

Slowly Changing Dimensions – a Pattern Language for Coping With Change in Analytical Information Processing

Hans Wegener

Swiss Re
Mythenquai 50/60, 8022 Zürich, Switzerland

Hans.Wegener@swissre.com

Robert Marti

Swiss Re
Mythenquai 50/60, 8022 Zürich, Switzerland

Robert.Marti@swissre.com

1. Motivation

In any enterprise, conducting and closing business transactions such as selling products and services, buying supplies, etc. leads to the collection of business facts that measure things such as volume, speed, and quality, of the business conducted. These business facts are in turn used to calculate various performance indicators (e.g., profit, margins, return on capital, etc.) and risk indicators (e.g., volatility of investments etc.) which serve as one of several inputs to steer the future direction of the enterprise, including establishing targets and limits for the next business cycle.

The connection between collected “raw” measures in business facts on one hand, and performance indicators on the other can be quite intricate, involving complex calculation steps. In a large enterprise with partly autonomous business entities, this complexity is compounded when business facts collected in various business processes and/or locations only match partially, e.g., due to different encodings of currencies, countries, business lines etc. In addition, the business environment is changing over time, e.g., due to the development of new products, changes of customer characteristics, and internal reorganizations, making it difficult to reconcile and compare business facts collected over the years.

In the following, we outline an approach to cope with these issues, especially dealing with change in the business environment over time. We shall focus on changes to attributes which categorize business transactions, which – following e.g. [7, 8] – are called dimensional attributes, given that these attributes define an n -dimensional space or hypercube for the collected measures (see also below)*. Such changes demand specific treatment because they often affect time-series analysis† of performance indicators. Imagine that a business application deals with an abstraction of “Germany”. When the notion of “Germany” changes (as happened with reunification in 1990), some of the existing data managed by the application – e.g. old gross national product figures – may have to be adapted to the new notion. (The same is true for former Yugoslavia, Czechoslovakia, Congo, etc., as well as for changes in the profit center structure of a company and changes to the structure of companies caused by mergers and acquisitions.)

The execution and closing of business transactions – selling a product or service, buying supplies, signing and terminating employee contracts etc. – is the main source of business facts that need to be recorded in an enterprise. Further data sources external to the enterprise – mostly changes in the geo-political and economic environment such as the rise and fall of inflation, interest rates, mergers and acquisitions of clients and competitors, etc. – need to be tracked and recorded as well.

A business fact can be viewed as a record that typically consists of a series of attribute values including

- A unique identifier to distinguish this business transaction from other transactions
- A transaction date stating when the data pertaining to the transaction was recorded
- An effective date stating when the result of closing the transaction is effective, e.g. the time period during which an insurance coverage is in effect
- Measures which indicate quantities or monetary amounts involved, e.g. total sum insured, deductible, premium due, base salary, ...

* Given that changes to dimensional attributes (addition and removal as well as hierarchical restructuring of the values of such an attribute) is relatively slow with respect to the rate at which business transactions are captured, the issues related to the change of dimensional attributes has come to be called „slowly changing dimensions“.

† A time-series is a sequence of data points, measured typically at successive times, spaced apart at uniform time intervals. Time-series analysis comprises methods that attempt to understand such time-series, often either to understand the underlying theory of the data points, or to make forecasts [12].

- Dimensional attribute values that describe the context of a business transaction, e.g. the identification of the business partner, the product or service offered or sold, the responsible profit center, etc.

For some measures, we do not know (or care) how their value was arrived at – whether they were computed from other data, are the result of a measuring or counting process, represent an estimate, or were obtained from a third party. Such a measure is called an observed measure. In contrast, a derived measure is computed from observed measures and other derived measures, e.g., by applying a formula describing how it is computed.

For example, at least for the purpose of information integration and analysis, the annual base salary of an employee is usually treated as an observed measure – even though the annual base salary might actually have been computed according to a complex formula which takes other measures and parameters into account, e.g., highest educational degree reached, age, number of years experience in the field, rank, and number of years employed.

On the other hand, total annual compensation, computed from the observed measures annual base salary, family and child allowance, and annual bonus, is a derived measure.

Dimensional attributes* are attributes which describe the context of business transactions: They refer to objects such as business partners, products or services, (internal) profit centers, responsible employees etc. (Note that in rare cases, an attribute may play different roles in different transactions: For example, when underwriting an insurance contract, the contract identifier helps to identify this business transaction, together with the year of coverage. This same contract identifier will be used as a dimensional attribute for claims processing, as every claim must refer to an existing insurance contract.)

Dimensional attributes typically assume values taken from a pre-defined set of values. This set of legal values is usually defined by an enumeration of values, which is more or less static, e.g.,

1. CurrencyCode = { ‘USD’, ‘GBP’, ‘EUR’, ‘JPY’, ‘CHF’, ... }
2. CreditRating = { ‘AAA’, ‘AA’, ‘A’, ‘BBB’, ... }
3. LineOfBusiness = { ‘Any Line’, ‘P&C Line’, ‘Property’, ‘Casualty’, ... , ‘L&H Line’, ... }

or, somewhat more dynamically, by a set of object identifiers for specific business objects, e.g.

- Partner = { x | “x is an id of a currently active business partner” }[†]

For some dimensional attributes, a (more or less) natural ordering relationship can be imposed on the elements of the set of legal values. For example, ‘AAA’ is a better credit rating than ‘AA’, etc.

For other dimensional attributes, a taxonomy relating more specific terms to more general ones can be imposed on its attribute values. For example, for LineOfBusiness, ‘Any Line’ is more general than ‘P&C Line’ and ‘L&H Line’, while ‘P&C Line’ in turn is more general than ‘Property’ and ‘Casualty’.

In the following, the structure of a dimensional attribute is considered to consist of the set of values that belong to the dimensional attribute and, if and when needed, also the ordering and taxonomical relationships among these values.

2. Patterns for Coping with Changes to a Single Attribute

2.1 Overview

2.1.1 Common Problems

Events within and outside an enterprise may lead to the decision to change how it looks at the world. As an effect, the structure of a dimensional attribute will be changed. Such changes carry the potential to affect (derived) measures computed at an earlier point, i.e. based on a historic attribute structure. Which structural change to an attribute is taken as commensurate to the business change in the world thus determines how the amended view must be computed. One common problem therefore is to understand

What is the impact of a structural change to a dimensional attribute on the amended view, i.e. how do (derived) measures change, if at all?

* Instead of “dimensional attribute”, papers on statistical analysis often use the term “categorical attribute”.

[†] Technically, such a set could be defined by an expression of relational algebra or by an SQL SELECT statement, e.g., SELECT PartnerId FROM Partners WHERE Status = ‘active’.

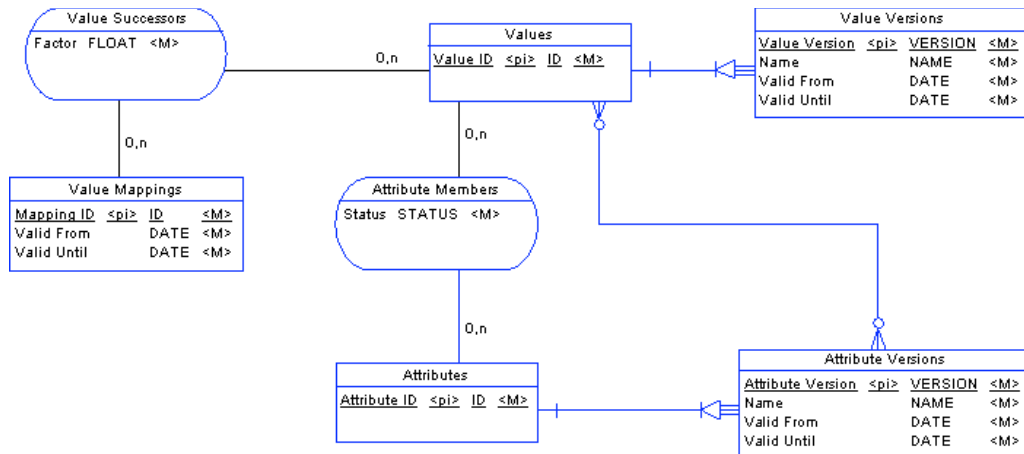


Figure 0: Conceptual data model example of a combined reference and mapping data store.

Changes can become rather complex; comprehensive, composite changes may be just as appropriate to the change as a group of simple, atomic ones. However, the way composite changes are put together may affect the way history is looked at. Once (at least) two alternatives have been identified, it becomes beneficial to know

What are the consequences of different courses of action, i.e. different ways of interpreting a change? How does this affect the analysis of historic data?

We assume for all changes that knowledge of them is obtained before their date of effectiveness. This simplification holds true for most of real life. Furthermore, should it happen that one learns of a change after the fact, adopting it follows the same line of thought and course of action, only the impact may be different.

2.1.2 Pattern Summary

The patterns address changes to a single attribute. Such changes reflect different a refined or altered understanding of how a single business category, i.e. analysis dimension, should be looked at. These are

- Introduce Category
- Invalidate Category
- Integrate Category with Existing
- Integrate Category into New
- Partially Split Category
- Fully Split Category

The first is an atomic change; all others are composite changes, because they involve more than one-dimensional value.

2.1.3 Running Example

We use relational databases as technology substrate. The conceptual model of reference and mapping data store contains the entity types depicted in Figure 0. The model implies a certain way of storing reference and mapping data change history, but does not restrict the applicability of the patterns. For the purpose of simplicity, we are taking a few shortcuts and assume that all data stores (reference data, mapping data) are part of one single database instance, set apart by their schema names (REF, MAP). The physical model is depicted in Figure 0.

The table names are, from top left to bottom right: MAP.VALUE_MAPPINGS, MAP.VALUE_SUCCESORS, REF.VALUES, REF.VALUE_VERSIONS, REF.VALUE_LIFECYCLE_STATUS, REF.ATTRIBUTE_MEMBERS, REF.ATTRIBUTES, and REF.ATTRIBUTE_VERSIONS.

For all the DML examples below, we assume that a new version of the attribute in question has already been created, holding the same values. The DML for that is (with example validities and identifiers):

```
-- create new version of
-- attribute 2344
INSERT INTO REF.ATTRIBUTE_VERSIONS (
    ATTRIBUTE_ID,
    ATTRIBUTE_VERSION,
    VALID_FROM,
    VALID_UNTIL
```

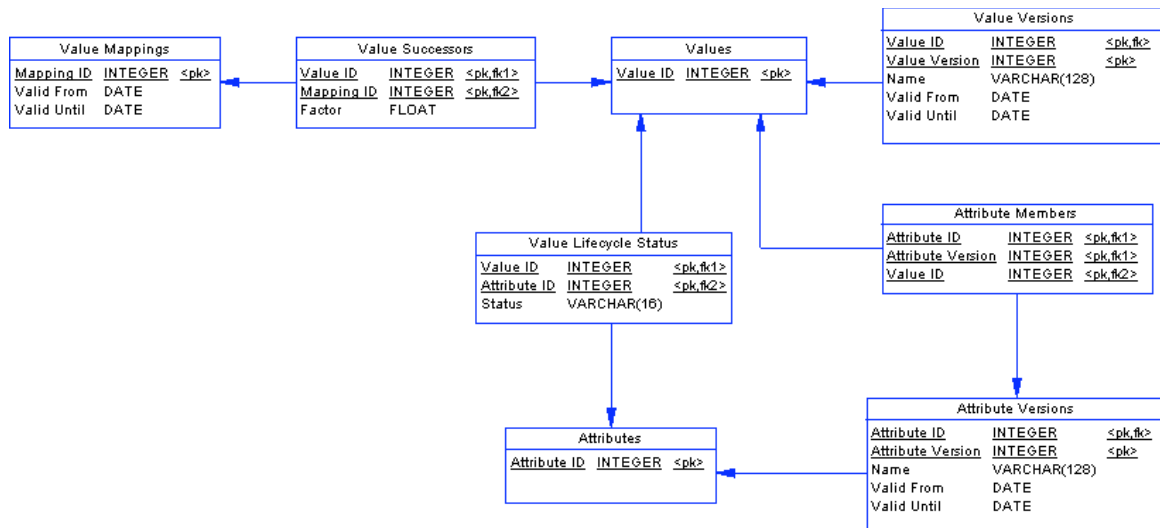


Figure 0: Physical data model of the example reference and mapping data store.

```

)
SELECT
  ATTR.ATTRIBUTE_ID,
  MAX(ATTR.ATTRIBUTE_VERSION) + 1,
  '2001-01-01',
  '2008-12-31'
FROM
  REF.ATTRIBUTE_VERSIONS SV
WHERE
  ATTR.ATTRIBUTE_ID = 2344
GROUP BY
  ATTR.ATTRIBUTE_ID;
-- copy members of attribute
-- 2344 from previous into
-- new version
INSERT INTO REF.ATTRIBUTE_MEMBERS (
  VALUE_ID,
  STATUS,
  ATTRIBUTE_ID,
  ATTRIBUTE_VERSION
)
SELECT
  SV.VALUE_ID,
  SV.STATUS,
  ATTR.ATTRIBUTE_ID,
  MAX(ATTR.ATTRIBUTE_VERSION)
FROM
  REF.ATTRIBUTE_VERSIONS SV
WHERE
  ATTR.ATTRIBUTE_ID = 2344
  AND
  ATTR.ATTRIBUTE_VERSION = (
    SELECT
      MAX(ATTRIBUTE_VERSION) - 1
    FROM
      REF.ATTRIBUTE_VERSIONS
    WHERE
      ATTRIBUTE_ID = 2344
  );

```

2.2 Introduce Category

Introduce a value to an attribute as soon as business of a new kind needs to be understood.

2.2.1 Motivation

Business opportunities show up all the time—some of them you have seen before, others are new in nature. You engage in these opportunities, conducting transactions as they come along. At some point you recognize patterns

emerging in these transactions and try to understand them better: for example, you might want to know how profitable a particular product is or what customer segment is most interested in it. In order to perform such an analysis, you must introduce a new dimensional value and start recording transactions based on it.

It may take you some time to decide to introduce a new category—you may not recognize the new pattern early enough, or for lack of its business importance you may have decided to tentatively ignore it. Therefore, by the time of making the decision a number of business transactions will have been recorded that, in the light of the new category, would have to be re-categorized. Since the data required to perform such a re-categorization may have to be obtained manually, or worse, be irrevocably lost, this can become rather costly. The question is thus how to avoid such a situation. This pattern does not answer the question (it is beyond its scope), but instead emphasizes the consequences and forces to be kept in mind.

A new category can be part of a bigger (i.e., composite) change that entails more than just one such category. Take the case of reorganizations: some of the organizations being introduced have been designed to replace others. These cases must be treated differently to ensure historic continuity of measures computed for previously existing organizations. This pattern is not about such cases, but rather just about isolated, entirely fresh business concepts that have not been seen before. The good news about these is that they do not carry any history we would have to restate. The bad news is that we must go to great lengths to ensure the category about to be introduced really is new, which is not always such an easy matter.

2.2.2 Context

- You are using “Reference Data Store,” “Mapping Data Store,” and “Value Lifecycle.”
- You need to analyze a new aspect of business transactions.
- A value has already been introduced to represent the concept in the technical world.
- The concept associated with the value is genuinely new, independent of other concepts or changes, both currently and historically.
- It has never been an element of the attribute before.

2.2.3 Forces

- Depending on how late you introduce a new category, transactions conducted previously may not be appropriately categorized. They will either have to be re-categorized, or reported measures will not give a fully accurate picture of history.
- Typically you introduce a new category on a green field. However, it may happen that the value has to be re-introduced. In this case it must be ensured that it has no (longer) business facts associated.

2.2.4 Problem

Which way do you introduce (genuinely) new categories in the face of history that it also covers?

2.2.5 Solution

From a certain day (in the future) on, introduce the new category. Ignore historic business facts that might be categorized the same way.

Before date of effectiveness, add the new value to the attribute in the reference data store. Give it the lifecycle status “active”, valid as of date of effectiveness. No action is required on the side of the mapping data store.

2.2.6 Implementation

Based on the data model defined in Section 2.1.3, the following DML is required to implement this pattern (with example validities and identifiers):

```
-- assume a new version of the
-- attribute 2344 has been created
-- add value 15680 to
-- attribute 2344
INSERT INTO REF.ATTRIBUTE_MEMBERS (
    VALUE_ID,
    ATTRIBUTE_ID,
    ATTRIBUTE_VERSION
)
SELECT
    15680,
    ATTR.ATTRIBUTE_ID,
    MAX(ATTR.ATTRIBUTE_VERSION)
```

```

FROM
  REF.ATTRIBUTE_VERSIONS SV
WHERE
  ATTR.ATTRIBUTE_ID = 2344;
GROUP BY
  ATTR.ATTRIBUTE_ID;
-- start lifecycle of value
-- 15680 in attribute 2344
INSERT INTO REF.VALUE_LIFECYCLE_STATUS (
  VALUE_ID,
  ATTRIBUTE_ID,
  STATUS
) VALUES (
  15680,
  2344,
  'ACTIVE'
);

```

2.2.7 Consequences

- The change can be propagated through the architecture without hindrance.
- Business transactions conducted before introduction of the new value are not re-categorized.
- No history needs be restated, no measures recomputed.

2.2.8 Related Patterns

- Adding values occurs more often in the context of “Integrate Category with Existing,” “Integrate Category into New,” “Partially Split Category,” and “Fully Split Category” than in isolation.

2.2.9 Examples

Introducing a value „Nanotechnology“ or „Genetically Modified Organism“ to the attribute „Type of Peril“ in commercial liability or „Computer Hacking“ in property insurance is an instance of this pattern: it would be difficult to claim that many existing business transactions (i.e., business facts) were associated with other values which would now have to be transferred to the new ones. Hence, once these perils are covered in a company’s insurance contracts they can be added to the attribute describing them.

The German information service “Heise Online” reported recently [5] that Lycos had observed a 400% increase in the volume of unsolicited e-mail offering “Luxury Goods” over the past year. This new category of “Type of Spam” would now have to be added to the likes of “Get Rich Quick,” “Phishing,” “Health Pills,” or “Sexually Explicit.” Interestingly, as of 12 March 2004 anti-spam tool vendor Sophos [10] only recognized the spam classes

- offensive,
- scam,
- entertainment,
- financial,
- health,
- internet, and
- other content.

The ACM Computing Classification System [1] introduced a new category “B.2.4 High-Speed Arithmetic” in 1998 that did not exist beforehand. Within the category “B.2 ARITHMETIC AND LOGIC STRUCTURES“ there were four categories before 1998, none of which had any resemblance to the former.

2.3 Invalidate Category

Prepare a value for removal from an attribute.

2.3.1 Motivation

Things go out of fashion, cease to exist or slowly lose importance. At one point or another you may find that they have lost their significance for your analytical purposes. Two questions will concern you: how to deal with the outdated business category itself in terms of its use for time-series analysis, and how and when to remove it from its attribute.

When a value is replaced by (one or more) others, historic transactions associated with this value will be looked at in a different way. However, sometimes successor values are not easily identified. Forcing restatement of transactions (i.e., choosing successors at all cost) may lead to reporting of distorted figures, as the underlying facts do not really belong where they are claimed to belong. This would be confusing, for which reason you may decide to discontinue the time-series analysis for this particular value. Once you have decided so, you must prevent the creation of new business facts associated with that value, which would be confusing as well.

2.3.2 Context

- You are using “Reference Data Store,” “Mapping Data Store,” and “Value Lifecycle.”
- An existing value appears to have lost significance as a business concept for analytical purposes.
- A successor value in the same attribute has not been identified (yet).

2.3.3 Forces

- Business people often demand the continued use of outdated values for a long time. It can be difficult to get rid of such values.
- The larger the volume of business recorded for the outdated value, the more likely it is that you must find a successor so you can account for it in your analysis.
- Some data stores may choose to ignore the hint that the value is “inactive” and continue to record transactions based on it.

2.3.4 Problem

How should you treat values that have become obsolete?

2.3.5 Solution

Before date of effectiveness, change the lifecycle status of the value to “inactive.”

2.3.6 Implementation

Based on the data model defined in Section 2.1.3, the following DML is required to implement this pattern:

```
-- a new version of attribute 2344
-- will *not* be created
-- terminate lifecycle of value 324
-- in attribute 2344
UPDATE REF.VALUE_LIFECYCLE_STATUS SET
    STATUS = 'INACTIVE'
WHERE
    VALUE_ID = 324
    AND
    ATTRIBUTE_ID = 2344;
```

2.3.7 Consequences

- You have to actively manage the elimination of the invalidated category (i.e., complete removal from the attribute).
- Based on the lifecycle status, data stores are able to prevent new facts for the outdated value be added.
- Historic data associated with the value will ultimately be lost for time-series analysis.
- No history needs be restated, no measures recomputed.

2.3.8 Examples

When VW introduced its “Golf” car in 1974, the “Beetle” was still being produced. While some hailed the “Golf” as the successor to the “Beetle,” it was a strong diversion. The “Golf” series of vehicles exhibits a product line concept that allows auto part variants—engine, bodywork, brakes, etc.—to be combined as the market requests them. Hence, in terms of profitability you cannot easily compare historic sales and profitability figures from the “Beetle.” It therefore makes little sense to map historic “Beetle” to “Golf” figures. You would instead rather invalidate the value “Beetle” in the attribute “Type of Car” and ultimately drop associated business transactions from analysis. Both types of cars would be analyzed separately for their sales, profitability, or any other measure.

2.4 Integrate Category with Existing

Merge the business facts associated with two or more values into one, letting one of them survive.

2.4.1 Motivation

The environment your company operates in is subject to change as well. Markets have an influence on what you can sell. Regulators apply the law of your jurisdiction and determine what your company must report. Political decisions can affect almost any aspect of doing business. The importance of categories—like the type of products or services your company sells, processes, or reports on—ebbs and flows as an effect of these changes. It is therefore not surprising that some categories may fall out of fashion or become operationally irrelevant. While it is understood that they must be phased out, the question is what to do with existing business facts associated with them. Sometimes an outdated category has a close resemblance to another category that continues to be used, for example because they originated from a joint ancestor. From the perspective of information analysis, it is reasonable in this case to re-categorize the business facts associated with the outdated dimensional value and associate them with the survivor value.

Obviously, this way of re-interpreting history will have side-effects. Depending on how much business was associated with the outdated category (or categories), the survivor experiences a sudden spike of change that cannot be explained unless one remembers the cause. Certainly it has nothing to do with the underlying business reality associated with the survivor. As such, and this is what the pattern highlights, you should have ways of assessing the effects of merging the business of a number of categories into one survivor.

2.4.2 Context

- You are using “Reference Data Store,” “Mapping Data Store,” and “Value Lifecycle.”
- One or more concepts have ceased to be relevant from a business point of view.
- The outdated values are (closely) similar to an existing value, which continues to be relevant.
- The (observed or derived) measures aggregated across the outdated values are small in comparison to the survivor.

2.4.3 Forces

- A loss of business relevance may be temporary. You should wait a certain time to ensure a category is truly outdated. This, on the other hand, may worsen the distorting effects of re-categorization.
- If you integrate (one or more) categories with the scavenger category “Miscellaneous,” you don’t have to bother with the question whether the dip in importance is permanent, but that category may grow rather large over time, causing problems of its own.
- If the values are organized hierarchically (as is typical in MOLAP*) the parent value of a set of values may take on the role of the survivor, but that is reasonable only if all of its children lose their importance.
- It is not unusual for business to demand keeping the outdated values in the data stores for quite long. Bringing change propagation to a conclusion can require a lot of time.

2.4.4 Problem

Which survivor do you choose to replace outdated values, how do you time your change and how do you map outdated values?

2.4.5 Solution

The timing of integrating categories is a matter of applied statistics. You should make sure that the observed dip in importance can be considered permanent. Choose a value that has a close resemblance to the outdated value(s), but don’t use “Miscellaneous.” If you can, choose the parent value.

Before date of effectiveness, remove the value(s) to be phased out from the attribute in the reference data store, change the lifecycle status of these values to “inactive” or “superceded,” add a mapping from old values to the survivor at 100% in the mapping data store, valid as of date of effectiveness.

* MOLAP = Multidimensional Online Analytical Processing

2.4.6 Implementation

Based on the data model defined in Section 2.1.3, the following DML is required to implement this pattern:

```
-- assume a new version of the
-- attribute 2344 has been created
-- remove outdated values 324 and
-- and 7301 from attribute 2344
DELETE FROM REF.ATTRIBUTE_MEMBERS
WHERE
    ATTR.VALUE_ID IN (7301, 324)
    AND
    ATTR.ATTRIBUTE_ID = 2344
    AND
    ATTR.ATTRIBUTE_VERSION = (
        SELECT
            MAX(ATTRIBUTE_VERSION)
        FROM
            REF.ATTRIBUTE_VERSIONS
        WHERE
            ATTRIBUTE_ID = 2344
    );
-- terminate lifecycle of values 324
-- and 7301 in attribute 2344
UPDATE REF.VALUE_LIFECYCLE_STATUS SET
    STATUS = 'SUPERCEDED'
WHERE
    VALUE_ID IN (7301, 324)
    AND
    ATTRIBUTE_ID = 2344;
-- create new mapping 44332
INSERT INTO MAP.VALUE_MAPPINGS (
    MAPPING_ID,
    VALID_FROM,
    VALID_UNTIL
) VALUES (
    44332,
    '2001-01-01',
    '2008-12-31'
);
-- map outdated values 324 and
-- and 7301 to existing value 10032
INSERT INTO MAP.VALUE_SUCCESSORS (
    MAPPING_ID,
    VALUE_ID,
    SUCCESSOR_ID,
    FACTOR
) VALUES (
    44332, 324, 10032, 1.0E00,
    44332, 7301, 10032, 1.0E00
);
```

2.4.7 Consequences

- Data stores can detect that they should treat the old value differently at their interfaces.
- (Derived) measures associated with the old values must be recomputed.
- Computationally this is less expensive than “Integrate Category into New,” because only parts of the recorded facts must be re-categorized, but not the survivor.
- It is no longer intended to record business transactions for the old values, but for the survivor.

2.4.8 Related Patterns

- If there is no natural survivor for the outdated values, consider using “Integrate Category into New”.

2.4.9 Examples

When the “Federal Republic of Germany” and the “German Democratic Republic” (re-)united in 1990, the former took over the legal and financial rights and liabilities of the latter. One continued to exist as a political entity, while the other vanished. Hence, from the perspective of the attribute “Country,” business conducted in

the “German Democratic Republic” was integrated with existing business conducted in the “Federal Republic of Germany”.

2.5 Integrate Category into New

Merge the business facts associated with two or more values into a completely new one.

2.5.1 Motivation

When certain categories describing the nature of your business become outdated, it may happen that a survivor candidate from out of them cannot be identified. The question may now be raised what to do with the business facts stored in your data stores.

You may want to re-categorize them, in which case a new category must be introduced. The history of business associated with outdated categories is cut off and the dimensional values describing facts mapped to a new one. This is common when you intend to conduct time-series analysis that presents old categories as new ones in the amended view. For that to make sense, the new category must have a close resemblance to the old it bundles together. However, it may happen that there is no point in re-categorization. For example, imagine a car company selling a particular type of car. In July 2003, VW stopped production of the “Beetle.” From an analytical perspective it would not make a lot of sense to compare the historic performance of sales of “Beetle” cars with current sales of, say the “Golf” line of vehicles. In such a situation the facts associated with the “Beetle” would remain to be so, no mapping would take place.

2.5.2 Context

- You are using “Reference Data Store,” “Mapping Data Store,” and “Value Lifecycle.”
- One or more particular categories have ceased to be relevant from a business point of view.
- None of the outdated categories might act as survivor.
- The new category is closely related to the outdated ones, i.e. most people will consider it to be a reasonable substitute.
- You intend to re-categorize the outdated categories for the purpose of time-series analysis.

2.5.3 Forces

- Sometimes you will find yourself combining the names of all old values, concatenated, into one, effectively making the new business category the union of all previously existing categories. However, these concepts may seem artificial to business people and not help them understand their data any better.
- A loss of business relevance may be temporary. You should wait a certain time to ensure a category is truly outdated. This, on the other hand, may worsen the distorting effects of re-categorization.
- If you integrate (one or more) categories with the scavenger category “Miscellaneous,” you don’t have to bother with the question whether the dip in importance is permanent, and that category may grow rather large over time, causing problems of its own. However, for now you’ll be fine.

2.5.4 Problem

What is a good choice of successor to replace the outdated values and how do you map them?

2.5.5 Solution

Identify a business concept that circumscribes all the concepts that have become outdated. If needed, use the concatenation rule, but not for more than three categories. If none of that helps, use “Miscellaneous.”

Before date of effectiveness, remove the value(s) to be phased out from the attribute in the reference data store, change the lifecycle status of these values to “inactive” or “superseded,” add a mapping from old values to the survivor at 100% in the mapping data store with an optional linear factor, valid as of date of effectiveness.

2.5.6 Implementation

Based on the data model defined in Section 2.1.3, the following DML is required to implement this pattern:

```
-- assume a new version of the
-- attribute 2344 has been created
-- remove outdated values 324 and
-- and 7301 from attribute 2344
```

```

DELETE FROM REF.ATTRIBUTE_MEMBERS
WHERE
    ATTR.VALUE_ID IN (7301, 324)
    AND
    ATTR.ATTRIBUTE_ID = 2344
    AND
    ATTR.ATTRIBUTE_VERSION = (
        SELECT
            MAX(ATTRIBUTE_VERSION)
        FROM
            REF.ATTRIBUTE_VERSIONS
        WHERE
            ATTRIBUTE_ID = 2344
    );
-- terminate lifecycle of values 324
-- and 7301 in attribute 2344
UPDATE REF.VALUE_LIFECYCLE_STATUS SET
    STATUS = 'SUPERCEDED'
WHERE
    VALUE_ID IN (7301, 324)
    AND
    ATTRIBUTE_ID = 2344;
-- create new mapping 44332
INSERT INTO MAP.VALUE_MAPPINGS (
    MAPPING_ID,
    VALID_FROM,
    VALID_UNTIL
) VALUES (
    44332,
    '2001-01-01',
    '2008-12-31'
);
-- add new value 15680 to
-- attribute 2344
INSERT INTO REF.ATTRIBUTE_MEMBERS (
    VALUE_ID,
    ATTRIBUTE_ID,
    ATTRIBUTE_VERSION
)
SELECT
    15680,
    ATTR.ATTRIBUTE_ID,
    MAX(ATTR.ATTRIBUTE_VERSION)
FROM
    REF.ATTRIBUTE_VERSIONS SV
WHERE
    ATTR.ATTRIBUTE_ID = 2344;
GROUP BY
    ATTR.ATTRIBUTE_ID;
-- start lifecycle of value
-- 15680 in attribute 2344
INSERT INTO REF.VALUE_LIFECYCLE_STATUS (
    VALUE_ID,
    ATTRIBUTE_ID,
    STATUS
) VALUES (
    15680,
    2344,
    'ACTIVE'
);
-- map outdated values 324 and
-- and 7301 to the new value
INSERT INTO MAP.VALUE_SUCCESSORS (
    MAPPING_ID,
    VALUE_ID,
    SUCCESSOR_ID,
    FACTOR
) VALUES (
    44332, 324, 15680, 1.0E00,
    44332, 7301, 15680, 1.0E00
);

```

2.5.7 Consequences

- Depending on the speed of adoption in the information architecture, reported performance indicators may differ between different (analytical) data stores.
- Computationally this is more expensive than “Invalidate Category,” because all recorded facts must be re-categorized, because there is no survivor.
- (Derived) measures associated with the old values must be recomputed.

2.5.8 Examples

In insurance, so-called „run-off“ business (coverage—or portfolios thereof—underwritten earlier, but no longer offered to clients) is often detached from normal operations and captured in special vehicles that manage them to optimize cost structures. Run-off business was typically managed earlier in particular parts of the company and then pooled in a different part—or even different legal entity. There you will find disparate kinds of coverage that—from a content point of view—have little in common.

The introduction of the “Euro” on 1 January 1999 eventually eliminated a host of other national “Currencies” like the “French Franc,” “Belgian Franc,” “Deutsche Mark,” or “Dutch Guilder.” For each of these a different conversion rate was set by the European Central Bank, e.g.

Belgian Franc	40.3399
Deutsche Mark	1.95583
French Franc	6.55957
Dutch Guilder	2.20371

The linear factor in here would be the conversion rate, the date of effectiveness 1 January 1999. The change became manifest by means of a SWIFT broadcast on 31 December 1998.

The 2002 NAICS [11] industry type 52213 (“Credit Unions”) was created from the two 1987 SIC industry types 6061 (“Credit Unions, Federally Chartered”) and 6062 (“Credit Unions, Not Federally Chartered”).

2.6 Partially Split Category

Refine the business perspective on a value while keeping it alive.

2.6.1 Motivation

You may find time and again that some part of your business is particularly lucrative or attracts a higher than average transaction volume. In a statistical analysis, transactions would be associated with one particular describing dimensional value more often than others. Your reaction as a business person would be to try to understand what is going on so you can increase profits. One such possibility is that different customer segments purchase your product and that one such segment is interested in a slight variation of the product that appeals more to its needs.

In this case you would want to start offering an alternative product to the identified segment that differs from the existing along one characteristic dimension. The rest of your customer base would continue to be offered the existing product. Hence, the dimension describing the distinguishing product characteristic would experience a split of an existing category into two. As is the case with “Introduce Category,” you have to decide what to do with historic business transactions, i.e. whether to recategorize them.

One further question is to decide whether you really should diversify along (only) one characteristic product dimension. For once, you must make sure that the survivor value retains at least some level of importance. Maybe all of your customers would want to buy a product customized to their needs, effectively eliminating the need for the existing category. Or, which is something even more difficult to handle, the product characteristics appealing to specific customer segments vary along more than one property, which would entail changes to more than one dimension.

2.6.2 Context

- You are using “Reference Data Store,” “Mapping Data Store,” and “Value Lifecycle.”
- Your business transactions are described by at least two characteristic dimensional attributes.
- You want to be able to describe transactions at a more fine-grained level along one such attribute.
- One value in this attribute closely resembles the new value(s) to be introduced, but is not quite equal.
- The new values have never been an element of the attribute.

- The old value continues to be important to business.
- It has been ruled out that refinement of other attributes is required.

2.6.3 Forces

- Even in a setting where more than one describing attribute changes, you may want to use “Partially Split Category,” alas in phases. First, it is sometimes a matter of speculation which value to split and which not. Hence, you may want to try it out one after the other. Second, the data mappings required to re-categorize historic data along a couple of dimensions may become too costly or outright intractable. In this case, perform splits along each dimension, one by one, treating it as independent of the others.
- Depending on how late you introduce the new values, transactions conducted previously may not be appropriately categorized. They will either have to be re-categorized, or reported measures will not give a fully accurate picture of history.
- With values arranged hierarchically (MOLAP), the survivor can take on the role of parent of the new values with 0% of business facts associated. It is then likely to be re-named and one of its children taking on the parent’s old name.

2.6.4 Problem

How do you choose the new values and how do you map existing data?

2.6.5 Solution

If the name of the existing category is a concatenated form, try using parts of its constituent names for the concepts behind the new categories. If that doesn’t work, try using the existing category as parent of the new ones (i.e., children), in which case you are guided in the choice of their name and meaning.

Before date of effectiveness, add the new values to the attribute in the reference data store, lifecycle status “active,” add a mapping from the old value to the new values (with the appropriate weight, summing up to less than 100%) in the mapping data store, valid as of date of effectiveness.

2.6.6 Implementation

Based on the data model defined in Section 2.1.3, the following DML is required to implement this pattern:

```
-- assume a new version of the
-- attribute 2344 has been created
-- add new values 3498 and 665
-- to attribute 2344
INSERT INTO REF.ATTRIBUTE_MEMBERS (
    VALUE_ID,
    ATTRIBUTE_ID,
    ATTRIBUTE_VERSION
)
SELECT
    3498,
    ATTR.ATTRIBUTE_ID,
    MAX(ATTR.ATTRIBUTE_VERSION)
FROM
    REF.ATTRIBUTE_VERSIONS SV
WHERE
    ATTR.ATTRIBUTE_ID = 2344;
GROUP BY
    ATTR.ATTRIBUTE_ID;
INSERT INTO REF.ATTRIBUTE_MEMBERS (
    VALUE_ID,
    ATTRIBUTE_ID,
    ATTRIBUTE_VERSION
)
SELECT
    665,
    ATTR.ATTRIBUTE_ID,
    MAX(ATTR.ATTRIBUTE_VERSION)
FROM
    REF.ATTRIBUTE_VERSIONS SV
WHERE
    ATTR.ATTRIBUTE_ID = 2344;
GROUP BY
    ATTR.ATTRIBUTE_ID;
```

```

-- start lifecycle of values 3498
-- and 665 in attribute 2344
INSERT INTO REF.VALUE_LIFECYCLE_STATUS (
    VALUE_ID,
    ATTRIBUTE_ID,
    STATUS
) VALUES (
    3498, 2344, 'ACTIVE',
    665, 2344, 'ACTIVE'
);

-- create new mapping 44332
INSERT INTO MAP.VALUE_MAPPINGS (
    MAPPING_ID,
    VALID_FROM,
    VALID_UNTIL
) VALUES (
    44332,
    '2001-01-01',
    '2008-12-31'
);

-- split up existing value 10032 and
-- assign 64% of business facts to
-- value 3498 and 21% to value 665;
-- 15% remain with the existing
INSERT INTO MAP.VALUE_SUCCESSORS (
    MAPPING_ID,
    VALUE_ID,
    SUCCESSOR_ID,
    FACTOR
) VALUES (
    44332, 10032, 3498, 0.64E00,
    44332, 10032, 665, 0.21E00
);

```

2.6.7 Consequences

- The new values can be used to record business transactions.
- (Derived) measures associated with the survivor value and its split-off siblings must be (re-)computed.

2.6.8 Related Patterns

- If the split (old) value retains 0% of associated business facts and values are not arranged hierarchically, consider using “Fully Split Category”.

2.6.9 Examples

The context of this pattern often occurs when organizations are rearranged. For example, a department may have changed its purpose and one team in it should rather be part of a different (new) department. It is then moved over, and the headcount of both the original and the new department changes.

For a while, our company tracked the line of business „Aviation & Space“ (covering operation of, e.g. satellites, commercial airplanes, or spaceships) without further distinction. Business volume attained a level that we wanted to understand it at a more fine-grained level, resulting in two new lines, both children of the existing line, namely “Aviation” and “Space.” All previous transactions were split up between the two new so that “Aviation & Space” did not have any business facts associated with it, anymore.

2.7 Fully Split Category

Refine the business perspective on a value, replacing it by others.

2.7.1 Motivation

As opposed to situations where “Partially Split Category” might be more applicable, you may run into a situation where the old (outdated) value has lost all its significance for analytical purposes. Here the main question concerns the choice of successors.

When a value becomes obsolete, you may be able to identify other (new) values that seem appropriate, but do not represent all business transactions to be re-categorized. You may want to continue your search, but that might merely lead to a different composition with a few transactions still not re-categorized. You may be

tempted to use a placeholder like “Miscellaneous” or “Other” to cover these cases, but this effectively anonymizes them; their business meaning will become unspecific.

It may help to take a look at the name of the business concept represented by the value. This name gives hints as to what concepts it subsumes, especially when it is of the form “One Thing & Another Thing.” When this concept ceases to exist, the constituents “One Thing” and “Another Thing” may continue to have business significance. These are the obvious choice for replacing the outdated value.

2.7.2 Context

- You are using “Reference Data Store,” “Mapping Data Store,” and “Value Lifecycle.”
- Your business transactions are described by at least two characteristic dimensional attributes.
- You want to be able to describe transactions at a more fine-grained level along one such attribute.
- One value in this attribute closely resembles the new value(s) to be introduced, but is not quite equal.
- The new values have never been an element of the attribute.
- The old value is outdated and will be eliminated.
- It has been ruled out that refinement of other attributes is required.

2.7.3 Forces

- With values arranged hierarchically (MOLAP), you will typically want to partially split up an existing value, but retain it with 0% of business facts associated.
- Sometimes you may find that you ended up with a number of new values replacing the existing one, but a (small) number of transactions cannot be properly re-categorized, effectively requiring you to add another new value named “Miscellaneous”.
- Quite often you will want to choose new values whose names, concatenated, form the name of the existing value.

2.7.4 Problem

How do you choose the new values and how do you map existing data?

2.7.5 Solution

If the name of the existing category is a concatenated form, try using its constituent names for the concepts behind the new categories. If that doesn’t work, try it with less constituents and assign the remainder to “Miscellaneous.”

Before date of effectiveness, add the new values to the attribute in the reference data store, lifecycle status “active,” add a mapping from the old value to the new values (with the appropriate weight, summing up to 100%) in the mapping data store, valid as of date of effectiveness. Furthermore, valid as of the same date, remove the old value from the attribute and mark it as “superceded.”

2.7.6 Implementation

Based on the data model defined in Section 2.1.3, the following DML is required to implement this pattern:

```
-- assume a new version of the
-- attribute 2344 has been created
-- add new values 3498 and 665
-- to attribute 2344
INSERT INTO REF.ATTRIBUTE_MEMBERS (
    VALUE_ID,
    ATTRIBUTE_ID,
    ATTRIBUTE_VERSION
)
SELECT
    3498,
    ATTR.ATTRIBUTE_ID,
    MAX(ATTR.ATTRIBUTE_VERSION)
FROM
    REF.ATTRIBUTE_VERSIONS SV
WHERE
    ATTR.ATTRIBUTE_ID = 2344;
GROUP BY
    ATTR.ATTRIBUTE_ID;
```

```

INSERT INTO REF.ATTRIBUTE_MEMBERS (
    VALUE_ID,
    ATTRIBUTE_ID,
    ATTRIBUTE_VERSION
)
SELECT
    665,
    ATTR.ATTRIBUTE_ID,
    MAX(ATTR.ATTRIBUTE_VERSION)
FROM
    REF.ATTRIBUTE_VERSIONS SV
WHERE
    ATTR.ATTRIBUTE_ID = 2344;
GROUP BY
    ATTR.ATTRIBUTE_ID;
-- start lifecycle of values 3498
-- and 665 in attribute 2344
INSERT INTO REF.VALUE_LIFECYCLE_STATUS (
    VALUE_ID,
    ATTRIBUTE_ID,
    STATUS
) VALUES (
    3498, 2344, 'ACTIVE',
    665, 2344, 'ACTIVE'
);
-- remove existing value 10032
DELETE FROM REF.ATTRIBUTE_MEMBERS
WHERE
    ATTR.VALUE_ID = 10032
    AND
    ATTR.ATTRIBUTE_ID = 2344
    AND
    ATTR.ATTRIBUTE_VERSION = (
        SELECT
            MAX(ATTRIBUTE_VERSION)
        FROM
            REF.ATTRIBUTE_VERSIONS
        WHERE
            ATTRIBUTE_ID = 2344
    );
-- terminate lifecycle of values 324
-- and 7301 in attribute 2344
UPDATE REF.VALUE_LIFECYCLE_STATUS SET
    STATUS = 'SUPERCEDED'
WHERE
    VALUE_ID = 10032
    AND
    ATTRIBUTE_ID = 2344;
-- create new mapping 44332
INSERT INTO MAP.VALUE_MAPPINGS (
    MAPPING_ID,
    VALID_FROM,
    VALID_UNTIL
) VALUES (
    44332,
    '2001-01-01',
    '2008-12-31'
);
-- split up existing value 10032 and
-- assign 64% of business facts to
-- value 3498 and 36% to value 665
INSERT INTO MAP.VALUE_SUCCESSORS (
    MAPPING_ID,
    VALUE_ID,
    SUCCESSOR_ID,
    FACTOR
) VALUES (
    44332, 10032, 3498, 0.64E00,
    44332, 10032, 665, 0.36E00
);

```

2.7.7 Consequences

- The new values can be used to record business transactions.
- It is no longer possible to record business transactions for the old value.
- (Derived) measures associated with split-off values must be computed.

2.7.8 Examples

The split of “Czechoslovakia” into the “Czech Republic” and “Slovakia” is a typical case in point. The old “Country” ceased to exist, and legally the two new ones took over.

Appendix A Supporting Changes to the Model of the Business

This section outlines the patterns supporting “Patterns for Coping with Changes to a Single Attribute.” We intend to complete them at a later stage. While the above patterns are concerned with how one can look at and handle changes to the business model, the patterns listed here discuss—using Zachman’s terminology [13]—their translation into changes to the system model.

Coming up with a “Ubiquitous Language” [4] has well-known benefits, such as the ability of engineers and business to communicate effectively during the design of an application. Yet, we are more interested in how to *decouple* both parties—only during the application maintenance phase. This is achieved by four patterns, namely

- “Reference Data Store”
- “Mapping Data Store”
- “Automatic Business to System Model Transformation”
- “Value Lifecycle”

The “Reference Data Store” stores the history of dimensions and the changes to them, while the “Mapping Data Store” stores mappings between values in different such versions. “Value Lifecycle” helps you steer, at a fine-granular level, the phase-out of a value and thus the behavior of data stores at (inbound) interfaces for the purpose of smoother change adoption. “Automatic Business to System Model Transformation” keeps business and system model of the world, as it is stored in or obtained from reference data stores, synchronized.

A.1 Reference Data Store

A “Reference Data Store” is a time-oriented “Repository” [3, 4] of the history of attributes, their values, as well as their “Value Lifecycle”. Its contents are administered by business—based on the business model of the world. Applications read the contents—based on the system model of the world. The “Reference Data Store” ensures that an “Automatic Business to System Model Transformation” takes place each time the business model changes.

A “Reference Data Store” enables applications to abstract from changes to dimensional attributes in that it provides a stable element—the dimensional attribute—to both business and engineers as a concept to build their model of the world on. The handling of volatile elements—dimensional structure, values—is divided thus

- Business maps changes in the real world to changes in attributes which are part of the business model and administers these changes in the “Reference Data Store” accordingly
- Applications are designed based on the attributes and regularly adopt structural changes obtained from the “Reference Data Store,” which includes mapping values and updating (derived) measures

As such, a “Reference Data Store” decouples business and engineers during the maintenance phase of an application by tying them together via the attribute (type).

A.2 Mapping Data Store

A “Mapping Data Store” is a time-oriented data store of the mapping between attribute values across time or versions, respectively. It separates responsibilities between historization of attribute structures (which is in the realm of the “Reference Data Store”) and mapping of attribute values. Architecturally, the “Mapping Data Store” always depends on the “Reference Data Store.”

Mappings can take all sorts of shapes and sizes. To keep things simple, yet not lose too much scope of applicability, linear functions are used to express mappings. They address an important subset of mapping problems, but your mileage may vary.

The “Mapping Data Store” stores the (successor) mappings of values for each measure their attribute segments. Mappings are provided between two points in time; the mapping data store must ensure that all values involved

in a mapping are “Active” or “Inactive” where required, never “Superseded”. It provides an interface that exposes mappings to other data stores based on the validities and identities of the underlying “Reference Data Store”.

The main task the “Mapping Data Store” leaves applications with is when and how to adopt changes. Potential consistency issues must be managed as part of change propagation.

A.3 Automatic Business to System Model Transformation

The “Ubiquitous Language” is used to express changes to the business model, but that is only one part of change adoption; next comes their translation into changes to the system model. When business concepts find their way into the system model (e.g., business rules), they effectively bind a system concept to the existence of a business concept. This prevents business from adopting certain types of changes in an undue fashion, practically requiring the continued presence of technical staff during the operational phase.

“Automatic Business to System Model Transformation” decouples business and technical people during the operational phase, while keeping them in touch during the design phase. It liaises dimensions in the business model with domains in the system model and defines a simple mapping between them for use in forward engineering. The “Reference Data Store” uses this mapping.

As opposed to Eric Evans’ notion of a “Layered Architecture” [4], we do not require all business concepts be reified as part of the system model. As long as behavior is not tied to specific attribute values, the system model only depends on the attributes.

A.4 Value Lifecycle

Once you decide that a business term is no longer associated with the business concept it used to refer to (or if the concept itself has lost any significance), you must phase it out. This means that it must eventually be eliminated from data stores. However, completely banning its use throughout your organization from a point in time on (“all or nothing”) incurs a certain risk.

“Value Lifecycle” phases out an attribute’s value in stages once it experiences a material change in meaning. It ensures smooth transition from outdated terms to their successors. Depending on your data analysis needs, you may want to associate a single value with several attributes, which effectively creates several lifecycles (one per attribute). The “Value Lifecycle” is stored in the “Reference Data Store,” which should at least comprise the set {“Active”, “Inactive”, “Superseded”}. Once a value becomes “Inactive,” it must become “Superseded” within a timeframe set by the attribute owner. It requires data stores to validate data at inbound interfaces. “Active” values are always accepted, “Superseded” always rejected. “Inactive” values may be accepted or not, depending on business and technical needs.

Appendix B References

1. Association for Computing Machinery: Computing Classification System. Downloaded from <http://www.acm.org/class/> on 24 March 2005
2. Dirk Bäumer, Dirk Riehle, Wolf Siberski, Carola Lilienthal, Daniel Megert, Karl-Heinz Sylla, and Heinz Züllighoven. Values in Object Systems. Ubilab Technical Report 98.10.1. Zurich, Switzerland, UBS AG, 1998
3. Stephen Berczuk, Brad Appleton: Software Configuration Management Patterns. Addison-Wesley, 2003
4. Eric Evans: Domain-Driven Design. Addison-Wesley, 2004
5. Heise Online: Massive Zunahme von Werbe-Spam für gefälschte Luxusartikel (22 March 2005, in German). Downloaded from <http://www.heise.de/newsticker/meldung/57817> on 27 April 2005.
6. Martin Fowler: Patterns for Things that Change with Time. Downloaded from <http://www.martinfowler.com/ap2/> on 24 March 2005
7. Ralph Kimball, Margy Ross: The Data Warehouse Toolkit, 2nd Edition: The Complete Guide to Dimensional Modeling. John Wiley, 2002.
8. Steve Peterson: Stars: A Pattern Language for Query-Optimized Schemas. Proc Pattern Languages of Programs (PLOP) '94. (Also: Downloaded from <http://c2.com/ppr/stars.html> on 7 June 2005.)
9. Richard Snodgrass: Developing Time-Oriented Database Applications in SQL. Morgan-Kaufmann, 1999

10. Sophos, Plc.: Press Release “Sophos PureMessage introduces new spam classification support” (12 March 2004). Downloaded from <http://www.sophos.com/> on 27 April 2005
11. United States Census Bureau: North American Industry Classification System, Revisions for 2002. Downloaded from <http://www.census.gov/epcd/naics02/> on 27 April 2005
12. Wikipedia: Time-Series. Downloaded from <http://en.wikipedia.org/wiki/Time-series/> on 10 June 2005
13. John A. Zachman: A Framework for Information Systems Architecture. IBM Systems Journal 26(3), 1987

Appendix C Acknowledgements

Thanks go to our shepherd Dirk Riehle for his comments on this pattern language as well as his encouragement to take on the issue of time and history in business systems. Thanks go also to Boris Bokowski and Dora Zosso for their comments on earlier versions, specifically illustrative examples and document structure.