

# The information perspective of SOA design, Part 6: The value of applying the data quality analysis pattern in SOA

Skill Level: Intermediate

[Brian Byrne \(byrneb@us.ibm.com\)](mailto:byrneb@us.ibm.com)  
Industry Models and Integration Architect  
IBM

[John Kling \(jkling@us.ibm.com\)](mailto:jkling@us.ibm.com)  
Consulting and Services Architect  
IBM

[David McCarty \(davidmccarty@fr.ibm.com\)](mailto:davidmccarty@fr.ibm.com)  
IT Architect  
IBM

[Dr. Guenter Sauter \(gsauter@us.ibm.com\)](mailto:gsauter@us.ibm.com)  
Senior IT Architect and Manager  
IBM

[Harald Smith \(smithha@us.ibm.com\)](mailto:smithha@us.ibm.com)  
Product Manager  
IBM

[Peter Worcester \(pworcest@us.ibm.com\)](mailto:pworcest@us.ibm.com)  
Services Solution Marketing Manager  
IBM

17 Apr 2008

Discover the value and approach of data quality analysis in the context of an SOA environment. Learn about the concepts involved in data quality analysis and see the basic steps needed to initiate a data quality assesment project within the broader SOA project. Analyze these issues so that appropriate implementation choices can be made. This is the sixth article in a series called the “The information perspective of

SOA design, " and will be followed by a related article that describes in more detail how the related IBM® products (WebSphere® Information Analyzer) can be used in this context.

## Introduction

### Read all the articles in this series

1. [Introduction to the information perspective of an SOA](#)
2. [The value of applying the business glossary pattern in SOA](#)
3. [Use the IBM WebSphere business glossary in SOA design](#)
4. [The value of applying the canonical modeling pattern in SOA](#)
5. [Use of Rational Data Architect in SOA](#)
6. [The value of applying the data quality analysis pattern in SOA](#)
7. [The execution approach for the data quality analysis pattern in SOA](#)
8. [Use of IBM WebSphere Information Analyzer in SOA design](#)

For SOA services to be successful and reusable, it is important that the data they expose is of acceptable quality for all the consumers. Exposing quality data is necessary in any SOA initiative. Whether it is application services being exposed or a simple generated data query service, the resulting data must be accurate and appropriate to the business context to be of any value.

Data profiling and, in particular, quality analysis are traditional components of enterprise data architecture. A well-designed enterprise data environment includes a continuous cycle of data quality improvement split into three phases:

1. Identifying the data quality problem:
  - Review data quality related issues raised by business and IT users relevant to the project requirements.
  - Define business rules and data quality requirements for each business term and entity that is relevant to the project.
2. Measuring the data quality level:
  - Analyze the source system
    - Apply data quality metrics to data from identified data stores to ascertain data quality levels.
    - Interpret data metric results and translate these results to business terminology. Create detailed reports, charts and summaries that portray data quality levels, and present recommendations.

- Analyze target systems  
Identify gaps between the analyzed source systems and the target system. Create recommendations to resolve gaps.
  - Assess alignment and harmonization requirements for each relevant data element.  
Apply data quality measures to identified data elements to assess current standardization and matching support and translate these results to business terminology. Create detailed reports, charts, and summaries that portray standardization and matching levels and present recommendations.
3. Resolving the data quality problem:
- Identify the root cause of the problem.
  - Formulate a correction process with business case justifications for a particular data quality improvement.

In SOA analysis and design, the practice of data quality analysis plays an additional role, which is to influence service implementation decisions. After service interfaces are identified and designed using top down business requirements, the next step is to decide how the service can be implemented by leveraging existing systems or by writing new code. Each service exposes function and data to the service consumers, and, in doing so, it must meet agreed service levels including data accuracy, response times, and the like. This can only be achieved if the service designer has a detailed understanding of the characteristics of the data stored in each of the systems being leveraged by the service implementation. Data quality analysis provides this understanding.

This article discusses causes of poor data quality. It introduces how to perform a data quality analysis within an SOA project and suggests an approach for finding and investigating those issues. Part 7 of this series introduces the concepts and detailed approach for conducting a data quality analysis.

## Data quality definition

Data quality in SOA can be considered in two dimensions:

- The *technical data quality dimension* defines the data quality criteria often found in both the entity integrity and referential integrity relational rules found in logical data modeling. This dimension includes comparative assessment across source systems and between source and target systems.

- The *business-process data quality dimension* defines the understanding of the key data quality elements in terms of the business definition for a data quality element and the business rules associated with that element. This dimension incorporates the business requirements for standardization and matching or linkage of data quality elements.

Both of these data quality dimensions influence SOA service and process design. Data quality analysis quantifies the characteristics of operational data in the context of its intended use. Understanding these characteristics enables the service designer to be confident that each service can meet its service level requirements and, in some cases, identifies issues that require design changes in the service or the underlying systems to ensure these objectives are met.

## The causes of poor data quality

During the analysis and design lifecycle of an SOA project, a set of services is identified and formally specified, and the necessary preparation steps are made for their implementation. It is important that you understand if the data that you need to expose through a service satisfies the business requirements. For example, to specify a service `retrieveFullCustomerDetails` for which the data resides in multiple repositories, it is important to understand how to integrate the information from the various sources. Can you access each data source and simply join the resulting information, or will there be problems with this approach? Answering this question gives you important information for the implementation of the service.

Figure 1 shows some typical data quality problems that can exist and that can complicate service design.

**Figure 1. Examples of data quality problems**

Cust No	Cust Name	Product	Cost
10	Ms. John Smith	Seats	\$1,200
	Sam Reilly	Chairs	\$2,300
11	Jack Jones	Stools	\$1,750
13	Charles Nelson	Tables	\$A,AA

4. Inconsistent definition

3. Inaccurate data

2. Missing data

1. Invalid data

Bad business-process data quality

Bad technology-defined data quality

Poor data quality's effect on the ability to make sound business decisions can be found both at the technical level and the business level of data definition.

Technology-driven data quality issues are those caused by not applying technology constraints either to the database or during data integration. These include:

- **Invalid data:**  
For example, by not applying constraints, alphanumeric data can be

allowed in a numeric data field (or column)

- **Incomplete or missing data:**  
By not applying key constraints in the database, a field that requires information has been left empty

Business-driven data quality issues are those caused by end users inaccurately maintaining data. Examples include:

- **Inaccurate data:**  
Data quality can be corrupted when inaccurate data is entered into a database. For example, someone could create a record for "Ms. Anthony Jones", rather than "Mr. Anthony Jones", resulting in unnecessary poor data quality.
- **Inconsistent definitions:**  
By having disparate views on what the definition of data quality is, poor quality can be experienced through incorrect use of data. This is especially true in SOA where the business context of the service consumer is not necessarily known by the user or application creating the data.

In order to investigate data quality for an SOA project, the data analyst must have a good understanding of the full scope of both technical and business data quality issues and how they can impact the SOA solution.

### The technical data quality dimension

The technical data quality dimension defines the data quality criteria often found in logical data modeling entity integrity and referential integrity rules. This dimension can be considered at two levels: the domain (data element) level and the entity level. The domain level represents an atomic component (an attribute) such as entry date or gender. The entity level represents a distinct object such as a person, a location, or a product comprised of distinguishing domains or attributes. Particularly in complex or composite services that need bring together information from different sources, this additional aggregated/entity level must also to be considered technically and in its business context.

Key aspects of this dimension are:

**Table 1. Technical data quality dimensions -- Domain level**

Name	Description	Examples of poor, technical data quality
Valid	Data element passes all edits for acceptability and is free from variation and contradiction based on the condition of another data	A customer account type is in a reference list of valid account types. The first name of an individual (person) customer record

	<p>element (a valid value combination).</p>	<p>contains numbers. The Social Security number field should be numeric integer but is populated with alphanumeric characters instead. A customer order record has a ship date preceding its order date.</p>
<b>Unique</b>	<p>The data element used as a primary key or alternate identifier is unique — there are no duplicate values.</p>	<p>Two distinct customer records have the same value in the Social Security number field.</p>
<b>Complete</b>	<p>Data element is:</p> <ol style="list-style-type: none"> <li>1. always required to be populated and not defaulted; or</li> <li>2. required based on the condition of another data element.</li> </ol>	<p>The Unit Type of a product record is missing a value. Married (y/n) field should have a value of 'y' or 'n', but is populated with a "null" value instead. The Social Security number has a default value of all 9's.</p>
<b>Consistent</b>	<p>The data values persist from a particular data element of the data source to another data element in a second data source. Consistency can also reflect the regular use of standardized values, particularly in descriptive elements.</p>	<p>The customer type is an alpha code in one system but a numeric code in another. Different customer number sequences are used in two distinct sources from which data must be obtained. The street address in a data source of 123 Main St Apt 2 and #2 123 Main St are both valid and complete, but not standardized, hence not consistent.</p>
<b>Timely</b>	<p>The data element represents the most current information resulting from the output of a business event.</p>	<p>A customer record references an address that is no longer valid.</p>
<b>Understood</b>	<p>The metadata of the data element clearly states or defines the purpose of the data element, or the values used in the data element can be understood by metadata or data inspection.</p>	<p>A gender code with values of '0' and '1' cannot be understood without additional metadata. The data element named ACED cannot be determined to be the Account Entry Date.</p>
<b>Precise</b>	<p>The data element is used only for its intended purpose, that is,</p>	<p>Product codes are used for different product types between</p>

	<p>the degree to which the data characteristics are well understood and correctly utilized.</p>	<p>different records. The address contains both the address data and an “*” to signal specific downstream system processing.</p>
--	---	--

**Table 2. Technical data quality dimensions -- Entity level**

Name	Description	Examples of poor, technical data quality
<b>Unique</b>	The entity is unique — there are no duplicate values.	While the customer identifier (primary key) is unique, the combined elements of customer name and address are duplicated.
<b>Complete</b>	The required domains that comprise an entity exist and are not defaulted in aggregate.	The product entity has a populated product number and description, but the unit type is defaulted.
<b>Consistent</b>	The entity’s domains and domain values either persist intact or can be logically linked from one data source to another data source. Consistency can also reflect the regular use of standardized values particularly in descriptive domains	The customer entity is comprised of name, address, and area in one system but of name, date of birth, and Social Security number in a second system. The address entity in one system is comprised of 5 address lines while in a second it is highly parsed into unit, street name, street type, directional, and other address information.
<b>Understood</b>	The metadata of the entity clearly states or defines the purpose of the entity and its required attributes/domains.	The entity EMPLOYEE is defined to include name and demographic information, but not address, which is a distinct entity. <b>Note:</b> This dimension borders on the business process dimension of definition, but represents the understanding of the technical implementation of the entity within a given system.
<b>Timely</b>	The entity represents the most current information resulting from the output of a business event.	The customer order system has updated the address entity, but the customer master system does not.

### The business process data quality dimension

The business-process data quality dimension defines the key data quality elements in terms of what the business definition a data quality element is and the business rules associated with that element.

The problem with business-process data quality definitions is that many organizations have inconsistent definitions of and different business rules for similar data. Each line-of-business may have its own understanding of what that element is. The data meaning is always dependent upon:

- The context of the business domain
- The context of an application
- The choices made by the designer
- The usages and practices of end-users

And this varies from one application to another. For example:

- Marketing definition of Net Sales = Sales – Selling expense
- Finance definition of Net Sales = Sales – Selling expense – Merchandise returns

Hence, with disparate views on the definition of a data quality element and its related business rules, when information is compared from different lines-of-business (LOB), the perception of poor quality is created.

**Table 3. Business process data quality for a single data source**

Name	Description	Examples
<b>Semantic definition</b>	The data element has a commonly agreed upon <b>enterprise</b> business definition and calculations.	The "net sales" value is calculated using different algorithms/equations and using different source data for each department within an enterprise.
<b>Accuracy</b>	The data value correctly reflects the real-world condition.	Pat Smith with a gender code of 'F' is in fact a male individual. The address 123 Main St in Ellsworth, WI actually exists.

The perception of poor data quality can lead to the reality of poor data quality when misunderstood or incorrectly applied data definitions are used to build integrated information services joining data from multiple data sources. The following table shows a checklist of the semantic interoperability conditions to consider before exposing information through services in an SOA solution.

**Table 4. Business process data quality dimension -- Integrated data sources**

Name	Description	Examples
<b>Common semantic definitions</b>	Joined data elements with the same name have a commonly agreed upon enterprise business semantic definition, usage and related calculations.	An aircraft company sells an airplane to "customer" and records that information in the sales system. The aircraft company services the plane and stores the "customer" in the engineering system. If the airplane is leased by an operator, then it is quite possible that the two systems have different values of "customer" for the same airplane. A consumer of a service returning the "customer" for a specific aircraft must understand the business context under which the customer is being provided.
<b>Overlapping populations</b>	Interoperability is possible between two systems sharing a partially common population. Sometimes the cross-constraints might be unsustainable.	An internet sales system stores a list of customers who have made a purchase through the website. A retail sales system stores a list of customers who have purchased products supported by after sales warranty. An accounting system stores list of all customers holding a fidelity card. These 3 sets are partially overlapping but each can contain customers that do not exist in the others. The services and processes consuming this data must understand the population gaps and overlaps in order to meaningfully use the data.
<b>Comparable generalization levels</b>	Interoperability is possible between two systems sharing a partially similar generalization level. However, the mapping rules must be clearly defined to avoid potentially costly impedance mismatch errors.	Education institute stores "person" details in one system but subtypes these in "staff", "student" and "faculty" in another system. Services and processes must understand how to navigate this subtype relationship to accurately combine data taken from the different systems.
<b>Comparable aggregation levels</b>	Two data systems may use the same data structure to store semantically different representations of the same real life entities.	A retail distribution company stores each customer as a single financial entity in the accounting system but as multiple physical retail locations in the order management

		<p>system. Integrating information from these two systems requires understanding how this aggregation is resolved. A manufacturer uses the Customer table to store both upstream and downstream supply chain partners. In some cases, the same partner appears multiple times to represent the different relationship roles that exist between them. Services and processes accessing this data must understand the rules that differentiate the reasons why an entity exists in this data store.</p>
<p><b>Semantic drift</b></p>	<p>Unplanned modification of data meaning caused by non-compliant or non-controlled use of operational data.</p>	<p>Obsolete customers cannot be removed from an operational system because no suitable cleanup exists for orphaned records. Instead the users flag obsolete customers by prefixing the customer name with "EXPIRED - DO NOT USE: customer name". A service process not understanding this creative use of the customer name field returns these records in the query result sets.</p>
<p><b>Synchronization mismatch</b></p>	<p>When two source data systems are synchronized outside of a shared transaction boundary, there exists a timing window during which a service accessing both systems may encounter mismatched data.</p>	<p>A customer has a credit limit in an accounting system which is updated nightly based on orders placed. If the order process uses a service from the accounting system to check available credit before accepting an order then the result will not include other orders already placed that day. The result is that the process can allow orders that exceed the customer credit limit.</p>
<p><b>Agreed scale, codes and units</b></p>	<p>Scales, units and measures, as well as type codes (such as country codes, part numbers, etc.) must be matched or mapped for all different sources providing data to a service or process.</p>	<p>Multinational company sells products using metric units in Europe but imperial units in USA. Analyzing sales volumes by unit may be misleading if measurements are not mapped. Joining two data sets sorted by country code will be meaningless if the country codes are not correlated.</p>

<b>Accuracy agreement</b>	The data values correctly reflects the real-world condition across all systems.	The address of Pat Smith has been manually verified and updated in all systems from the system of record.
---------------------------	---	---

Applying a commonly agreed business definition and rules to the data elements provides insurance against inconsistent data quality issues. Building an enterprise-wide business glossary and conceptual data model provides the foundations for getting consistent standards and rules for shared data elements and entities across the enterprise.

## A case study

An IBM customer in the industrial sector implemented a data warehouse without employing data quality analysis. The data model for the data warehouse was based on the attributes identified during the requirements phase. These attributes were mapped to source system logical data models in order to ensure that attributes were available in the source systems that were to feed the data warehouse.

At the data model level, all appeared to be fine. The data models were supported by business glossaries that provided the detail definitions of each attribute and domains of valid values where needed. The problem was that the legacy systems did not have the data entry edits and validations to ensure that the data captured in the application was consistent with the logical data models. Because it was often easier to misuse a field than to change the application, the data values as entered drifted over time away from the data models. When a "Credit Approval Date" was needed but unavailable, users appropriated an unused address line field. The "Short Name" included embedded information about the customer type.

Prior to implementation, the customer's attitude was, "We run our business on this data. It must be good enough for a data warehouse." This was half true. It was good enough for specific users who used applications in their ways to run specific parts of the business. It was not good enough for the managers and analysts who relied on report headings based on metadata. Their reports and queries produced by the data warehouse reflected these defects. This made the data warehouse look very bad. As reports were run and defects identified, they were corrected. However, the cost in repair time and lost credibility was substantial.

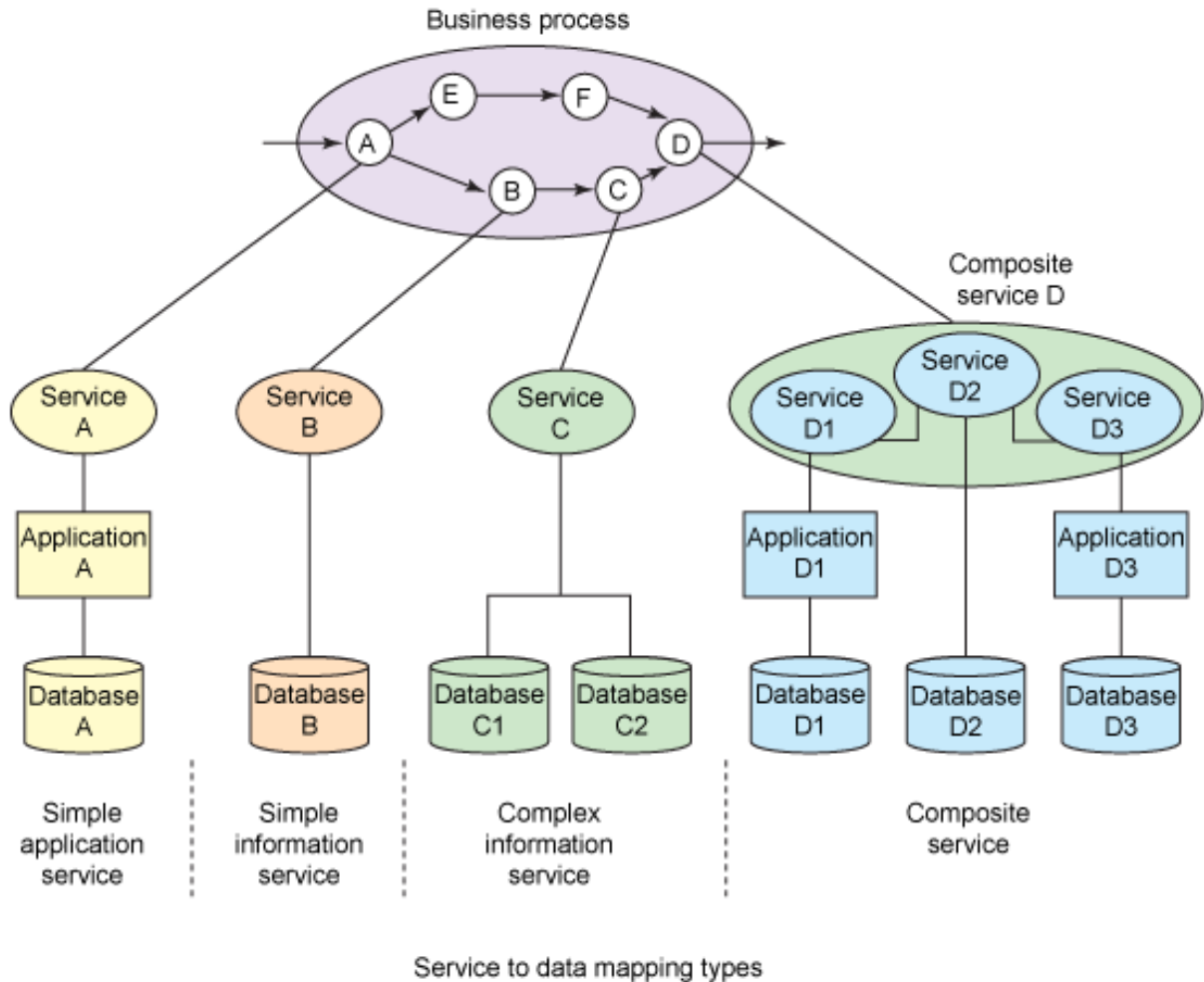
Data quality analysis could have helped to avoid these problems. This is especially important for a data warehouse versus a data conversion to an ERP system. In the latter, defective data is identified during the load process by the ERP system's own edits. In a data warehouse, this is not the case. A custom data warehouse has very few edits for data loaded during the ETL process. For this data, it becomes a "pay me now or pay me later" proposition. Given this choice, up-front data quality analysis is usually the much less painful choice.

## Data quality analysis for an SOA project

The critical component in SOA is the service interface because it delivers the underlying data to the service consumers. There may or may not be a direct relationship between the data elements exposed on the service interface and the underlying physical data stores.

Figure 2 shows examples of the different service implementation types and the mapping variations between the service interfaces and physical data stores.

**Figure 2. Mapping a service to data stores**



All implementation approaches are susceptible to data quality issues if data quality analysis is ignored. The service implementation types are:

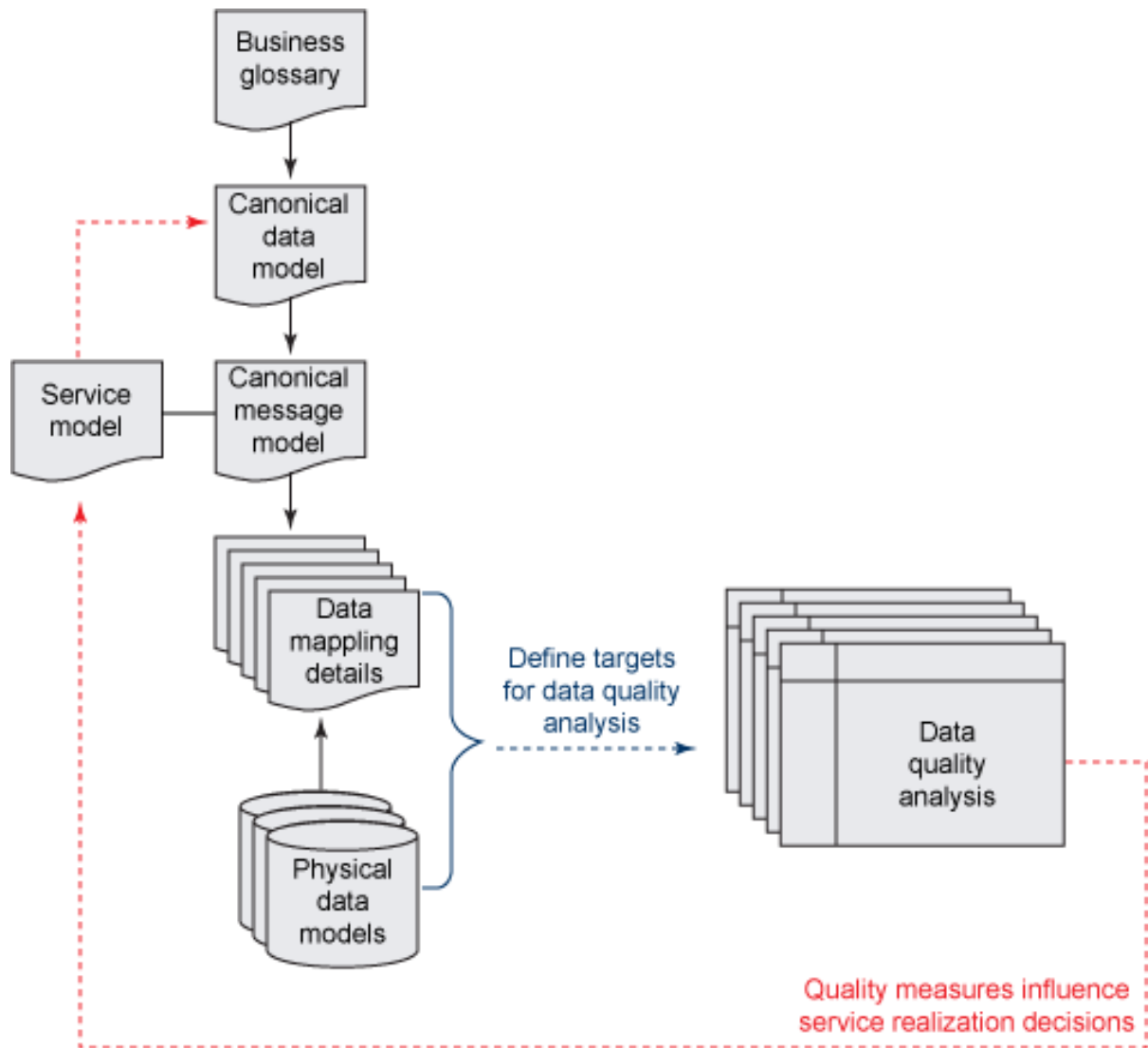
- Simple application service:** The implementation of Service A is as an application API wrapper. The data exposed on the service interface comes from the underlying database via the application code.

Understanding the data quality for this service requires that the analyst first understands how the service interface maps to the application API and then how that API accesses the underlying data. With this knowledge, the correct data elements can be identified and analyzed. Depending on the complexity of the business logic, it may not be possible to assess the quality of the information on the service level just by understanding the quality of the information on the persistence layer.

- **Simple information service:** The implementation of Service B is a simple data access directly to the physical database. For this scenario, the mapping of the service interface to the physical data is all that is needed to identify which data elements should be analyzed.
- **Complex information service:** The implementation of Service C is a data integration access to related databases. For this case, the data integration rules must be identified and then mapped onto the service interface. With this information, the data analyst can determine which data elements should be checked for data quality.
- **Composite service:** Service D represents the most complex case -- the composite service. Here, data flows through service composition logic, service implementation logic and potentially application and data integration logic. All the rules and mappings of these components must be understood for the analyst to identify the correct source data and rules needed to analyze the data quality.

Given the potential complexity of each service implementation, a structured approach is needed to understand where data quality analysis should be applied in any SOA project. This structure comes from the data modeling activities which are performed during service analysis and design. This process is outlined in Figure 3:

### **Figure 3. The role of data quality analysis in SOA design**



In SOA, the service design lifecycle is very iterative, but from a data design lineage perspective, you can see the following sequence of activities:

1. Business terms are defined in the business glossary, and these terms set the shared business definitions for terms used across all other design artifacts.
2. The canonical data model defines data structure, attributes and relationships used for business process and service modeling.
3. As specification detail is added to the service model, you develop the canonical message model which represents the physical instantiation of the canonical data model.

4. Mapping between the SOA canonical data model and the SOA canonical message model against the existing data stores can be leveraged to define potential data quality analysis targets. These mappings are used to identify the mapping and integration rules needed to explore the data quality characteristics within the context of the service model.
5. The results of the data quality analysis are then fed back into the service model to validate implementation decisions or identify cases where alternative design decisions are needed to achieve target levels of data quality.
6. The results of the data quality analysis may also feed back into the canonical data model. For example, if hidden attributes are discovered to be embedded in a text string during data profiling, then these can be made explicit in a revised canonical data model.

So before embarking on a data analysis project, plan to remember these key points:

- Data quality is being assessed against the service interfaces through which it is exposed.
- The primary objective of data quality analysis is to arrive at a yes or no answer to the question: "Will the planned service implementation deliver data quality to the level agreed with the business consumers?"

## An SOA project approach to data quality analysis

The data quality analysis approach for an SOA project is similar to any data quality initiative, but includes some variations in the task details. Critical to success, the data quality analysis itself must be treated as a project itself within the broader confines of the SOA project.

Each project should include the following tasks:

### 1. Define the scope:

The service model is the final definition of what data from the enterprise is in scope for data analysis. The first activity in data quality analysis is to reduce this scope definition from what "can be" investigated to what "will be" investigated.

- Review the service model interfaces and how they can be mapped to physical data stores.
- Use facilitated sessions with business and IT subject matter experts to

identify critical entities and data elements for the solution. Not all defined columns, fields, and elements are relevant to data quality, only those that affect the structure and understanding of information. The data analyst works with business domain experts and application experts to assess which elements are more meaningful than others in order to identify how data quality will be analyzed.

## **2. Review existing data quality information:**

Assemble and review any existing data quality information for the intended project's data elements and entities. This includes previous formal data quality analysis, data quality related problem reports from users, and anecdotal evidence and opinions from subject matter experts. The objective of this task is to reduce the work required by leveraging what is "already known" to prioritize further work and avoid repetition.

## **3. Identify project level data requirements:**

As services are identified during SOA analysis and design, both functional and non-functional requirements are collected in order to define a service level agreement for each service. The data analyst should participate in this process to ensure high-level data quality requirements are included. These high level requirements are then used to formulate the data quality criteria which will be measured and analyzed for the project. These criteria can include all aspects of technical and business process data quality issues for SOA outlined earlier in this paper. The resulting criteria should be reviewed with both business and IT for completeness

Addressing every potential data quality exposure in the solution is often an overwhelming task. Given finite resources (human or product), how is attention focused on what data is critical to analyze? How much data is there to analyze? Data quality criteria should therefore be prioritized as they are collected according to both the perceived severity of each problem as well as the business risks associated with not addressing it.

## **4. Design the data quality analysis execution plan**

From the data quality criteria identified, the data analyst must now design the data execution plan. This includes identifying: who is executing the plan, what is being tested, what tools will be used to test with, when tests will occur, and what the expected outputs of the plan are. Data analysts must be clear on what they are trying to achieve and the deliverables they need to produce to ensure effective execution of the plan.

As part of the execution plan, investigation tests will be identified that meet the project-level data requirements. Some tests may be manual, such as review of gathered models to ensure the technical domains are understood. Some tests can be implemented as queries and/or by configuring data quality analysis/profiling tools

within the available project time frame. The investigation tests examine the data for identified dimensions and generate quantitative results.

The data analyst must, as part of the execution plan, determine against which data sets the investigation tests can be run and who has access to those data sets. For the most accurate results, the investigation tests should be run against full production data. However, there are many reasons which may make this impossible such as workload limitations, security and privacy rules, or if the data is largely unknown. In such cases, the data analyst must build representative data samples from existing test data or sample the production data. If this is done, the results must be reviewed carefully to understand which problems are genuinely representative of the corresponding production data.

Security and authorization to the data sets must be considered here as well. It is important to identify which individuals can perform specific tests, and whether the data, particularly production data, should be analyzed in place or through data extracts.

### 5. Execute the data quality analysis

The process of performing data analysis in the context of a set of SOA requirements is very similar to any data quality initiative. This resultant analysis can be used to reduce project risk and cost as well as to provide more accurate scoping of the project.

In this task, the investigation tests are executed and results are gathered. Details are tooling dependent. If execution occurs over extended periods of time, then care must be taken to ensure no false errors are introduced by timing and data synchronization issues.

Typically, there are 3 phases to consider when doing a full data assessment. Table 5 represents the steps of analysis performed in each phase. These steps are defined and detailed in Part 7 of this series.

**Table 5. Steps of data analysis**

Source system analysis	Target analysis	Alignment and harmonization analysis
<ul style="list-style-type: none"> <li>• Technical dimensions — Domain-level (metadata, domain, structural, and relational integrity, plus business rule assessment)</li> <li>• Technical</li> </ul>	<ul style="list-style-type: none"> <li>• Attribute gap analysis</li> <li>• Data gap analysis</li> <li>• Field length analysis</li> <li>• Data migration scoping</li> </ul>	<ul style="list-style-type: none"> <li>• Attribute alignment analysis</li> <li>• Standardization analysis</li> <li>• Matching remediation analysis</li> </ul>

dimensions –  
Entity-level  
(entity integrity)

## 6. Analyze and report results

In this task, the results are compiled, annotated, and summarized for presentation and discussion with the business and project team. While the format will be tool-dependent, it will most often consist of summary tables and graphs from which it is possible to drill down into the supporting detail data. How and to whom the results are reported should be documented upfront in the data quality analysis plan and indicated in the execution plan. Data analysts should ensure that consistent and standard annotations are created for results. Failure to do so may cause others to unnecessary review and potentially re-do investigation tests and work.

Where results fail to meet the identified requirements, or in cases where results conflict with preconceived expectations, it may be necessary to iterate the tests or to add supplementary tests to explain the results. As with any project, these additional tests represent potential scope change and must be reviewed for risk and impact within the available time line. The availability of automated tools at this point may allow for such scope expansion without significantly impacting schedules.

Where significant problems are discovered, there are three options available to the SOA designer:

- Change the service implementation to mitigate the data problems.
- Request that a change is made outside of the SOA project team. This may be a catalyst for initiating an enterprise data quality initiative in the organization if one does not already exist.
- Choose not to expose the service on the grounds that it cannot meet business service level requirements.

## Conclusion

SOA enables opportunities for wide reuse of the functions and data. With this freedom comes additional responsibility to ensure that service implementations can meet the service levels demanded by the consumers of those services.

This article has outlined the data quality issues that can impact the effectiveness of an SOA service. It then listed the basic steps needed to initiate a data quality assessment project within the broader SOA project and analyze these issues such that appropriate implementation choices can be made.

The detailed data quality analysis pattern will be covered in Part 7 of this series.

# Resources

## Learn

- [IBM WebSphere InformationAnalyzer and Data Quality Assessment](#): This IBM Redbooks® publication discusses how to implement IBM WebSphere Information Analyzer and related technologies in a typical financial services business scenario.
- In the [Information Integration area on developerWorks](#), get the resources you need to advance your skills on IBM's Information Platform & Solutions portfolio of products.
- Browse the [technology bookstore](#) for books on these and other technical topics.

## Get products and technologies

- Download [IBM product evaluation versions](#) and get your hands on application development tools and middleware products from DB2®, Lotus®, Rational®, Tivoli®, and WebSphere.

## Discuss

- Check out [developerWorks blogs](#) and get involved in the [developerWorks community](#).

# About the authors

## Brian Byrne

Brian Byrne has over 10 years experience in the design and development of distributed systems, spending 7 years driving the architecture of Industry Models across a range of industries. Brian is currently an architect within IBM's Information Management organization.

---

## John Kling

John Kling is an architect in the Information Services Practice within IBM's Global Business Services. He is responsible for leading large client engagements that focus on data quality, data integration and master data management. He is currently the data team lead for the SAP implementation of a Fortune 500 industrial company.

---

### David McCarty

David McCarty is based at IBM's European Business Solution Center in La Gaude, France and has 20 years experience designing and developing IT systems with IBM customers. He is currently a member of the Information as a Service Competency Center developing techniques and best practices for leveraging data systems in SOA solutions.

---

### Dr. Guenter Sauter

Guenter Sauter is an architect in the Information Platform & Solutions segment within IBM's software group. He is driving architectural patterns and usage scenarios across IBM's master data management and information platform technologies. Until recently, he was the head of an architect team developing the architecture approach, patterns and best practices for Information as a Service. He is the technical co-lead for IBM's SOA Scenario on Information as a Service.

---

### Harald Smith

Harald Smith is a product manager for data profiling and analysis solutions in IBM's Information Management software group. He has been directly involved in Information Quality work for the last nine years, including work in consulting and product software development and support. With more than 25 years of working with information technology and solutions, Harald has extensive experience in system implementations, solution architecture, project methodology, data auditing and analysis, entity and object modeling, and business process re-engineering. He has worked in software, financial services, healthcare, and education sectors.

---

### Peter Worcester

Peter joined IBM three years ago after almost 25 years at institutions like the US Dept. of Defense, GE Corporate and Morgan Stanley where he held technical leadership positions and gained valuable experience in Enterprise Architecture and Enterprise Data Integration. He initially joined IBM as a Sr. IT Architect as part of the architect team for Information as a Service. Currently he is a Solutions Marketing Manager for the IPS Global Services organization, specializing in MDM solutions.