Patterns in Data Quality

A Method for Organizing Enterprise Data Quality (Web) Services in Service Oriented Architectures

Navin Sharma • Director of Global Product Strategy, Data Quality, Pitney Bowes Business Insight
with Michael C. Overturf
Patterns in Data Quality
A Method for Organizing Enterprise Data Quality (Web) Services in Service Oriented Architectures

INTRODUCTION

DIFFERENT INDUSTRIES HAVE VARYING BUSINESS PROCESSES TO SATISFY THEIR CUSTOMERS AND WHILE THE DATA UNIVERSE OFTEN DIFFERS, PATTERNS CAN BE DEVELOPED TO IMPROVE DATA QUALITY. IMPROVED DATA QUALITY NOT ONLY INCREASES PRODUCTIVITY, IT ALSO ENHANCES THE VALUE AND UTILITY OF BUSINESS INFORMATION FOR OPERATIONAL AND ANALYTICAL PURPOSES.
IMPROVED DATA QUALITY ENHANCES THE VALUE AND UTILITY OF BUSINESS INFORMATION FOR OPERATIONAL AND ANALYTICAL PURPOSES

Data Quality Recognized

“When you cannot measure it, when you cannot express it in numbers, your knowledge is of a meager and unsatisfactory kind.” With this statement, made in 1883, it is entirely possible that Lord Kelvin – Sir William Thomson – was the first to observe the importance of data quality. His quote was preceded by a rich career in science and engineering, where he, among other things, developed models for the data bandwidth of undersea telegraphy cables. He was at least the first to define constraint models on the business value of data quality.

Most data or system architects understand the need for good data quality, and the deleterious effects of its absence. In our experience most enterprise architects leave the quality of data up to database administrators, with the perfectly reasonable assumption that if you own the data, you’re responsible for its quality.

But is this a sensible approach? Are the restrictions with which a database administrator is forced to work such that it is practically impossible to reasonably warrant the quality of the data?

Our assertion is: no reasonable warranty can be given without organization and measurement. As proof we can easily show that in a nodal network (such as a group of independently managed databases), differing interpretations of the same signal will cause the system to break down - as Lord Kelvin argued on the reliability of lighthouses in 1873. Moreover, duplicating normalization files and duplicating rule development lead to unnecessary costs.

Data quality services should therefore be a common architectural resource, with consistent and repeatable data interpretation and measurement. These services should be easy to understand, simple to use and deployable in minutes. All of this can be achieved through a well considered organization of data quality services.

Managing Complexity

Setting up enterprise-wide services is a daunting task. Architects will end up with hundreds, perhaps thousands of different services, many interrelated, and hierarchical. Part of this web services universe is data quality services. How best does one organize these so that application developers can find, understand, re-use and measure them with intended effect?

In attempting to understand design principles for enterprise web services in general, and data quality web services specifically, we encountered work done by mathematician and architect Christopher Alexander. In 1964 he proposed a method for deriving architecture by examining and structuring the universe of constraints imposed upon it. An entire school of thought has since been developed around this principle: examine the constraints of a design challenge, understand the constraint relationships, and the design almost creates itself. In particular, he suggested enumerating and describing constraints and their resulting solutions in detail, such that solutions to small constraint ’pieces’ could be re-used. These ’pieces’ could be rearranged in different ways, thereby creating a pattern of good design principles. Since human beings are good at understanding and recognizing patterns, design work could now be made much more intuitive and accessible to more people.

The word ‘architecture’ figures prominently in Service-Oriented Architectures, and we suggest that the same principles used for designing good buildings also apply to designing good web services. Alexander’s work allows us to understand data quality services in terms of a ‘language of patterns.’ If a database administrator or programmer faces
a data quality constraint, now he or she can select the pattern that best fits the need and deploy it quickly and effectively.

In the remainder of this section we try show how some data quality services are derived from patterns, how they are orchestrated, and then show how templates and patterns are used. We did not attempt to derive a full language syntax, leaving this to a future effort, but rather we are simply showing the organizing effects of its taxonomy.

Patterns

We show some example data quality patterns in tabular form. A reasonably sized enterprise implementation will feature hundreds of data quality patterns, so we only list a few here. Our intention is to offer a near-complete set of data quality patterns that meet the vast majority of enterprise needs.

Patterns are defined in terms of number, name, description, flow, and constraint. The flow is the representation of the Enterprise Designer orchestration layer of Pitney Bowes® Spectrum™ Technology Platform. The constraints are typically data or modal conditions. Data quality patterns are implemented as web services by exposing the web service with an automatically generated Web Service Definition Language (WSDL) entry for public registry with a single click.

Furthermore, we implement the constraint hierarchy using atomic and composite services. An atomic service is one that performs a discrete, singular function; a composite service is one that aggregates atomic patterns to meet some required constraint.

The following example patterns are organized into consistency, uniqueness, and accuracy patterns mainly for illustrative purposes.

Consistency

Consistency patterns ensure that data items are encoded consistently with reference to some standard. We call this data normalization: a data item is normalized with respect to a file that contains all permissible forms of that data. Examples of this are social security numbers, member numbers, addresses, legal names, etc.

The example in Pattern C1-C4 illustrates just a few atomic data quality services. Any organization will likely have thousands. But for clarity, this examples shows how a series of atomic data quality services can build in complexity. A complete set of patterns of use in a typical commercial setting would consist of hundreds of such patterns.

Uniqueness

Uniqueness is the attribute that defines singularity to a data item: it implies the lack of ambiguity in the interpretation of a data item. For example, a unique part number is not shared across multiple parts trees as there is one and only one such part number. A data item can be consistent, but yet not unique. The following example patterns ensure a uniqueness attribute. (See page 6.)

Accuracy

Accuracy describes the alignment between metadata and data. Accuracy ensures that a data item is in fact correct: that a certain person actually lives at an address, that a part number does in fact represent that part, etc. The following examples show some atomic services that ensure the accuracy of various pieces of data. (See page 6.)
DATA QUALITY SERVICES SHOULD BE A COMMON ARCHITECTURAL RESOURCE WITH CONSISTENT AND REPEATABLE DATA INTERPRETATION AND MEASUREMENT

<table>
<thead>
<tr>
<th>#</th>
<th>NAME</th>
<th>DESCRIPTION</th>
<th>CONSTRAINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.1</td>
<td>Normalize Corporate Name</td>
<td>Given name string input, normalize according to default rules</td>
<td>Single mapped name field input, normalized name field output</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Firm-Name-in</td>
<td>Normalize</td>
</tr>
<tr>
<td>C.2</td>
<td>Normalize Address</td>
<td>Normalize a postal address</td>
<td>Single mapped address field input, normalized address field output</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Address-in</td>
<td>Normalize</td>
</tr>
<tr>
<td>C.3</td>
<td>Normalize Address – 2i</td>
<td>Normalize a postal address from two asynchronous inputs</td>
<td>Two mapped address fields input, normalized address field output</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Address-in-1</td>
<td>Stream Combiner</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Address-in-2</td>
<td>Normalize</td>
</tr>
<tr>
<td>C.4</td>
<td>Normalize Address – 3i</td>
<td>Normalize a postal address from three asynchronous inputs</td>
<td>Three mapped address fields input, normalized address field output</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Address-in-1</td>
<td>Stream Combiner</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Address-in-2</td>
<td>Normalize</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Address-in-3</td>
<td></td>
</tr>
</tbody>
</table>
Patterns in Data Quality

A Method for Organizing Enterprise Data Quality (Web) Services in Service Oriented Architectures

### Example Patterns: UNIQUENESS

<table>
<thead>
<tr>
<th>#</th>
<th>NAME</th>
<th>DESCRIPTION</th>
<th>CONSTRAINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>U.1</td>
<td>Simple customer identifier</td>
<td>Read pre-defined customer record, identify whether customer exists, if so, list as duplicate</td>
<td>Identify unique customer records, list duplicates</td>
</tr>
</tbody>
</table>

### Example Patterns: ACCURACY

<table>
<thead>
<tr>
<th></th>
<th>NAME</th>
<th>DESCRIPTION</th>
<th>CONSTRAINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.1</td>
<td>Validate Address</td>
<td>Determine whether the address is a valid address for postal delivery to a named person</td>
<td>Input proposed address, return correct address</td>
</tr>
<tr>
<td>A.2</td>
<td>Validate Telephone</td>
<td>Determine whether a telephone number is a functioning telephone number</td>
<td>Input telephone number, return functioning telephone number</td>
</tr>
<tr>
<td>A.3</td>
<td>Validate Address Location</td>
<td>Determine valid latitude/longitude and geocode for the front door of an address</td>
<td>Provide address, return ?</td>
</tr>
</tbody>
</table>
WITH A PATTERN-BASED APPROACH TO WEB SERVICES ONE CAN MEASURE THE QUALITY OF THE UNDERLYING DATA STREAMS

Composite Patterns
A composite service is a non-atomic service that is defined in an orchestration.

Figure 1 shows an example pattern, exposed as a composite service, that encapsulates the various atomic level services to normalizes record attributes of a customer. It does so by referring to atomic services that normalize the name (C1), normalizes the address (C2), validates the address against referential source (A1), geo-codes the address against a referential source for pinpoint accuracy (A3) and validates the telephone number (A3), running these as parallel threads by attribute type and provide a complete, consistent and accurate customer information as the final output.

Figure 2 depicts a pattern that validates various attributes of a customer record, in this case, calling a distinct atomic service that normalizes the name (C1), normalizes the address (C2), validates the address (A1) and matches against existing customer repository to establish uniqueness (U1).

![Figure 1 – Customer Record Maintenance Pattern](image1)

![Figure 2 – Customer Validation Pattern](image2)
Patterns in Data Quality

A Method for Organizing Enterprise Data Quality (Web) Services in Service Oriented Architectures

Patterns and Templates

How does one apply these principles towards real world applications? We suggest organizing data quality rules in terms of atomic and composite flows as shown in Figure 3, which is consistent with the pattern language described by Christopher Alexander. If atomic patterns are the verb-phrase/noun phrase sentences, composite patterns are paragraphs, and templates are narratives, with meaningful semantics for data administrators. Distribution of pattern-based data quality services in the form of templates achieves the organizational objectives described above.

Templates can be loaded and modified using the Pitney Bowes Spectrum Technology Platform Enterprise Designer as the primary orchestration layer. A template can be directly exposed as a web service or API-ready service for enterprise use. This gives database administrators and application developers the freedom to choose and apply a re-usable web service and measures specific data quality standards. This discipline provides a compliance vehicle for enforcing corporate data governance.

Fabric

Different industries have varying business processes to satisfy their customers. While the data universe often differs, the different business process requirements change the way data is used and governed from one industry to the other. We define industry-specific sets of data quality templates as industry fabric.

For example, a key productivity and risk reduction enabler for insurance underwriting departments is the straight-through processing concept seen in Figure 4. This improves quote-to-bind ratios, automates manual underwriting tasks, and ensures completeness and accuracy of customer information.

Figure 5 shows a typical policy application and quotation process. When we examine the specific transaction where an application is submitted to the underwriting staff (for homeowner’s insurance), the following information is usually provided in the application: Name and SSN of owner, address of home, type, telephone number, etc.

The following template (Figure 6) is part of the insurance fabric. It performs basic identity normalization, followed by a lookup of the customer record in the underwriting system. If the application is from an existing customer, the application information is appended to the existing record and a transaction risk score is calculated. If the application is from a new customer, the template checks for compliance with the Office of Foreign Asset Control (OFAC), then determines whether the property resides in a flood plain, before computing the risk score.

Data quality fabrics allow database administrators and application developers to quickly apply business process best practice composite data quality services by providing those narratives relevant to their industry.
DATA INSPECTION INSTILLS CONFIDENCE IN THE USE OF TEMPLATES AND EXPECTED OUTPUT CAN BE VALIDATED AGAINST ACTUAL RESULTS

**Figure 5** – Example business process in the insurance industry

**Figure 6** – Example Fabric Pattern for Insurance Process
Measuring Effectiveness

We’ve discussed the structure of a pattern-based approach to organizing data quality services. The approach gives a data administrator the ability to choose a pattern than best fits the constraints at hand, modify it if required, and deploy that service within minutes.

As discussed, a reasonable warranty of the data quality can only be given if a measurement of that quality is afforded. Our concluding proposition is that the strongest argument for a pattern-based approach to web services is the ability to measure the quality of the underlying data streams.

A data quality solution should deliver capabilities that allow organizations to inspect the impact of applying such patterns against their data domain, providing quick and immediate feedback without requiring integration efforts. Data inspection instills confidence in the use of templates and expected output can be visually validated against actual results. Pitney Bowes® Spectrum™ Technology Platform Enterprise Data Quality Solution provides design-time data inspection of templates, applicable at any point, which allows designers and administrators to examine the effectiveness of patterns before they are published.

The Spectrum Enterprise Data Quality Solution further provides monitoring of data quality during run-time execution of templates. This then provides a basis for a reasonable warranty: the ability to immediately measure it, and if found defective, ameliorate it immediately. The knowledge that a company’s data is meaningful and of satisfactory quality should allow business managers to confidently serve their customers. We hope Lord Kelvin would agree.
MEANINGFUL AND SATISFACTORY DATA QUALITY ALLOWS BUSINESS MANAGERS TO CONFIDENTLY SERVE THEIR CUSTOMERS

1 In an early treatment of the value of data normalization, Sir William Thomson argued in Lighthouses of the Future (1873) that lighthouse signals in Morse Code are considerably more effective than prevailing methods of the time: “The exceeding simplicity, ease, and certainty of this system of telegraphing may be easily tested by any reader with no other apparatus than a candle or lamp, and a screen held in the hand.”

2 Notes on the Synthesis of Form, Harvard University Press.
