

## Proposal for Minor Revision to/Improvement of the DDI Specification

**Working Name of Proposal:** Reinstating Nested Categories

**Working Group/Initiator:** ICPSR

**Alliance Member Sponsors (2 minimum):** ICPSR, Nesstar, Yale University - Social Science Libraries and Information Services (SSLIS) and Social Science Research Services (SSRS/ITS)

**Architect (name and email):** Sanda Ionescu (sandai@icpsr.umich.edu)

**Date of Submission:** 10/07/2004

**Type of Change Proposed (see definitions below):**

*(Choose one type.)*

Change to the Conceptual Specification

Change to a Technical Implementation

Bug Fix

**Validation Status**

*(Indicate if the change is validating or invalidating.)*

Validating

Invalidating

### Definitions

*Conceptual Specification:*

The substantive content of the DDI specification. Previously, this information has appeared in the “Tag Library Documentation” and in the comments in the DTD. The Conceptual Specification provides the logical framework of how the DDI is structured.

*Technical Implementation:*

A “schema” derived from the DDI Specification to be utilized to maintain structured metadata on a specific platform or data format. Examples of this are (for XML) DTDs, W3C Schemas, and RELAXNG. Other plausible technologies include various relational and hierarchical database schemas. There may be a significant number of comparable ways to accomplish the same validation and enforcement of the DDI Specification. There is a one-to-many mapping of the DDI Conceptual Specification to its many Technical Implementations.

*Bug Fix:*

A correction to a Technical Implementation that addresses an error that conflicts with the Conceptual Specification.

*Validating vs. Invalidating:*

Backwards-compatible vs. non-backwards-compatible. A validating (or non-invalidating) change is one that will not break the validation of a DDI instance against a Technical Implementation. Older instances of the DDI specification will still parse against the new, revised implementation; new instances using the changed item will require some form of version note. The opposite is true for an invalidating change: it will break the validation of older instances.

## Proposal for Minor Revision to/Improvement of the DDI Specification

### Description of Revision:

1. Reinstate nested categories in the DDI specification.
2. Reinstate <catgry> attributes “other” and “total”, which had been part of the nested categories model.
3. Introduce new element <catLevel> as a child of <var>, to assign a name to each category level in a category hierarchy, and link this name to the level’s number, as identified in the nesting pattern (opening and closing of <catgry> tags).

### Rationale for Change:

**Background note:** The nesting feature had been present at category level in development Version 1.3 (introducing the aggregate data model) but was removed from Version 2.0. The reason for removal was that there were two ways to specify groupings of categories (category groups and nested categories) and it was deemed desirable to enforce only one.

#### **In the present proposal, we argue that:**

- 1) Category groups are adequate for marking up “conceptual” groupings, where categories are NOT in a hierarchical relationship.
- 2) Category groups are inappropriate for marking up category hierarchies, where categories are placed in a relationship of subordination.

#### **We therefore propose to:**

- 1) Maintain category groups, but restrict their use to marking up “flat” groupings of categories.
- 2) Reinstate nested categories as the most appropriate markup model for category hierarchies.

The detailed examples below are designed to illustrate:

- our contention that category groups are not an adequate specification for category hierarchies;
- the advantages of using nested categories, and the way this model works when tested against search/retrieval software.

#### **“Flat” groupings of categories versus category hierarchies.**

In social science codebooks we come across two different types of category groupings:

## Type A.

(Example extracted from Euro-barometer 10, ICPSR #7728, Variable 0100)

### SELF-EMPLOYED :

- 01. FARMERS, FISHERMEN (SKIPPERS)
- 02. PROFESSIONAL - LAWYERS, ACCOUNTANTS, ETC.
- 03. BUSINESS - SHOPOWNERS, CRAFTSMEN, PROPRIETORS

### EMPLOYED :

- 04. MANUAL WORKERS
- 05. WHITE COLLAR - OFFICE WORKERS
- 06. EXECUTIVES, TOP MANAGEMENT, DIRECTORS

### NOT EMPLOYED :

- 07. RETIRED
- 08. HOUSEWIVES, NOT OTHERWISE EMPLOYED
- 09. STUDENTS, MILITARY SERVICE
- 10. UNEMPLOYED

This example is typical for microdata. The categories are only grouped by concept, in the study documentation, and they do NOT form a hierarchy: “Self-employed”, “Employed” and “Not Employed” are NOT categories (they do not stand for entries in the data file) but simply labels entered in the codebook with the purpose of classification. The actual categories are not in a hierarchical relation.

## Type B.

(Example extracted from U.S. Census 2000, SF4, PCT86)

<b>-Management, professional and related occupations</b>	<b>(Catgry C1)</b>
<b>-Management occupations</b>	<b>(Catgry C2)</b>
<b>-Top executives</b>	<b>(Catgry C3)</b>
<b>-Financial managers</b>	<b>(Catgry C4)</b>
<b>-Business and financial operations occupations</b>	<b>(Catgry C5)</b>
<b>-Computer and mathematical occupations</b>	<b>(Catgry C6)</b>
<b>-Architecture and engineering occupations</b>	<b>(Catgry C7)</b>
<b>-Architects</b>	<b>(Catgry C8)</b>
<b>-Engineers</b>	<b>(Catgry C9)</b>
<b>-Legal occupations</b>	<b>(Catgry C10)</b>
<b>-Education, training and library occupations</b>	<b>(Catgry C11)</b>
<b>-Teachers</b>	<b>(Catgry C12)</b>
<b>-Librarians</b>	<b>(Catgry C13)</b>

This type of grouping is commonly found in aggregate data. All of the entries listed above are “real” categories, represented in the data file. They form a hierarchy, with categories at lower levels being directly subordinate to the categories above them. When the data measure is additive (i.e. sums, counts, etc.), the upper level categories will represent “totals” of their

subordinates. For example, population counts for “Top executives” and “Financial managers” will add up to the entry for “Management occupations,” etc.

### **Marking up “flat” or conceptual groupings with category groups.**

Category groups are adequate for marking up conceptual category groupings, where there are no hierarchical relationships among categories. Here is a sample markup for the Type A example shown above:

```
<catgryGrp catgry="C1V100 C2V100 C3V100">
    <labl>SELF-EMPLOYED</labl></catgryGrp>
<catgryGrp catgry="C4V100 C5V100 C6V100">
    <labl>EMPLOYED</labl></catgryGrp>
<catgryGrp catgry="C7V100 C8V100 C9V100 C10V100">
    <labl>NOT EMPLOYED</labl></catgryGrp>

<catgry ID="C1V100"> <catValu>1</catValu>
    <labl> FARMERS, FISHERMEN (SKIPPERS) </labl></catgry>
<catgry ID="C2V100"> <catValu>2</catValu>
    <labl>PROFESSIONAL - LAWYERS, ACCOUNTANTS, ETC. </labl></catgry>
<catgry ID="C3V100"> <catValu>3</catValu>
    <labl> BUSINESS - SHOPOWNERS, CRAFTSMEN, PROPRIETORS </labl></catgry>
<catgry ID="C4V100"> <catValu>4</catValu>
    <labl>MANUAL WORKERS </labl></catgry>
<catgry ID="C5V100"> <catValu>5</catValu>
    <labl>WHITE COLLAR - OFFICE WORKERS</labl></catgry>
<catgry ID="C6V100"><catValu>6</catValu>
    <labl>EXECUTIVES, TOP MANAGEMENT, DIRECTORS </labl></catgry>
<catgry ID="C7V100"> <catValu>7</catValu>
    <labl> RETIRED </labl></catgry>
<catgry ID="C8V100"> <catValu>8</catValu>
    <labl>HOUSEWIVES, NOT OTHERWISE EMPLOYED</labl></catgry>
<catgry ID="C9V100"> <catValu>9</catValu>
    <labl>STUDENTS, MILITARY SERVICE </labl></catgry>
<catgry ID="C10V100"> <catValu>10</catValu>
    <labl>UNEMPLOYED</labl></catgry>
```

Category IDs are referenced in the description of every group. That is sufficient information to enable a piece of software to recreate and display the groups, because all the software has to do is to affix the category group label to the categories referenced in the catgry attribute.

### **Category groups are not appropriate for marking up category hierarchies**

Using category groups is not an appropriate way of describing category hierarchies, because it is not possible to precisely tag the subordinate relationships within a hierarchy and to attribute specific lower level categories to specific upper level categories.

In multilevel hierarchies (**Type B** groupings, as exemplified on p. 3), if a level 2 group is described as including a number of categories, and one or several level 3 groups are also

described as including a number of categories, the current category groups model **does not allow us to preserve the links between the two levels**, since we cannot specify which categories in the level 2 group form their own subgroups at level 3.

Using category groups to mark up our Type B example, we would create a group called “**Total**” that would include only C1, then a group called “**Management, professional and related occupations**” that would include C2, C5, C6, C7, C10 and C11, and three groups called “**Management occupations**” (C3 and C4), “**Architecture and engineering occupations**” (C7 and C8), and “**Education, training and library occupations**” (C12 and C13):

```
<catgryGrp ID="G1" catgry="C1" levelno="1">
<labl>Total</labl>
</catgryGrp>
<catgryGrp ID="G2" catgry="C2 C5 C6 C7 C10 C11" levelno="2">
<labl>Management, professional and related occupations</labl>
</catgryGrp>
<catgryGrp ID="G3" catgry="C3 C4" levelno="3">
<labl>Management occupations</labl>
</catgryGrp>
<catgryGrp ID="G4" catgry="C8 C9" levelno="3">
<labl>Architecture and engineering occupations</labl>
</catgryGrp>
<catgryGrp ID="G5" catgry="C12 C13" levelno="3">
<labl>Education, training and library occupations</labl>
</catgryGrp>
```

In the markup, each of the groups appears as containing a number of identified categories. The levels are correctly represented, but there is no information as to which categories from level 2 the level 3 groups need to be linked to (or, for that matter, that C1 “includes” C2, C5, C6, C7, C10, and C11).

Software trying to recreate the hierarchy for search/display purposes will look at the IDREFs contained in the markup, NOT at the labels (which are free text and can basically be “anything” or nothing). It is quite obvious that in the markup there is nothing to indicate, for example, that the “parent” of C8 and C9 is in fact C7, and not C2 or C6, or C10, etc.

### **Nested categories are an adequate markup model for category hierarchies**

Nested categories are the most appropriate way of describing even the most complex hierarchies, with both levels and subordination patterns immediately recognizable in, and retrievable from, the nesting structure (**see explanation following markup example below**). This model also allows the inclusion of aggregate data “Totals” as a separate category, complete with its own value, at the uppermost level of a hierarchy.

The markup for our Type B example above will look like this:

```
<catgry><catValu>1</catValu>
<labl> Management, professional and related occupations </labl>
  <catgry><catValu>2</catValu>
  <labl> Management occupations</labl>
    <catgry><catValu>3</catValu>
    <labl> Top executives</labl></catgry>
    <catgry><catValu>4</catValu>
    <labl> Financial managers</labl></catgry></catgry>
  <catgry><catValu>5</catValu>
  <labl> Business and financial operations occupations </labl></catgry>
<catgry><catValu>6</catValu>
<labl> Computer and mathematical occupations</labl></catgry>
<catgry><catValu>7</catValu>
<labl> Architecture and engineering occupations </labl>
  <catgry><catValu>8</catValu>
  <labl>Architects</labl></catgry>
  <catgry><catValu>9</catValu>
  <labl>Engineers</labl></catgry></catgry>
<catgry><catValu>10</catValu>
<labl> Legal occupations</labl></catgry>
<catgry><catValu>11</catValu>
<labl> Education, training and library occupations </labl>
  <catgry><catValu>12</catValu>
  <labl>Teachers</labl></catgry>
  <catgry><catValu>13</catValu>
  <labl>Librarians</labl></catgry></catgry></catgry>
```

The hierarchy levels are recognizable directly from the nesting structure. A piece of software (and the human eye) will identify the hierarchy levels by looking at the position of each beginning- and end-tag of the <catgry> elements. The first category for which a tag is opened, (as well as any other categories for which a tag is opened *after* the end-tag for the first category), will represent the first level of a hierarchy (highlighted in red in the example above). The categories for which tags are opened and closed WITHIN the tags for first-level categories will represent the second level of the hierarchy (highlighted in blue in the example above). Categories whose beginning- and end-tags are placed WITHIN the tags for second-level categories will represent the third level of the hierarchy (highlighted in green in the example above), etc., etc.

### **New <catLevel> element designed to name the category levels**

The main function of the new <catLevel> element that we propose to introduce is to assign a *\*name\** (or “label”) to each level of the hierarchy. This *\*naming\** of each level is necessary for recreating/redisplaying the hierarchy, usually in the form of a table.

In a hierarchy like

```
--Country X (catgry C1)
  -- County 1 (catgry C2)
  -- County 2 (catgry C3)
    --Municipality A (catgry C4)
    --Municipality B (catgry C5)
  -- County 3 (catgry C6)
  -- County 4 (catgry C7)
    --Municipality X (catgry C8)
    --Municipality Y (catgry C9)
--Country Y (catgry C10)
```

all level 1 categories would fall under a general name like “Countries”, all level 2 categories would fall under “Counties”, and all level 3 categories would fall under “Municipalities”. The <catLevel> elements for this particular variable would look like this:

```
<catLevel levelnm="Countries" levelno="1"/>
<catLevel levelnm="Counties" levelno="2"/>
<catLevel levelnm="Municipalities" levelno="3"/>
```

Software trying to reconstruct such a hierarchy would first identify each level by looking at the position of the beginning- and end- tags of the <catgry> elements, as explained under 1. above. It would then match each level with the name assigned to it within the <catLevel> element, that also contains the level number.

In geographic hierarchies, <catLevel> would also support the geoMap element, that links maps to specific levels by referencing the levels’ numbers. The <catLevel> element would thus provide a name (or “label”) for those levels that are linked to a map.

(While geographic maps \*could\* be referenced within the new <catLevel> element, thus making the element <geoMap> unnecessary, we do not propose to make such a change to the DDI at this point, because it would be invalidating.)

#### **Additional Information (optional):**

Category attributes “other” and “total” will also be reinstated, as being part of the nested categories model.



```

<catgry><catValu>1</catValu>
<labl>Country X</labl>
  <catgry><catValu>2</catValu>
  <labl>County 1</labl></catgry>
  <catgry><catValu>3</catValu>
  <labl>County 2</labl>
    <catgry><catValu>4</catValu>
    <labl>Municipality A</labl></catgry>
    <catgry><catValu>5</catValu>
    <labl>Municipality B</labl></catgry></catgry> </catgry>
  <catgry><catValu>6</catValu>
  <labl>Country Y</labl></catgry>

```

***[The following paragraph to be inserted in the Tag Library description for category groups]:***

Use <catgryGrp> to mark up “flat” groups, in which categories are only grouped conceptually, as in the following instance:

SELF-EMPLOYED:

- 01. FARMERS, FISHERMEN (SKIPPERS)
- 02. PROFESSIONAL - LAWYERS, ACCOUNTANTS, ETC.
- 03. BUSINESS - SHOPOWNERS, CRAFTSMEN, PROPRIETORS

EMPLOYED:

- 04. MANUAL WORKERS
- 05. WHITE COLLAR - OFFICE WORKERS
- 06. EXECUTIVES, TOP MANAGEMENT, DIRECTORS

NOT EMPLOYED:

- 07. RETIRED
- 08. HOUSEWIVES, NOT OTHERWISE EMPLOYED
- 09. STUDENTS, MILITARY SERVICE
- 10. UNEMPLOYED

2) Use element catLevel to specify name (attribute levelnm) and number (attribute levelno) of the levels in a category hierarchy. The \*naming\* of each level is necessary for recreating/redisplaying the hierarchy, usually in the form of a table. Software trying to reconstruct a hierarchy would first identify each level by looking at the position of the beginning- and end- tags of the <catgry> elements. Then, it would match each level with the name assigned to it within the <catLevel> element, that also contains the level number. In geographic hierarchies, <catLevel> would also support the geoMap element, that links maps to specific levels by referencing the levels’ numbers. The <catLevel> element would thus provide a name (or “label”) for those levels that are linked to a map. When geoMap element is present, levelno should refer to the levelno attribute in geoMap. This is an “empty” element, that only contains the attributes listed below.

Example: <catLevel levelnm="Country" levelno="1"/>  
<catLevel levelnm="Counties" levelno="2"/>  
<catLevel levelnm="Municipalities" levelno="3"/>

- Optional
- Repeatable
- Attributes: ID, xml:lang, source, levelnm, levelno

## Proposal for Revision to/Improvement of the DDI Specification

### Process

Bug fixes and validating changes may be routed through an expedited process. All other changes are handled through the full standards review process outlined in the Alliance Bylaws.

#### *Expedited Process:*

1. *Submission:* The Working Group sends the proposal to the Chair of the Expert Committee for review.
2. *Technical Review:* The Chair conducts a Technical Review involving the Alliance Structural Reform Working Group and/or the XML consultant. This review brings to light the implications of the proposed change on various Technical Implementations of the DDI and its “fit” with the current Conceptual Specification. The Expert Committee Chair sends the proposal to the Alliance Director.
3. *Director Review:* For some minor changes, the Director has the discretion to implement the change immediately after the Technical Review if the Review so recommends.
4. *Public Review:* The Director sends the proposal via email to the Expert Committee and posts the proposal on the public site ([ddialliance.org](http://ddialliance.org)) and on the Expert Committee’s private bulletin board for a period of two weeks. Public review and comment is solicited through the *ddi-users* listserv (Expert Committee members should all join that group to monitor public feedback). Expert Committee members may comment publicly or privately.
5. *Voting:* After two weeks of review and comment, each Expert Committee member has two weeks to vote “Yes” or “No.” A “Yes” vote indicates that the validity and usefulness of the proposed modification have been demonstrated and that the specification should now be accepted as a part of the DDI standard. A “No” vote indicates the case has not yet been made for the proposed modification. A “No” vote must be accompanied with comments to explain the vote. For the specification to be approved, at least one-half of the Expert Committee must vote and two-thirds of those voting must vote “Yes.” The Director will consider the votes and make an informed decision as to whether to accept the specification or to restart the process at some earlier stage.
6. *Change Implementation:* If the proposal is approved by the vote of the members of the Expert Committee, by the end of the two-week voting period the Director will ordinarily begin the process of incorporating the proposal into the DDI specification. However, the Director has the right to veto the proposal with a full explanation for that decision to the members of the Expert Committee. That veto, in turn, can be overridden by a two-thirds vote of the Steering Committee.
7. *Publication:* If the proposal is accepted, a revised DDI specification is posted on the public site as the new development version or, if a development version already exists, the change is incorporated. The approved change will then be incorporated into the next scheduled release of a new production version of the DDI specification.

#### *Full Standards Review Process and Procedures:*

1. See the DDI Procedures Manual and the Alliance Bylaws at <http://www.icpsr.umich.edu/DDI/org/bylaws.html/>.