOR INFORMATION.....	106
TABLE 3K(B). SENSOR MODELS (USE TABLE 3J(B) FOR MANUFACTURERS).....	114
TABLE 3K(C). SENSOR DIRECTION CODES FOR USE WITH SENSOR.ORIENTATION ONLY (AVOID) ...	116
TABLE 3L. DATA SERIES (INCLUDING ORIGINAL): TIME-SERIES OR SPECTRUM INFORMATION	118
TABLE 3M. RECORDER TIMING METHOD	131
TABLE 3N. PROCESSING INFORMATION.....	132
TABLE 3O(A). CODES FOR FILTER TYPES USED IN PROCESSING	143
TABLE 3O(B). CODES FOR FILTER TYPES USED IN PROCESSING.....	144
TABLE 3P. RELATIONAL TAGS INVOLVING TWO GEOGRAPHIC LOCATIONS.....	145
TABLE 3Q. PRIVATE TAGS (AVOID)	147
TABLE 3R. DATASERIES TAGS CLOSELY ASSOCIATED WITH TABLE 4	149
 Critical Notes on Formats.....	150
TABLE 4. THE TIME SERIES OR SPECTRUM (THE DATASERIES VALUES)	154
REFERENCES	157
INDEX OF TAGS.....	159

Introductory Material

Overview

The COSMOS v1.20 fixed-field format was an effort to unite in a single forum the various derivatives of the CalTech “Blue Book” formats being used to store and distribute strong-motion data. Issues of readability, generality, comprehensiveness, structure, and autonomy motivated revisiting the format issue in depth, resulting in this proposed COSMOS VDC Tagged Format (VTF) and its equivalent xml mirror as additional tools for the VDC (Virtual Data Center) and others who may find these tools useful. Efforts are underway to revise the fixed-field format, as well.

Primarily, the VTF tool is designed to be readily and unambiguously readable throughout by both humans and computers, to be expressly international in its viability, to be inherently both backward compatible and extensible with little effort (thus, easily maintained), to be inherently self documenting and autonomous (that is, not requiring reference to any external information for the meaning of all parts to be clear in the future even when all supporting information, including this document, are lost), and to be translated easily to and from structured formats, including xml, Excel™, databases, and other common media. To a lesser degree, this format tool also is designed to minimize redundancy and duplication in each file (to express a given piece of information in one location only, feasible because of its dual human and computer readability). The notion on text, real, and integer header sections is supplanted by a universal form of header information.

The tagged format’s applications include providing a generalized storage and translation medium for the COSMOS VDC, a medium that can absorb and translate data from all sources internationally. Thus, the format is intended to be a superset of all formats now used in strong-motion work and to anticipate likely future needs. The tagged format also is available for use by any interested party, along with a series of tools designed to make that use straightforward.

By “backward compatible” we mean that parsing (reading) software for VTF Version VTF.1.1 and beyond will be able to read earlier VTF versions without difficulty, and this functionality will persist as a natural part of the format definition.

To accomplish the goals enumerated above, we have chosen a simple “tagged” format (and within this tagged format a modern “structured” or “class” design that will be both user friendly and still readily translatable by computers to and from other formats).

We have made every tag name (and its value) explicitly descriptive and have added extensive comments in this governing document about each tag’s purpose and rules of use. Indeed, most of these tag names should look familiar to those who know the Blue Book formats and, where not so, we have commented about what equivalent variables our tags replace.

This design is the essence of dual human- and computer-readability. Everything expressed in VTF is made easy to read, understand, and bring into analysis systems.

Two departures from the Blue Book formats are, first, that there is no longer a fixed number of real or integer variables; this allows both greater efficiency and extensibility. Second, there are no longer any “coded” entries—no references to tables of entry definitions that must survive into posterity. Where there are tables of values that can be entered, all the values in those tables are clearly understandable words, phrases, or values whose meanings will remain the same and will not require reference to the tables from which they come.

In all instances we have tried to keep the essence of the v1.20 fixed-field format alive and well in the VTF tool. Although rearranged and recast into a more parseable form, all of the Blue Book data dictionary is reproduced here in some form. It is, therefore, straightforward to translate unambiguously between VTF and any of these Blue-Book formats. The VDC will be providing translation matrices and Java translation facilities, as well as subroutines for parsing and for writing VTF; these subroutines can become the front and back ends of existing *Processing* systems if desired. We will provide these front- and back-end routines in Fortran, C, MatLab™, and Excel™, and will ask the COSMOS community for consensus on other widely used software. The general translation routine will be in (automatically selfupdating) Java™ and driven by Excel™ tables describing the various formats. Thus, the translation routine is easily extended to additional formats and we anticipate supporting all widely used formats. Similarly, VTF can be used as the core of a “universal translator” connecting any of the formats described in those Excel™ tables. This translator and the Java™ interface to VTF will be key tools for implementing, refining, and extending COSMOS VDC functionality and user friendliness.

Finally, we have added items not available in v1.20 of the fixed-field format, such as various new types of instruments and manufacturers from Japan and Germany, Laplace, and z-plane descriptions of instrument and filter responses and to correct a number of trivial typographical errors.

The version of this document released by the COSMOS VDC is the sole formal definition of the COSMOS VDC Tagged Format Tool (VTF). Therefore, no change to the format or its use-rules from any other source is permitted. This document is long and detailed; a more easily understood executive summary and access tools for the format are available from the COSMOS VDC.

Formal Definition of the Cosmos Virtual Data Center Tagged Format

1. Exactly one time series or spectrum is allowed per VTF file, except in the case of response spectra, for which multiple damping values may be represented in a single file, all derived together from a single time series. Multiple channels are not to be concatenated.
2. In each VTF file:
 - a. Two mandatory header lines begin the file, giving the VDC Tagged Format version name and text-encoding scheme (for example, "US-ASCII"),
 - b. Next follow a variable number of other "header" lines,
 - c. Optionally, a URL (Web address) pointing to an alternate source for this/these **DataSeries** (time series or spectrum),
 - d. Optionally, a data-series checksum, and
 - e. A **DataSeries** listing matrix containing the primary data as described by the following three items, f. through h.:
 - f. A data-initiator line, "**DataSeries.DataSeriesValues_txt = {**",
 - g. A **DataSeries** listing matrix, each row (that is, line) containing, if given explicitly, an abscissa value (time, frequency, or period) followed by: (i) one time-series or amplitude- or power-spectral value (real or complex, hence one or two columns); or (ii) response spectral value(s) (real) for one or more damping values, such that columns 2 and beyond of the listing matrix form vectors of one response spectral period with varying damping;
 - h. Lastly, a data-terminator line, "**};**", closing the **DataSeries** listing matrix,
3. All tag-value pairs shall be of the following format:

Formal Syntax

```
<TagName>_<VariableType> = <value>[ <units>];[[<white space>]|| <comments>]
or
[<white space>]|| <comments>
```

where **TagName** is defined by

```
<TagName> = <ClassName>[ (m) ] [ .<SubClassName>[ (n) ] [ .<SubSubClassName>[ (o) ] ... ] ] <MemberName>
```

As is common in software documentation, “◇” delineates a description of what belongs there while “[]” delineates optional or occasional elements, possibly nested. Note that **<TagName>_<VariableType>** (as well as any text values taken from the tables below) **are case sensitive**. Tag values are usually in English; tag names are always in English.

Similarly, neither tag names nor text values may contain any of the following characters: tabs, “, ‘ (double and single quotes), and ` (grave accent). Such characters can be misunderstood by xml and databases.

Furthermore, while tag names shall not contain white space characters, there shall be white space on both sides of the equal sign (“ = ”) and between the **<value>** and any **<units>**. White space may be blanks or tabs. Finally, note the presence of the semicolon after either the **<units>** or a unitless **<value>**—that is, the semicolon terminates the “**tag = value [units]**” sequence and must be present.

White space also is allowed almost anywhere else, but not within tag names nor within numerical values (you can put white space inside text that you create yourself, but you must not inject it into text values taken from tables—use those exactly as they appear in the tables).

Numerical values may appear in integer, floating point, or scientific notation, but must be in base ten. (To avoid decimal-binary-decimal translations, thus preserving accuracy, numerical values are treated as text by most VTF utilities until such time that they must be operated upon as numbers, such as for plotting the *DataSeries*.)

In most cases tags can be given in any order, however, it is recommended that either an alphabetic order or the functionally organized order shown here generally be respected so that tags can be located easily. The *Index of Tags* near the end of this document is a comprehensive alphabetical list that may aid users.

(There are three exceptions to the flexible-order rule, one at the top and two at the bottom of the file. (1) The first tag in table 3A must appear first in the file, (2) the tags surrounding the *DataSeries* are in a specific order, and (3) the *DataSeries* listing matrix, when the abscissa is implicit, must be in increasing order of the abscissa's value. We encourage the same practice even when the abscissa is given explicitly as the first (or first two) columns of the *DataSeries* listing matrix.

As implied above, there are instances in which the abscissa of a time series or spectrum must be given explicitly as the first one (real) or two (complex) columns of the *DataSeries* listing matrix. However, there are many common situations in which it is sufficient to define the abscissa implicitly, as with regularly sampled time series and Discrete Fourier Transform (DFT) spectra. Many other spectral forms will require an explicit spectrum. The rule is that if an abscissa can be correctly defined by a starting point and an increment, it may be (but does not have to be) given explicitly. All other cases must provide an explicit abscissa. Instances of requisite explicit abscissas commonly include response spectra and power spectra computed from a DFT that is spaced evenly in frequency but where the power spectrum is provided as a function of period.

Authority

The only authoritative definition of the COSMOS VDC Tagged Format (VTF) is this document, as maintained by the COSMOS Virtual Data Center or its assign.

Nevertheless, the VTF has many “user-extensible lists” (tables of values) for which a user may assign any reasonable value, as long as it starts with the string “**User’s description:** ” (to help us make format-verification work well in the most general case). Such user-created values in extensible lists are likely to help us expand this authoritative VDC document. Similarly, use of *Private* tag sets, defined in table 3Q, also may inform our modifications of the VTF.

Private tag sets and user-created text values should be limited to necessity and kept reasonable and likely to be understandable to any user at any future time.

Language

The official language of all tags (*including Private* tags) and of most text values is English. However, we recognize that certain text values, such as the descriptive name of a geographic location (*GeoLocation.Name.Description*) and addresses (*GeoLocation.Name.Address*), sometimes must be in their local languages to be meaningful. Furthermore, translation of comments and other errata into English should not be allowed to become a barrier to the dissemination of data.

A Note on Fourier Spectra

For a DFT, if only the non-negative frequencies for a real time series are to be given in this VTF file, then the power in the negative-frequency bins generally must be added to their corresponding positive-frequency bins so that the positive-frequency bins represent the total power at each frequency (generating a true one-sided spectrum). For real time series, this correction commonly can be accomplished by scaling up the non-Nyquist positive-frequency bins by the square root of two, however, caution is appropriate.

(Many FFT implementations of the DFT yield their output in some order or another, such that it contains negative-frequency bins, one 0-Hz (“DC”) bin, positive-frequency bins, and one Nyquist bin. Typically then, half the power of non-Nyquist finite-frequency bins is to be found in the negative-frequency bins. Thus all non-Nyquist positive-frequency bins are likely to be too small by a factor of the square root of two. In contrast, most of these FFT routines put all the power of the 0-Hz and Nyquist bins each into a single bin, so that these two bins do not need correction.)

Since many FFT routines differ in their output formats and rules, it is essential that the user test the behavior of their FFT routine with real data to determine whether this correction for power lost to negative-frequency bins is required as well as to determine the particular behavior of the 0-Hz and Nyquist bins.

To test an FFT routine, take the FFT of one or more pure sine waves fitting exactly into the input window and using no weighting in that window (that is, a boxcar window). “Fitting exactly” means that the time-series point just after the end of the input window and, therefore, not present in the data provided to the FFT, must exactly equal the first point in the input window and be in the same phase. That is, that the window duration is one sample shorter than a full integer

number of cycles. This arrangement is called “DFT symmetry”—because the DFT assumes a precisely periodic input, repeating the first point just after the last and so forth, DFT symmetry yields an effectively infinite-duration pure sine wave which will put all its power into exactly one frequency bin or exactly two complementary-frequency bins, positive and negative—there will be no side lobes. The total power contained in either the one bin or the positive-negative pair of bins will equal the power computed in the time domain. If a pair of bins is required to match the total power in the time domain, then the correction to non-Nyquist positive-frequency bins is required when providing only the non-negative bins in the VTF file. Similarly, it is possible to verify whether all or half the power of the 0-Hz or Nyquist bins is in one bin.

Tag Arrays

The VTF allows ***Subscripted arrays*** of variables (that is, arrays of tags). This facility allows for things like enumerations of filter poles and zeros, several earthquakes appearing in a single record (for example, rapid-fire aftershocks or swarm earthquakes), and historical information about an instrument.

These “tag arrays” are implemented by the “***(m)***” part that is shown at the ends of class or subclass names as indicated in this document—in other words, ***Subscripts*** are not allowed for a tag or portion of a tag unless shown in this document. ***Subscripts*** must be positive integers (1, 2, 3, ...) with no missing values, so they are in the MatLab™ or Fortran style. ***Subscripts*** are not in the C style of non-negative ***Subscripting*** (0, 1, 2, ...).

For example if a user wants to declare several possible versions of an ***Event*** location those tags would be ***Event.Hypocenter(n)***. In contrast, locations for multiple ***Events*** appearing within a single record would be rendered as ***Event(m)***. However, it would not make any sense to write ***Event.Hypocenter.Agency(n)***, which would imply one ***Event*** location with several authors, so this ***Subscripting*** is not permitted.

In another example, a user can document multiple instances of names for a spot on the Earth or in a structure (a ***GeoLocation***)—a history of changing names or differing names used by different ***Agencies*** for the same instrument. A user might, for example, want to document some name multiplicity with ***GeoLocation.Name(1).ShortName***, ***GeoLocation.Name(2).ShortName***, and so forth, in cases where the ***Sensor*** never moved or moved only slightly in the past and is functionally at the same location. ***GeoLocation(n)*** makes no sense since in the above example as it would imply a ***Sensor*** is in several places at once; this usage is disallowed.

Classes and subclasses end with periods while class members end with an underscore (“_”). The “(m)” is just before that period or underscore character and just after the class or member name, as in “*Sa(2).*” or “*ResponseSpectrumDamping(3)*”.

To limit ambiguity, users are urged to keep these *Subscripts* time-, frequency- or period-sequential where those concepts are meaningful. Note the prevalence of explicit time stamps in this format, which also facilitate documenting historical associations.

Our *Subscripting* guidance is either “[*(n)*]”, meaning “optional *Subscript* here”, or “(*n*)” meaning “required *Subscript* here”. As mentioned, *Subscripts* may not appear anywhere else. If a tag allows *Subscripts* but none are given, it defaults to an implicit *Subscript* of “(*1*)”—note that only the last of repeated un*Subscripted* tags will be saved.

Primary Classes

Both the physical grouping of tags and their <[Sub]ClassName> and <MemberName> are attempts to associate variables in a logical manner. For example, variables having to do with the Data Acquisition Units (*DAUs*, which are recorders and data loggers) are named by the *DAU* class; tags associated with the geographic locations (“station names”, loosely speaking) are *GeoLocation.Name*; and tags associated with the source event are *Event*. The following table includes a complete list of the primary (“top”) set of Classes and their definitions.

Table of Primary Tag Classes and Definitions of Terms

ClassName	Meaning
<i>ThisFile</i>	A small set of special tags which apply to this VTF <u>file</u> , such as those naming the version of the format and the VDC's unique identifier of this record.
<i>Array</i>	<p>Any set of <i>Sensors</i> plus <i>DAUs</i> at one or more <i>GeoLocations</i> that is managed as or logically forms a single data-acquisition system—for example, a structural <i>Array</i>, a geotechnical <i>Array</i>, or a single free-field instrument at a single <i>GeoLocation</i>. (An <i>Array</i> with a single-<i>GeoLocation</i> is the usual case in the free field, but it is still an <i>Array</i> in the same sense that a scalar is a one-element vector).</p> <p>Note that arrangements many of us think of as and call "an array" (such as the NESMP "array"), are <u>not <i>Arrays</i> as narrowly defined here</u>. In the VTF, such agglomerations are more properly called an <i>Agency</i>.†</p>
<i>GeoLocation</i>	<p>The exact geographic location in three dimensions of the <u>point of measurement</u> (strictly speaking, of the <i>Sensor</i>'s proof mass).</p> <p><u>Note that this is a very narrow definition</u> and defines the location of a single <i>Sensor</i> or package of several axes. Even some classical simple "stations" with <i>Sensors</i> spaced a few meters apart might be considered by some users to occupy several distinct <i>GeoLocations</i>, for example, in cases of precise GPS location and large ground-motion spatial variance.</p> <p>We <u>strongly discourage</u> the practice of giving the same set of coordinates for all members of an <i>Array</i>, such as the <i>Array</i> monitoring a dam site. This practice obviates many applications of the data.†</p>
<i>GeoLocationRupture</i>	Tags relating <i>GeoLocations</i> to fault-rupture surfaces.
<i>Event</i>	The <u>signal of interest</u> (for example, the natural or artificial source of ground motion, a calibration signal, and so forth).
<i>EventGeoLocation</i>	Tags relating <i>Events</i> to the <i>GeoLocations</i> of <i>Sensors</i> .

DAU	Data Acquisition Unit—the recorder or data logger. Typically a preamplifier, analog-to-digital converter, and some recording and/or telemetry medium, when taken in combination. Formerly, often a film recorder.
Sensor	The <u>transducer</u> that converts the Earth or structural motion to an electronic (or sometimes optical) signal. Generally, the accelerometer.
DataSeries	The single, delimited, <u>time-, frequency-, or period-series</u> or group of response spectra, as referenced by or included in this VTF file—that is, the primary data of this file. Also, any other values closely associated with, or generated from, these primary data (the start time of the time series, its RMS or maximum value, its Sa or Housner Intensity, the data format and number of points, and so on).
RawSeries	Values closely associated with, or generated from, the time- or frequency-series from which this processed DataSeries originated—the immediate predecessor to this VTF file.
Processing	Parameters describing the record- Processing steps applied to the RawSeries or original time series to obtain the DataSeries in this file.
Private	Private variables for use <u>only when</u> there is not yet any appropriate VTF tag. Search the Index of Tags carefully before defining a Private tag.

†The reader may notice that we rarely use the word “station”, instead nearly always using either the word **Array** or the word **GeoLocation** as they are narrowly defined above. We do so because the word “station” means many different things to different practitioners, in some cases signifying an entire **Array**. Similarly, we limit our use of the word “**Site**” to mean the site conditions at a particular **GeoLocation**; “site” has other special meanings to large portions of the COSMOS community, as in “work site”, which is broader than our definition (notwithstanding that many **GeoLocation.Site** data are generalized for an entire work site, often for reasons of cost, and not as specific to a particular **GeoLocation** as one might desire.)

In tags identifying entities responsible for various parts of this VTF file (such as the owner/operator of the instrument in question) we use the word “**Agency**”. By selecting this word, we do not mean to limit in any way the type of entity (organization or individual) that may be named in such variables. The term **Agency** is nothing more than a reasonably general <MemberName> with a meaning tolerably close to its intended use. We expect that many an “**Agency**” will be an individual person (such as amateurs of the Public Seismic Network), a secondary-school classroom, a Government agency, a consortium such as IRIS, or a college or university (this name is analogous to a GVDC “Business Associate”).

In all cases, we intend that preference be given to identifying **Agencies** by FDSN network names, some of which are shown in the second column of table 3E; additional names that may be used are given in the first column of that table. If no appropriate **Agency** name is available from these lists, the user may insert some terse, well known, identifiable abbreviation for an organization's name or insert the individual’s name. That is, the **Agency** list is “user extensible”—it is unlikely that the we have been comprehensive and we do not wish to exclude any **Agency** from the VTF.

Tag-Value Example

To illustrate this format, the following lines represent typical spectral acceleration values at the two periods, 0.3 and 1.0 s, for 5 percent damping:

DataSeries.Sa(1).Period_dbl = 0.3 s;		Period
DataSeries.Sa(1).StructureDamping_dbl = 0.05;		Damping fraction
DataSeries.Sa(1).Value_dbl = 32.0 cm/s/s;		Sa value
DataSeries.Sa(2).Period_dbl = 1.0 s;		Period
DataSeries.Sa(2).StructureDamping_dbl = 0.05;		Damping fraction
DataSeries.Sa(2).Value_dbl = 50.0 cm/s/s;		Sa value

A Note on Comments

There are two kinds of comments in the VTF: (1) comments associated with a particular group of tags (any **Comments**, **Description**, **Annotations**, or **Citation** tags) and (2) comments that can be put almost anywhere, but which are given limited treatment by the parser (for example, “|| This value needs updating.”).

For all important comments, please give preference to the group of tags ending with **Comments**, particularly those that you wish to associate rigorously with a particular subject matter. Although “||” comments are brought forward by the parser, they are associated only loosely with their original subject (by position in the VTF file). For example, in the xml mirror of the VTF “||” comments are added to the item at which they are given as a “**Note**” xml type. That is, the parser simply attaches all “||” comments to the nearest tag above the “||” comment). Thus, “||” comments are primarily for human consumption and not computer manipulation and should be avoided.

Nevertheless, “||” comments are used extensively in this governing document and other supporting materials to elucidate details of usage—these comments generally should not be included in any working file.

In practice, “||” comments might be used to provide *temporary* notes by the user to the user (for example, “|| still need to fix this”) or to make minor comments that do not need to be associated rigorously with any particular tag. However, “||” comments should not be used to make comments that are important in any significant way to the data.

Note that this is a change from the similar looking comments of the Blue Book formats. Important comments should be put into the variable **ThisFile.Annotations.TextValue** if not elsewhere associated with specific portions of the header in one of the other **Comments** (or **Description**, **Annotations**, or **Citation** tags) variables.

(The xml class **Note** is not visible to the user. It is used by parsers to store “||” comments in proximal association with the tags to which they are appended. All such comments are attached to the nearest previously encountered tag.)

Additional Matters of Format

Blank lines and lines consisting entirely of comments may appear anywhere after the first line excepting within the *DataSeries.DataSeriesValues_txt* = {...}; sequence of lines. Studies have shown that simply inserting blank lines in appropriate locations, such as between major header sections and between logically associated blocks of information, greatly improves the human readability of files. We commend such uses of white space to the reader and make extensive use of it in this governing document.

There is no firm limit on line length since the COSMOS community is no longer bound by the rigors of punched cards. Nevertheless, for the sake of keeping files easy to read on modern workstation screens, *users are urged to keep line lengths below 100 characters where feasible*. This length easily fits onto screens possessed of moderate resolution at reasonable character size without imposing unreasonable limits on the sizes of tags or their values. (This document uses about a 100-character line length, for example.)

The primary exception to this voluntary constraint is for *Comments, Description, Annotations*, and *Citation* tags (the preferred sorts of comments), which users may wish to make long in many cases. A word-based wrap-around display rule in your editor of choice should be sufficient to ensure readability of such text, as long as users are reasonably careful to write with adequate white space inside the comment text.

In particular, *please replace line break characters* with the four-character sequence “*\n*” to ease the reader’s job. (The “*\n*” is the C-language symbol for “newline”.)

In any case, *lines shall not be blank padded* on the right. (All lines will be read in as strings by the parsers, so blank padding simply wastes file space.)

File Naming

We suggest, but do not require, a format such as the following for naming VTF files:

```
<Date and time>[_<Network Code>]_<GeoLocation Name, possibly full SNCL>[_<Processing instance, stage, volume, etc.>] [_Ch<Channel>[In<Index>]] [_<Etc.>]_<[A|V|D|...]>.COSM
```

The COSMOS VDC will use such filenames in data it translates into VTF for a user.

For example, “20021103_221245_XX_PS09_Op1_Vo2_Ch2_V.COSM” is a record from the Denali fault earthquake of 03 November 2002 (“20021103_221245”) for Alyeska (“_XX” == “other network”) station [*GeoLocation*] “PS09”, evaluated with “*Processing* option/instance” one (“_Op1”), fully processed (“Vo2” == Volume 2), for Channel 2 (“_Ch2”), and is a velocity trace (“_V”). The “COSM” (or “cosm”) means “a COSMos vdc tagged format”.

The *<Date and time>* stamp should be in largest-to-smallest order and generally should be to the nearest second; we suggest the format YYYYMMDD_hhmmss, as in “20051117_173802”, which is a variant of the ISO time format we use here.

Note that this *<Date and time>* should be the best available time, preferably the time of the first sample. If that instant is not known, use the trigger time of that record. If even that is not known, use the origin time of the *Event*. (This ordered list of options is for the sake of consistency between records and to take account of legacy data, which often have very poorly known timing.)

The *<Network Code>* is from either *Array.Agency* or the “NN” field in *GeoLocation.Name.SNCL* or *GeoLocation.Name.ExpandedSNCL*.

The *<GeoLocation Name, possibly full SNCL>* is whichever of *GeoLocation.Name.SNCL*, *GeoLocation.Name.ExpandedSNCL*, or *GeoLocation.Name.ShortName* is used to identify the *GeoLocation* of this *Sensor*, with a SNCL name preferred if several are given. Obviously, use the *GeoLocation.Name* that is in force at the time of the *Event* and as used by the *Agency* providing this record.

The portion “[_Ch<Channel>[In<Index>]]” comes from the tags *Sensor.ArrayChannel* or *Sensor.DAUchannel* (giving preference to the former) and from *GeoLocation.Name.Index*.

In any case, the combination “[<Network Code>]_< GeoLocation Name> ... [_Ch<Channel>[In<Index>]]” should be as detailed as is required to identify uniquely the signals from a specific proof mass (both within that particular *Array* and among all your international colleagues). That is why a full SNCL name is preferred in place of the entire sequence of strings—the FDSN designed SNCL network names to be unique worldwide.

The portion “_<[A|V|D|...]>” should always be present and is a string of from one to three letters indicating the type of trace, “_A” for acceleration, “_V” for velocity, “_D” for displacement, and so forth as shown in column two of table 3B(b).

The optional “[_<Processing instance, stage, volume, etc.>]” identifies the stage of Processing, for example “_Vo2” for the equivalent of a Blue Book “Volume 2” record. This portion of the VTF filename should correspond to the meaning of one or more of the following five tags from table 3N as shown here:

Tag and value	Filename fragment
Processing.BlueBookVolume_int = [0-3];	“_Vo[0 1 2 3]”
Processing.Stage_txt = [“Raw” “Preliminary” “Final”];	“_St[R P F]”
Processing.HumanReview_txt = [“None” “Reviewed” “Updated”];	“_Hn[N R M]”
Processing.Instance_int = <non-negative value>;	“_Op<Value>”
Processing.SpecialProcessing_txt = “<Value>;”;	“_SP<Value>”

These substrings should be concatenated as needed, as in the example above of “_Op1_Vo2”.

Finally, some users may desire additional differentiating codes, the “[_<Etc.>]”.

When filenames begin in this manner with a <Date and time>, followed by a string that uniquely identifies the *DataSeries*, and end with a string that uniquely identifies the data-*Processing* stage, the resulting files generally will sort easily into *Events* with the help of typical system file-name-sorting and file-listing algorithms. It also will be clear to colleagues what the name means. Therefore, the portions of the filename after the <Date and time> should as specific as is required for the particular *Array* and *Agency*. Ideally, a filename should include a full SNCL name and strings identifying the type of *DataSeries* and data-*Processing* leading to the file. As long as an FDSN network code (*Agency*) appears in the name, there should be no naming conflicts with other *Arrays*.

Another example may make this convention clearer for the case of a SNCL name:

```
20041115_093415_CH4AW.NC.HNZ.01_V2_A.COSM
```

identifies a “Volume 2” vertical acceleration (“HNZ”) record from a temporary USGS instrument co-located with the CSMIP instrument “Cholame 4AW” near Parkfield. The corresponding CSMIP record might be named

```
20041115_093414_36412.CE.HNZ.01_V2_A.COSM
```

in this case using the CSMIP five-digit name for the same *GeoLocation*. There can never be confusion of a differing *DataSeries* because of the different FDSN network name “CE”.

Required and Lesser Tags

*VTF tags come in four flavors: **REQUIRED**, **CONDITIONAL**ly required, **Recommended**, and **Optional**.* On rare occasions, you also will encounter *Avoid*, which include *Optional* tags that are not recommended for use but are kept for backward compatibility and similar reasons.

All **REQUIRED** tags *must* be present in every VTF file, even when there is no value corresponding to the tag. **REQUIRED** tags with no values must be specified as NULL (*without quotes* for all variable types, including for text strings)—see discussion and table below. Of course, *Processing* software may not be able to function if required tags have NULL values (which are translated into a NaN (IEEE “not a number” value) or into an empty string, depending on tag type), so such values partly serve as reminders to the author that they still have values that probably should be filled in before the file is shipped. Nevertheless, if a required tag is simply unavailable, ship the file and let the users make due, “some data” generally being preferable to “no data”.

Conditionally required (**CONDITIONAL**) tags are required in certain contexts that are explained in the comments associated with each **CONDITIONAL** tag. For example, when poles and zeros are used, all associated tags are then **REQUIRED**, including that the number of poles and zeros become required and each of the poles and zeros themselves are also required.

