Data constraint coding protocol

Definitions

1. A **data constraint** is a restriction on the set of possible values of an *abstract variable*.
2. An **abstract variable** is a value in the software domain that usually corresponds to a piece of information in the real world, e.g., a person's name or a business address. Since these are problem *domain* variables, they are independent of any software platform or programming language, and they only have a name and a value, which can be of any type.

Data constraints

Our goal is coding every sentence or phrase that *explicitly* restricts the set of allowable values for any abstract variable. The text in which constraints are to be identified will correspond to textual artifacts of software systems, such as manuals and tutorials.

Abstract variables in these texts are rarely referenced directly, but are instead mentioned when talking about configuration properties or data that the software must handle. For example, consider the following sentences that imply the existence of an abstract variable:

- The user can input *their address*.
  - The sentence implies that "address" is a property of the user.
- The *amount of patients* that is displayed can be configured.
  - The sentence says the "amount of patients" can be changed, which means this is a property of the system.
- The page will show the *shipment's date.*
  - "Date" is a property of a shipment.

These sentences imply that the system stores and/or processes information that corresponds to these abstract variables. Sometimes this documentation directly specifies *constraints* in the value of an abstract variable, that is, it restricts the possible values to a single one or a specific set. This restriction is what we call a *data constraint*.

We now present some examples of data constraints. The original text appears in quotes, and the fragment that contains the data constraint is underlined. The part in bold after the arrow corresponds to the extracted constraint (the same constraint expressed in a clearer language):

- *"*If you specify a repeat count greater than 20, a confirmation dialog box will be displayed*"* → **Repeat count is > 20**

- “By default, the last 100 edits are retained; older edits cannot be undone.” → **Edits retained = 100**
- “The end of the interval is always greater than or equal to the start” → **Interval end >= Interval start**

Positive examples

The ways in which data constraints may be defined includes, but is not limited to:

- Specifying the value directly:
  - “X is the number of minutes to display along the bottom of the helicorder. Default is 15 minutes.”
  - “The Max number of backups setting determines the number of backups to save. Setting this to zero disables the backup feature.”
  - “The default date is 1582-10-15, as defined by Pope Gregory XIII.”
- Specifying an *exhaustive* set or range of allowable values:
  - “Since Ant 1.8.0, you may instead use property expansion; a value of true (or on or yes) will enable the item.”
  - “dayOfWeek | 1 (Monday) | 7 (Sunday)”
  - “We also prevent the entry of dates before year 1.”
- Implied Boolean values will be considered abstract variables:
  - “An alive patient “
    - Alive vs not alive is an implied boolean value.
  - “A project can have a set of tokens that might be automatically expanded if found when a file is copied, when the filtering-copy behavior is selected in the tasks that support this.”
    - Selected vs not selected is an implied boolean value.
  - “if the module-A-present property is set (to any value)”
    - Set vs not set is an implied boolean value
  - “if an object with that key already exists”
    - Exists vs does not exist is an implied boolean value.

Negative examples

However, some sentences that might look like constraints are **NOT** considered so. Some examples are:

- Concrete examples or hypothetical scenarios:
  - “If the package weight is 5, for example, the command will output ‘true’.”
- Non-exhaustive sets of values
  - “If “i” is an integer like 2, 3, or 5…”
- Actions or decisions that are outside of the system’s control

  - “If the patient displayed is not the one that the user intended, the user will go back to the search screen”
  - “If the field is never accessed by multiple threads, rename it to use lowerCamelCase.”
- User intentions or possibilities
  - “In case the file should automatically be saved on each change, you can direct the system to do so.”
  - “The system can be configured to exit if an input is invalid”
- Required values: saying that a value is NOT required does not constitute a constraint:
  - “The name field is not required in this form”

Since this list is not exhaustive, please keep in mind that the sentence has to explicitly refer to an abstract variable AND explicitly restrict the set of allowable values for it to be a constraint.

Note that we also **exclude** constraints that refer to **string patterns**, even though these could be considered data constraints:

- “If the line starts with a tab character.”
- “The password must contain at least one uppercase letter.”

We also **exclude** constraints that deal with the **ordering of items in collections**:

- “If the object is in the last position of the list…”

Also **exclude** constraints that deal with **UI elements**:

- “Un-toggling the pick button will hide the Pick Menu again.”