Data Documentation Initiative

DDI Working Paper Series -- Best Practices, No. 10

Subject

Management of DDI 3.0 Unique Identifiers (2009-02-15)

Document identifier:

http://dx.doi.org/10.3886/DDIBestPractices10

Authors:

Nikos Askitas, Arofan Gregory, Maarten Hoogerwerf

Editors:

Arofan Gregory and Maarten Hoogerwerf

Target Audience:

Software developers, data archivists, database administrators

Abstract:

This best practice specifies the technical requirements and procedures for ensuring well-managed identifiers for both internal use and external references. Management of identifiers is divided into two situations: during creation of new DDI 3.0 objects and during migration of older DDI versions into DDI 3.0 versions. These best practices will be described from both procedural and technical perspectives.

Status:

This document is updated periodically on no particular schedule. Send comments to editor: ddi-bp-editors@icpsr.umich.edu

23 **TABLE OF CONTENTS**

Introduction

While DDI 3.0 provides enhanced functionality in managing identifiers, one must proceed carefully to ensure that identifiers are constructed properly and managed consistently. This best practice lays out some steps to follow when building new identifiers and migrating DDI documents across versions.

1.1 Problem statement

All DDI 3.0 objects need to be uniquely identified to enable them to be referred to by other DDI 3.0 instances. Since in DDI 3.0 the scope of all objects has increased from just a DDI instance to the whole world, additional measures are needed to guarantee uniqueness within this broader scope.

1.2 Terminology

The key words *must, must not, required, shall, shall not, should, should not, recommended, may,* and *optional* in this document are to be interpreted as described in **[RFC2119].**
Additional DDI standard terminology and definitions are found in
http://www.ddialliance.org/bp/definitions

2 Best Practice Solution

2.1 Definitions

Uniqueness: A Globally Unique Identifier is a special type of identifier used in applications to provide a reference number that is unique in any context.

Migration: Migration in the DDI context refers to moving from a DTD to XML Schema in terms of document structure; and from DDI 2 to DDI 3 as well as from DDI 3 back to DDI 2 in terms of porting content.

URN: A URN (Uniform Resource Name) is an Internet resource with a name that, unlike a URL, has persistent significance - that is, the owner of the URN can expect that someone else (or a program) will always be able to find the resource.

2.2 Best Practice behavior

The assignment of DDI 3.0 identifiers happens in two situations: (1) when creating new DDI 3.0 objects; and (2) when migrating older DDI objects.

Creating New DDI 3 Objects

DDI identifiers should be used primarily by dedicated applications. End users should not have to deal with these.

DDI-aware applications should generate and interpret DDI URNs. DDI objects can be identified in two ways: explicitly by the use of a URN and implicitly by interpreting the identifying (inherited) element attributes: agency, version, and ID. The latter is automatically possible; the former requires a URN to be specified for each element. Adding such URNs has one advantage: clear identification of all elements. A drawback is that it creates a lot of redundant information. We recommend not adding them to DDI, but instead enabling software to be responsible for on-the-fly interpretation. See the Overview, Part 1, of the DDI specification documents for more information on DDI URNs.

The uniqueness of DDI URNs is assured via a hierarchical system. DDI delegates the responsibility for uniqueness to maintaining organizations. To this end, the maintaining organization is to register a DDI agency name at the DDI Alliance. Such names are allowed to contain semantic value such as the name or acronym of the maintaining organization. Such names should be chosen with care: they may be questioned after organizational changes.

Some organizations maintain many collections of DDI objects. If such collections are not tightly bound to the organization and are likely to be transferred to other organizations, then a separate agency identifier should be considered. This may result in multiple agency identifiers per organization. http://tools.ddialliance.org/?lvl1=community&lvl2=agencyid

This leads to the question of whether individual researchers can also apply for an agency identifier.

When multiple organizations participate in a project, they should consider requesting a special agency identifier because the project is likely to be maintained by different organizations.

Once uniqueness outside the agency has been dealt with, the agency should make sure that the uniqueness of the DDI 3.0 identifiers within the agency is guaranteed. Three suggestions:

1. A data manager specifies a prefix for which he or she guarantees uniqueness
    - Pro: User-friendly identifiers
    - Con: Responsibility lies with user
2. A registry is maintained that ensures uniqueness
    - Pro: User-friendly identifiers
    - Con: Must maintain a registry
3. An algorithm is used to guarantee uniqueness (e.g., uuid)
    - Pro: Automatic
    - Con: Long user-unfriendly identifiers and increases in file size

One technique that can be used in combination with the above is separating, for example, variables and variable schemes. If the variable scheme is unique, one only has to guarantee the uniqueness of variables within this scheme (by simply enumerating them). Another simplification would be to define mnemonics for element types, e.g., VS_ variable schemes, CS_ for concept schemes, etc.

Once a DDI instance is created, its uniqueness needs to be checked by means of a general purpose validation process. Other validations are to ensure that all required elements (agency, ID, and version) are specified and that no circular references exist.

The DDI should be validated in these ways:

- Identifiers within a scheme must be unique
- Schemes must be unique
- URNs must be unique
- URNs must be well-formed

See Part 1, Overview, of the DDI specification for more information.

Migration from DDI 2.0 to DDI 3.0

A special case is the migration of older DDI metadata to DDI 3.0.

A pragmatic approach is to do this in four separate phases:

1. Convert DDI 2.0 DTD to DDI 2.0 Schema if you have DTD-based documentation.
2. Transcribe all existing DDI into the DDI 3.0 format. Transcription means application of the table for conversion available at: http://www.ddialliance.org/sites/default/files/CorrespondenceMapping-spreadsheet.pdf This step is reversible.
3. Fill in the mandatory elements of DDI 3.0 such as variable schemes. If there are mandatory elements in DDI 3.0 that are not in the conversion table, those elements are guaranteed to be lost in going back from 3.0 to 2.0.
4. Use the advanced DDI 3.0 features.
    - E.g., identify any identical objects and assign correct DDI3 IDs to enable re-use, to add concepts and concept-schemes, etc. In applying advanced features of DDI 3.0, you disambiguate duplicate variable identifiers by using variable schemes which should be unique. You have to encode these schemes. This can be done by prefixing the variable with a scheme and a separator. The separator is required to enable distinguishing between the scheme and the variable when reversing this process again back to 3.0 (this applies to concepts, universes, codes, etc.).

This would result in:

2.0 v1 --- DTD 2 Schema → 2.0 v2 --- transcribe → 3.0 v1 --- fill mandatory elements –> 3.0 v2 --- advanced features → 3.0 v3

Migration from 3.0 to 2.0

A lot of existing tools can only work with DDI 2.0. How can you migrate 3.0 to 2.0?

- Strip off everything that is in 3.0 that is not convertible into DDI 2.0 by means of the table in Appendix 4 of Part I Overview. Obviously this cannot be undone when migrating back. There will be information loss in this case.

- You have to encode the DDI 3.0 schemes into the existing variable. This can be done by prefixing the original variable with this scheme and a separator. The separator is required to enable distinguishing between the scheme and the variable when reversing this process again back to 3.0. This prevents information loss (this applies to concepts, universes, codes, etc.).

- There is a special case in DDI 3.0 when there are no variables but there are elements that exist below the variable level in DDI 2.0 (e.g., question texts). A solution to this problem is to generate dummy variables. Be aware that this does introduce artificial/redundant variable information. A note can be made under the variable to document the origin of this dummy variable. If you want to reverse this migration, these dummy variables will appear in the DDI 3.0 instance.

Converting DDI 3.0 to DDI 2.0 can be done but loses the special DDI 3.0 functionality. Therefore this migration should only be done for downwards compatibility with, e.g., tools. Changes and maintenance should be done within the DDI 3.0 version if possible. Migration back to DDI 3.0 is not recommended. In this case we advise to start with the originating DDI 3.0 version.

2.3 Example

See the DDI specification, Part 1, Overview, available at:

http://sourceforge.net/projects/ddi-alliance/files/Data%20Documentation%20Initiative/DDI%203.0%20%282008-04-28%29/DDI_3_0_2008-04-28_Documentation_XMLSchema.zip/download

3 References

See the DDI specification, Part 1, Overview, available at:
http://sourceforge.net/projects/ddi-alliance/files/Data%20Documentation%20Initiative/DDI%203.0%20%282008-04-28%29/DDI_3_0_2008-04-28_Documentation_XMLSchema.zip/download

3.1 Normative

[RFC2119] S. Bradner, Key words for use in RFCs to Indicate Requirement Levels, http://www.ietf.org/rfc/rfc2119.txt, IETF RFC 2119, March 1997.

OASIS, Best Practice, http://www.oasis-open.org/committees/uddi-spec/doc/bp/uddi-spec-tc-bp-template.doc, 2003

Appendix A. Acknowledgments

The following individuals were members of the DDI Expert Workshop held 10-14 November 2008 at Schloss Dagstuhl, Leibniz Center for Informatics, in Wadern, Germany.

Nikos Askitas, Institute for the Study of Labor (IZA)

Karl Dinkelmann, University of Michigan

Michelle Edwards, University of Guelph

Janet Eisenhauer, University of Wisconsin

Jane Fry, Carleton University

Peter Granda, Inter-university Consortium for Political and Social Research (ICPSR)

Arofan Gregory, Open Data Foundation

Rob Grim, Tilburg University

Pascal Heus, Open Data Foundation

Maarten Hoogerwerf, Data Archiving and Networked Services (DANS)

Chuck Humphrey, University of Alberta

Jeremy Iverson, Algenta Technology

Jannik Vestergaard Jensen, Danish Data Archive (DDA)

Kirstine Kolsrud, Norwegian Social Science Data Services (NSD)

Stefan Kramer, Yale University

Jenny Linnerud, Statistics Norway

Hans Jørgen Marker, Danish Data Archive (DDA)

Ken Miller, United Kingdom Data Archive (UKDA)

Meinhard Moschner, GESIS - Leibniz Institute for the Social Sciences

Ron Nakao, Stanford University

200 Sigbjørn Revheim, Norwegian Social Science Data Services (NSD)

201 Wendy Thomas, University of Minnesota

202 Mary Vardigan, Inter-university Consortium for Political and Social Research (ICPSR)

203 Joachim Wackerow, GESIS - Leibniz Institute for the Social Sciences

204 Wolfgang Zenk-Möltgen, GESIS - Leibniz Institute for the Social Sciences

205

206 # Appendix B. Revision History

207

Rev	Date	By Whom	What
0.9	2009-02-15	Stefan Kramer	Formatting changes (standard across all BPs).

208

209

Appendix C. Legal Notices