A Variable Cascade

# Problem Statement

The DDI-Lifecycle standard is intended to address the metadata needs for the entire survey lifecycle. This particular document is dedicated to a description of variables as part of the DDI-Lifecycle. It contains a UML (Unified Modeling Language) class diagram of a model for describing variables, and the model is part of the larger development effort called DDI Moving Forward to express DDI-Lifecycle using UML.

Typical models for describing variables take advantage of much reuse, and the model provided here is no exception. It is reuse that makes metadata management such an effective approach. However, effectiveness is due to other factors as well, and an important one is to keep the number of objects described to a minimum. For finding relevant data is much more complicated as the number rises.

For variables, reusable descriptions are brittle in the sense that if one of the related records to a variable changes, then a new variable needs to be defined. This is especially true when considering the allowed values (the Value Domain) for a variable. Many small variations in value domains exist in production databases, yet these differences are often gratuitous (e.g., simple differences in the way some category is described that do not alter the meaning), differences in representation (e.g., simple changes from letter codes to numeric ones), or differences in the way missing (or sentinel) values are represented.

Gratuitous differences in expressions of meaning are reduced or eliminated by encouraging the usage of URIs (Uniform Resource Identifiers) to point to definition entries in taxonomies of terms and concepts. The principle of “write once – use many”, very similar to the idea of reuse, is employed. Pointing to an entry rather than writing its value eliminates transcription errors and simple expression differences, promotes comparability, and ensures interoperability.

Differences in representations, including codes, are simplified by separating them from the underlying meaning. This is equivalent to the terminological issue of allowing for synonyms and homonyms of terms. Through reuse, all representations with the same meaning are linked to the same concept. This is achieved through the use of Value Domains and Conceptual Domains in the model presented here.

Sentinel values are important for any processing of statistical or experimental data, as there are multiple reasons some data are not obtainable. Typically, these values are added to the value domain for a variable. However, each time in the processing cascade the list of sentinel values changes, the value domain changes, which forces the variable to change as well. Given that each stage of the processing cascade requires a different set of sentinel values due to processing requirements, the number of variables mushrooms. And this variable cascade is unmanageable and unsustainable.

The model developed here is based on two important standards for the statistical community, GSIM (Generic Statistical Information Model) and ISO/IEC 11179 (Metadata Registries). In fact, the model in the conceptual section of GSIM is based closely on the metamodel defined in ISO/IEC 11179. Both models help to perpetuate the problems described here, if each standard is followed in a conforming way. They reduce redundancy by separating value domains and conceptual domains. However, they do not directly support the use of URIs and they do not separate value domains from sentinel value lists. Also, they do not fully exploit the traceability that inheres in certain relationships between the value and conceptual domains to create a continuum of connected variables that further reduces redundancy.

The purpose of this document is to present a model that significantly reduces the overhead described above. In particular, we separate the sentinel values from the substantive ones. This separation allows us to greatly reduce the number of value domains, and thus variables, that need to be maintained. With this separation, now there are three classes of connected variables in which the ***represented variable*** specializes a ***conceptual variable*** by adding a ***value domain***, and the ***instance variable*** specializes the represented variable by adding a ***sentinel value domain***.

# The Variable Cascade Proposal

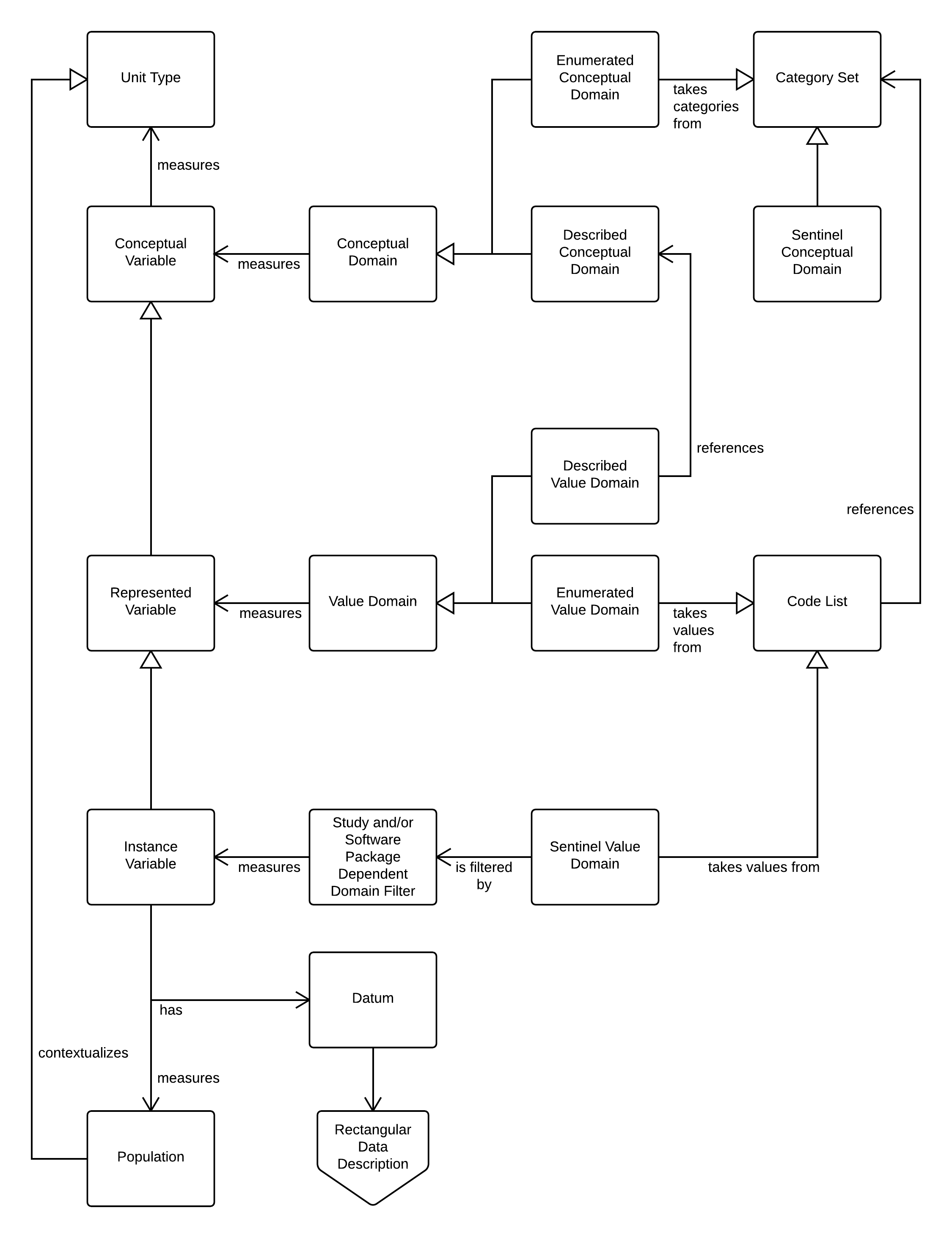


Figure : Variable Cascade Model

# Three Use Cases

## Detergents

Imagine we are assessing environmental influences at the household level. One question we might ask is “What detergents are used in the home?” In connection with this question we prepared show cards. Each show card lists a series of detergents. There are multiple show cards because products vary by region. Each show card corresponds to a code list. There is overlaps among the code list but there are differences too.

In our model there is a ***conceptual variable***. It has an ***enumerated conceptual domain*** that takes its categories from a ***category set***. Here the category set is an unduplicated list of detergents taken from all the show cards put together. The ***conceptual domain*** might be exposure to chemicals in household detergents. The ***unit type*** might be households.

In our survey we ask a question corresponding to multiple ***represented variables***, one corresponding to each show card. Each represented variable is measured by an ***enumerated value domain*** that takes its values from the show card ***code list***.

***All the represented variables here are children of the same conceptual variable***. This is the main point of the example: a conceptual variable under the right conditions can connect multiple represented variables.

## Sentinel Values #1

**Introduction**. The represented variable code list in our model excludes sentinel values. Sentinel values were introduced into ISO/IEC 11404 in 2007. ISO/IEC 11404 describes a set of language independent datatypes and defines a sentinel value as follows: *a sentinel value is a “signaling” value such as* ***nil****,* ***NaN*** *(not a number),* ***+inf*** *and* ***–inf*** *(infinities), and so on.* Depending on the study, sentinel values may include missing values. ISO 21090 is a health datatypes standard based on ISO/IEC 11404. ISO 21090 identifies 15 “nullflavors” that correspond to the concept of sentinel values introduced in ISO 11404[[1]](#footnote-1).

In our model the ***instance variable*** adds a ***sentinel value domain*** to the represented variable from which it inherits. In the process it grows the code list it inherits from the represented variable to include a set of sentinel values. These sentinel values reference a category set of sentinels called the ***sentinel conceptual domain***. The sentinel values included in any instance variable need not cover all the members of the sentinel conceptual domain. Instead they may refer just to a subset.

**Use Case**. In our first sentinel values use case, during data acquisition, we ask a question that allows don’t know and refused, which an interviewer may ask or not, depending on the skip logic. We create an instance variable corresponding to this question based on a represented variable. The instance variable includes sentinel values. Note that the ***value domain*** of the represented variable need not be an ***enumerated value domain***. Instead it can be a ***described value domain***. We choose to include three sentinel values corresponding to three ISO 21090 nullflavors:

|  |  |
| --- | --- |
| Nullflavor | Description |
| UNK | A proper value is applicable, but not known. *Corresponds to Refused.* |
| ASKU | Information was sought but not found (e.g., patient was asked but didn’t know). *Corresponds to Don’t Know.* |
| NA | No proper value is applicable in this context (e.g., last menstrual period for a male) |

Subsequently, we prepare the collected data for processing by SAS. SAS has its own set of sentinel values. For each sentinel category the data acquisition instance variable accounts for, SAS has its own set of sentinel values. As a consequence, the answers that correspond to sentinel values are different, and in the process of constructing a SAS file, a second instance variable is created for the purposes of data processing.

However, ***both the data acquisition instance variable and the data processing instance variable are children of the same represented variable***. That is the point of this example.

### Sentinel Values #2

Sometimes there is this progression of instances variables (IV):

Figure : Progression of Instance Variables

Recall that the purpose of the data processing instance variable was to prepare the raw data for a software package that had its own special codes for sentinel values. Among practitioners, however, “it is not uncommon to generate additional variables (dummy or index variables) representing these missing reasons (e.g. 1=”refused” 0= “other”), turning this metadata into data.”

Arguably nothing has changed except, in the event of the Data Refinement IV, now we can tabulate valid and missing values together.

In terms of the model, each instance variable (IV) has its own ***code list*** and that code list consists of an ***enumerated value domain*** and a ***sentinel value domain***. Across the progression the sentinel value domain may vary. However, because the variable sentinel value domain is mapped to the same subset of the ***sentinel conceptual domain*** in all three instances and there are no other changes, all three instance variables are equivalent and can be traced back to the same represented variable.

**Postscript**. Both the sentinel conceptual domain and the sentinel value domain are universal in scope. The sentinel conceptual domain is the set of all sentinel categories. Likewise, the sentinel value domain is the set of all values that reference the sentinel conceptual domain. However, only a subset of sentinel values are, as a rule, used in the measurement of an instance variable. Furthermore, it sometimes happens that the same subset is reused with all the variables in a study and/or in connection with a specific software package. In order to facilitate this type of reuse, the model supports the construction of a filtered set of sentinel values.

1. See section 7.1.4 at <http://tc215.behdasht.gov.ir/uploads/244_514_ISO_DIS_21090(E).pdf> [↑](#footnote-ref-1)