The XBRL Global Ledger Data Definition File (DDF)

Why XBRL GL can be useful even if you never create an XBRL GL instance document

# XBRL GL – The Standards Lens for Describing ERP Data

XBRL GL[[1]](#footnote-1) (XBRL’s Global Ledger Taxonomy Framework) is a way to holistically, generically and globally model the data found in typical ERP (Enterprise Resource Planning) systems in a standardized fashion. It has been designed as a bridge between transactional electronic standards and electronic reporting standards – in particular, those defined by XML Schema and XBRL taxonomies. Most ERP systems have functionality to create fixed length ASCII or delimited text files; however these text exports do not have standards for field name or other definitional attributed. This document describes a method to describe standard text exports from ERP systems for conversion into XBRL GL or for direct processing.

# XBRL GL is a standardized way to model and describe ERP data

XBRL GL is one definition of a cross-platform, application-independent, standardized import/export format for the data found in a back-end accounting system. It is well documented; there are training materials available; there is an ongoing working group expanding, evolving and improving it.

Some of the benefits of an XML-based standard for representing ERP data compared to typical text formats include

* Well-formed checking
* XML Schema validation, including data types and restrictions
* XML encryption and signature
* Application of additional business rules through ISO Schematron and XBRL Formula
* and XBRL labels

Almost every ERP system has a facility for exporting data from its data files - and definitely from its report writer - in text formats – fixed-length ASCII or delimited (typically comma-delimited in the US and semi-colon delimited in Europe and other places where the comma is the thousands separator). However, each ERP system uses its own expression for those text files, often modelled after the physical arrangement of the data files in their native format and the field names of the file, meaning there is often an inconsistency of similar data fields even within a single ERP module or instance. Being able to work with the files can be a challenge. If only there was an “esperanto” for text files, a standard to which the fields could be compared.

That is the goal of the XBRL GL DDF. XBRL GL can provide a logical way to look at accounting system data, and can act as a standardized interface for software to offer its underlying data up to report writers, web services, or other interfaces.

Imagine an auditor being able to walk up to any accounting system, fire up popular report writers, FRx or other integration and interrogation tools, and being able to query the data as if every system was the same – which it virtually could be.

Certainly, some enthusiasts might hope that every system can someday export XBRL GL natively. However, there are a number of reasons that some integrators believe that XML files (like XBRL GL) as export formats or outputs of queries are not practical (in particular, XML files can be “verbose”); but what if the user wanted the *benefit* of XBRL GL both for the output … AND the input.

This paper documents the concept of an XBRL GL Data Definition File – one that can describe how a potentially more “compact” text file – with the associated loss of validation, internal human readable tags, “object orientation”, and other benefits – can be accompanied by an XBRL GL DDF, so a consuming application can better understand the contents of the file.

# Practical requirement – minimize data transfer file sizes

In our discussions with tax auditors, some of the most experienced external users of other organization’s general ledger and accounting system data, we have found that they face a serious and practical problem; the extracts they are receiving from corporate systems and transferring to their machines can be gigabytes large already in CSV or fixed length formats. The concern, and one proven out by their tests, is that introducing markup of any kind increases the files sizes significantly (although once compressed with Zip or a similar archiver, the relative differences in size are much smaller). Therefore, XML-based representations of that data are currently considered impractical for audits of large corporations. The choice of XBRL GL or a proprietary format to represent the data does not change this perception.

While we can suggest that XML data compresses very nicely, it does need to be uncompressed for efficient processing. Although hard drive and USB memory device capacities are increasing, handling these large files is still a problem.

The question has been asked: how can we use XBRL GL to facilitate understanding, transferring and consuming the accounting system data while keeping the actual data in a CSV or fixed length ASCII format? Likewise is there some way to utilize a compressed format that can be accessed directly?

## Proposed Solution 1 – standardize the first line of CSV field labels alone

CSV files can have labels as the first line, or record. Applications such as Microsoft Excel and Access, when encountering a CSV file, will ask if the first line contains headers. Applications may be able to recognize the fields as XBRL GL fields and place the data into the XBRL GL structure. The field can be the XBRL GL element name alone; as the element names cross XBRL GL modules/taxonomies, the field may need to be the namespace/element name, or some other unique identifier.

## Proposed Solution 2 – Augment the more compact file with XBRL GL DDF

A representative of a company that caters to the data interchange community offered the idea that a relatively small file (the data definition file) could be provided along with a text export file – one that would contain instructions on how a consuming application could parse the transfer file, with instructions describing the files content using XBRL GL as a reference. That was in 2003-2004.

### Considerations

One can represent data with a text file in an unlimited number of ways.

The simplest, but most space costly and redundant way, is to treat every walk down a hierarchical tree as a complete new record. The most complex would use a code to signify how the rest of the line signified the data at that level of a hierarchy. In between are combinations of breaking out headers from line items to reduce the amount of redundant information expressed.

Some manipulation of the export file may be valuable or necessary; just as XML canonicalization operates on XML through sorts and other manipulations to ease comparisons, pre-processing of text files to facilitate groupings may be in order.

## Proposed Solution 3 –Binary XML

The World Wide Web Consortium is investigating the ongoing demand for small file sizes as people want to “get in on the XML game without – well, really getting in on the XML game.”[[2]](#footnote-2) Options include efforts like ASN.1-based representations of XML, BXML and a call for a gzip-raw-xml. Efficient XML Interchange 1.0[[3]](#footnote-3) is the W3C’s approach to “very compact representation for the Extensible Markup Language (XML) Information Set that is intended to simultaneously optimize performance and the utilization of computational resources”; representatives of the XBRL GL WG are continuing to evaluate its relevance to the efforts.

## Proposed Solution 4 – Shortcut table

In any XML file, there is a lot of repetitive information. XBRL GL has a limited number of tags, additional redundant content because of the contexts (2.0a: numericContext=””, nonNumericContext=””; 2.1 contextRef=””).

If applications understood a shortcut table (when you see #@@#01, replace it with ‘<sourceJournalDescription nonNumericContext=”’ and when you see #@@#01. replace it with ‘</sourceJournalDescription>’, such a table could be provided and the text replaced, reducing the size of the additional markup provided.

# XBRL GL DDF

XBRL GL DDF is defined by an XML Schema file. The file provides the guidance how to parse text files.

In the *documentInfo* section, the attributes of the file as a whole are expressed

Note: **bolded** items are mandatory

|  |  |
| --- | --- |
| **fileType** | Enumerated as *delimited* or *fixed* |
| linelength | Defines an expected length for fixed length files |
| delimiter | Captures the delimiter for delimited files, typically “,” or “;” |
| **firstLineHeader** | Boolean (true/false) whether the first record of the export file has headers |
| textEncoding | Guidance on the method of text encoding |
| textEnclosure | For delimited files, the character used to enclose text, typically single-quote (‘) or double-quote (“) |

*entryMaps* holds each *entryMap*, that associates information in the export file with XBRL GL concepts.

Note: **bolded** items are mandatory

|  |  |
| --- | --- |
| **xbrlElement** | The XBRL GL concept to which the export information will be assigned |
| filename | The filename of the relevant export information, necessary when multiple export files (such as a header and line item) are combined into one. |
| dateFormat | A transformation of dates, based on the Inline XBRL registry |
| fieldName | The name of the field, such as from the first/header row |
| additionalConstraint | Additional guidance to help in grouping |
| cvsInfo or  | For CSV files, the order for the item (e.g., the third item) |
| fixedInfo | For fixed length files, the starting position (startingPos) and length (len) of the text |

enumerationMapping represents how content in the export file maps to XBRL GL’s many enumerations

|  |  |
| --- | --- |
| localValue | The value in the export file |
| xbrlElement | The relevant XBRL GL element |
| xbrlGLEnumeration | The associated XBRL GL enumerated value |

The schema and examples of files (delimited text and the associated DDF) will accompany this document.

1. http://www.xbrl.org/GLTaxonomy [↑](#footnote-ref-1)
2. http://www.xml.com/lpt/a/2003/08/13/deviant.html [↑](#footnote-ref-2)
3. http://www.w3.org/XML/EXI/ [↑](#footnote-ref-3)