

Pre-print paper

The final publication of this paper is available at Springer via http://dx.doi.org/10.1007/978-3-319-43997-6_28

November 2016

Andrea Mannocci

DataQ: A Data Flow Quality Monitoring System for Aggregative Data Infrastructures

Andrea Mannocci^{1,2} and Paolo Manghi¹

¹ Consiglio Nazionale delle Ricerche,
Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo", Pisa, Italy
`name.surname@isti.cnr.it`

² University of Pisa, Pisa, Italy
`andrea.mannocci@for.unipi.it`

Abstract. Aggregative Data Infrastructures (ADIs) are information systems offering services to integrate content collected from data sources so as to form uniform and richer information spaces and support communities of users with enhanced access services to such content. The resulting information spaces are an important asset for the target communities, whose services demand for guarantees on their “correctness” and “quality” over time, in terms of the expected content (structure and semantics) and of the processes generating such content. Application-level continuous monitoring of ADIs becomes therefore crucial to ensure validation of quality. However, ADIs are in most of the cases the result of patchworks of software components and services, in some cases developed independently, built over time to address evolving requirements. As such they are not generally equipped with embedded monitoring components and ADI admins must rely on third-party monitoring systems. In this paper we describe DataQ, a general-purpose system for flexible and cost-effective data flow quality monitoring in ADIs. DataQ supports ADI admins with a framework where they can (i) represent ADIs data flows and the relative monitoring specification, and (ii) be instructed on how to meet such specification on the ADI side to implement their monitoring functionality.

Keywords: Monitoring, data flow, data quality, aggregative data infrastructures

1 Introduction

In recent years, the cross-discipline nature of science and the need of researchers to gain immediate access to research material often led to the realization of ADIs [5, 12]. ADIs are information systems offering functionalities to integrate content collected from multiple data sources in order to form uniform and richer information spaces and support communities of users with enhanced access services to such content. In particular, ADIs offer functionalities for (i) the collection and processing of content (metadata descriptions or files), possibly availing of advanced tools and workflows that combine independent components, (ii) the population of uniform graph-like information spaces, and (iii) the provision

of these information spaces to consumers (humans or services) via web portals and/or standard APIs.

Examples of ADIs in the scholarly communication, hence tailoring scientific communication literature data sources, are Google Scholar, OpenAIRE [13], CORE³. Other examples from thematic-domain and/or supporting access to different research products are Europeana⁴, the European Film Archive [2], the Heritage of the People’s Europe [5], the Europeana network for Ancient Greek and Latin Epigraphy [14] and several others.

Aggregating a potentially steadily-increasing amount of data and processing it are some of the major challenges in delivering high-quality and accurate applications based on the resulting information space. To this aim, consumers of the information space typically demand for guarantees on its “quality” and “transparency” over time both in terms of structure and semantics and in terms of the processes generating it [25, 20, 27].

First of all, the perceived data quality at a data source may not be perceived equally at the ADI side as, in general, data quality falls under the “fitness for purpose” principle [25, 27, 23, 7, 26]. Determining whether data has quality or not mainly depends on the application they are intended for. Secondly, data sources may be unstable and unreliable and part of their data may “appear and disappear” entirely because of unpredictable reasons, be them physical (e.g. HW failures, I/O problems, network errors) or logical (e.g. records moved to another collection or deleted by accident). As original data may be unreliable in terms of both quality and presence over time, any knowledge inferred on top of them can be faulty too, thus leading to unwanted, and potentially harmful, end-user dissatisfaction [20]. Lastly, ADIs are often built incrementally in order to adapt to natural evolution of data sources and requirements of the ADI consumers. As a consequence, services, methods, algorithms used by an ADI to collect and process data are subject to changes and updates over time and may introduce unforeseen subtle errors.

Given such premises, in order to deliver a trustful and reliable service to their consumers, ADI administrators must deliver tools for monitoring the internal status, the ongoing processes, and the consistency of results in the running infrastructure. Unfortunately, this operation is often far from trivial. In many cases, ADIs are “systems of systems”, i.e. patchworks of individual components possibly developed by distinct stakeholders with possibly distinct life-cycles, whose interactions are regulated by policies and mediation software developed at the infrastructural level. Accordingly, although there are exceptions, ADIs are generally not equipped with built-in application level monitoring systems and must introduce monitoring capabilities as *ex-post* software integration. Alas, this typically results in excruciating efforts and in a hardly sustainable monitoring software in the long run. Traditional monitoring tools can keep an eye on the system regarding lots of indicators valuable at “system administration level” (e.g. Nagios, Icinga, Ganglia), but they can do little when the focus is on the

³CORE - The UK Open Access Aggregator, <https://core.ac.uk>

⁴Europeana, <http://www.europeana.eu>

application level or, in particular, on the data content and its quality. Application monitoring frameworks such as Prometheus⁵ and the Elastic stack⁶ work great for exporting metrics from the application level, but again are mainly devised for platform operations and performance monitoring, while fail to capture time constraints between metrics generation and address typical problems of data quality.

In this paper, we describe DataQ, a general-purpose system for data flow quality monitoring in ADIs. DataQ supports ADIs administrators with the following features:

- *Workflow-sensitive monitoring*: models and monitors workflow entities of datum, data collection, and process;
- *Cross-platform compatibility*: its adoption is not bound to specific ADI technologies;
- *Cost-effective*: minimizes the amount of work required to integrate monitoring tools on top of ADIs.

DataQ models the *data flows* of a generic ADI taking inspiration from manufacturing processes, where “raw material” corresponds to data passing through sequential processing activities until final data products are produced. The ADI administrator is provided with a Web User Interface (WebUI) in which she can formally describe the data flows to be monitored in the infrastructure, intended as sequences of basic building blocks inspired by the Information Production Map (IP-MAP) model [4, 24]. Once the ADI’s data flows have been defined, DataQ allows administrators to assign *sensors* to the relative building blocks (or sets of them). Sensors specify a set of *metrics*, each corresponding to a number of observations collected over time.

Each defined sensor will require a corresponding *sensor hook* that is an ADI-specific software counterpart, capable of extracting observations for the identified metrics and send them to DataQ. Given the specification of the data flows and the relative sensors, DataQ supports ADI administrators at implementing the corresponding hooks in order to minimize the amount of code to be written and let them focus on the business logic required to compute the metrics observations.

Finally, the WebUI allows administrators to define which *controls* should be enforced over the metrics generated by sensors (e.g. rates, thresholds, upper bounds, moving averages, etc.), visualize trends of the monitored features, analyse comprehensive *reports* and receive *notifications* and *alerts* about potential troubles happening in the operation of the infrastructure.

Outline Section 2 introduces a motivating example based on the experience gathered with the OpenAIRE project which builds an ADI that represents at best the cardinality of the real-scale problem. Section 3 describes the DataQ framework, its languages for data flows, sensors, and controls. Section 4 describes an implementation of the system as deployed in the production system of the

⁵Prometheus, <http://prometheus.io>

⁶The Elastic stack, <https://www.elastic.co>

OpenAIRE infrastructure. Section 5 discusses related work and similar methodologies applied in other fields. Finally, Section 6 concludes the paper and points out future developments.

2 The OpenAIRE (aggregative data) infrastructure

The OpenAIRE⁷ infrastructure fosters Open Access (OA) to research products in Europe and beyond and provides an European data e-infrastructure enabling: (i) the generation of statistics measuring the impact of OA across Europe and individual countries joining the initiative, and (ii) the creation of a service for searching people, publications, projects, and datasets produced as scientific output within H2020 (as well as FP7) and browse their interconnections. The infrastructure collects metadata about millions of publications, scientific datasets, organizations, authors and projects from over 700 heterogeneous data sources and aggregates such content into an information graph in which nodes are interconnected by semantically-labeled relations. Infrastructure services then *harmonize* the structure and semantics of the information graph and *mine* full-text articles (4 million today) to infer extra metadata or links and *enrich* the information graph. Finally, the resulting information graph is exposed for search and stats (portals), bulk-download (OAI-PMH), and navigation (LOD). Certifying the quality of such “data product” and how much reliable is the information provided to the end-users is a non-trivial, yet vital, task for providing trustful and valuable search and statistical services.

However, data quality in OpenAIRE is dependent from a plethora of variables such as the variety and variability of data source, the refinement of algorithms currently in use for curation and mining, the cross-influence between the presence/absence of entire sets of information and the chosen mining algorithms, the presence of network and I/O errors, etc. Also, OpenAIRE persists the data collected and processed in several different back-ends with different purposes. More precisely, the so-called “provision workflow” materializes the information graph (which is stored in HBase⁸ for mining and de-duplication purposes) into: (i) a full-text index, implemented with Apache Solr to support search and browse queries from the OpenAIRE Web portal, (ii) a PostgreSQL and a dedicated Redis key-value cache for statistics, (iii) a NoSQL document storage based on MongoDB in order to support OAI-PMH export of the aggregated records, and finally (iv) a triple store realized with OpenLink Virtuoso to expose OpenAIRE’s information system as LOD via a SPARQL end-point.

OpenAIRE is indeed an ADI, and features serious monitoring challenges. Looking at the provision workflow above, we shall consider three representative examples of monitoring: (i) assessing (and keep assessing over time) whether the total number of “publications funded by Horizon2020 projects” indexed by Solr matches the same number of records delivered via the corresponding OAI-PMH Set; (ii) the number of publications harvested from a content provider should be

⁷The OpenAIRE EU project, <http://www.openaire.eu>

⁸Apache HBase, <https://hbase.apache.org>

monotonic increasing with a max percent variation between each harvesting over time; and, (iii) evaluating the “completeness” of records index by Solr evaluated as the ratio between number of mandatory attributes and number of not empty mandatory attributes.

3 DataQ: a Data Flow Quality Monitoring system

This section introduces DataQ, a general-purpose, flexible, and cost-effective system for data flow quality monitoring. It first presents its data flow description language and data flow monitoring language, finally it gives an overview of its architecture.

3.1 Data flow description language

Taking inspiration from the seminal work present in literature [3, 10, 28, 4, 24], a generic Aggregative Data Infrastructure (ADI), or a subpart of it, can be accurately modeled as an Information Manufacturing System (IMS): a series of processing steps that transforms raw input data into final data products. In this work, we adopted a subset of the construct blocks introduced in the Information Production Map (IP-MAP) model [24], reported in Fig. 1a. The *Data Vendor* block (*VB*) represents a generic source, either internal or external, providing data for downstream elaboration. Similarly, the *Data Consumer* block (*CB*) represents an entity downstream consuming final data information produced by the manufacturing process. The *Data Processing* block (*PB*) represents a generic processing step which processes a (stream of) *datum* from a status to another one. Finally, the *Data Storage* block (*SB*) stands for a back-end of choice persisting collections of data conforming to a common structure and semantics (e.g. E-R database, NoSQL, file system). According to this model, *PBs* and *SBs* handle data units of different granularity: *datum*, i.e. the observable unit of a *PB*, and *data collection* (a collection of data sharing a common property or semantic), i.e. the observable unit of a *SB*.

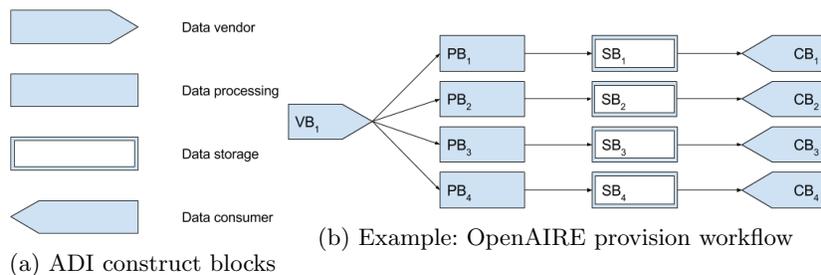


Fig. 1: ADI modeling description language

ADIs data flows are modeled as sequences of instances of these blocks interconnected by directed *edges* representing time-order relations among them. The

language does not model locality assumptions, i.e. ADI data flow blocks may refer to services running on different administrative domains. The example in Fig. 1b describes the OpenAIRE provision workflow (introduced in Section 2) with the formalism here introduced. The workflow ingests data from VB_1 (i.e. the OpenAIRE information graph) and transforms it through separated processing blocks PB_1 , PB_2 , PB_3 and PB_4 . Then, the result of each PB_i is stored separately in a dedicated backend SB_1 , SB_2 , SB_3 and SB_4 . Finally, each storage block SB_i serves a specific use case, hence the customers CB_1 , CB_2 , CB_3 and CB_4 are, at least logically, distinct.

3.2 Data flows monitoring language

Once the ADI *data flows* have been described, the ADI admin can start configuring the relative monitoring scheme. To this aim, DataQ supports four main concepts: *sensors*, *metrics*, *controls* and *actuators*.

Sensors A sensor s is a piece of software capable of generating an *observation* o about a data unit u (either a datum or a data collection) according to a metric m . A metric is a function intended to measure a specific quality feature of a target data unit u (e.g. the completeness of a XML record to be processed, the cardinality of a collection). A sensor is “anchored” to a block of reference b , be it a PB or an SB, and defines a set of metrics m_i . More specifically, a sensor s is described as a tuple of the form $s = \langle b, m_1, \dots, m_I \rangle$, which lives in the ADI and is triggered by it in order to produce observations o_i , described as tuples of the form $o = \langle time, sensor, metric, value \rangle$. When a sensor s , anchored on block b , is triggered at time t on data unit $u_{b,t}$ it produces a set of observations, one for each of the metrics m_i it declares:

$$trigger(s, t) = \{o_1, \dots, o_I\} = \left\{ \langle t, s, m_1, v_1 \rangle, \dots, \langle t, s, m_I, v_I \rangle \right\} \quad (1)$$

where $v_i = m_i(u_{b,t})$ and $u_{b,t}$ is the data unit relative to the block b at time t , e.g. the datum processed by a PB at time t or the *data collection* content at time t in a SB . When the sensor is anchored to a PB, it generates observations for each individual datum processed by the block (e.g. XML, JSONs, texts, images). When anchored to an SB, the sensor generates observations relative to the entire data collection at hand (e.g. query over SQL database, query over full-text index).

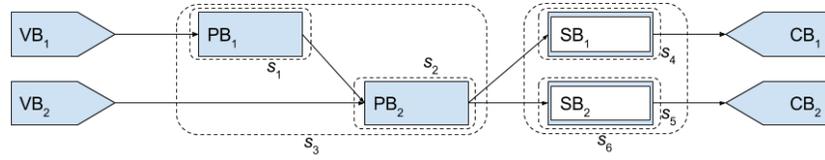
Compound sensors *Compound sensors* are sensors s' that can enclose previously defined sensors and thus inherit their *scope* in order to create *compound metrics* m'_k . A compound sensor can be defined as $s' = \langle s_1, \dots, s_J, m'_1, \dots, m'_K \rangle$, where each m'_k is a function that produces a new v'_k by combining observations produced by a subset of the metrics anchored to the sensors in the scope of s' . While basic sensors are triggered by the ADI, compound sensors are triggered by DataQ according to a given admin-defined schedule. When a compound sensor is triggered at time t , it produces a set of observations according to the m'_k

compound metrics defined, whose observation time is t :

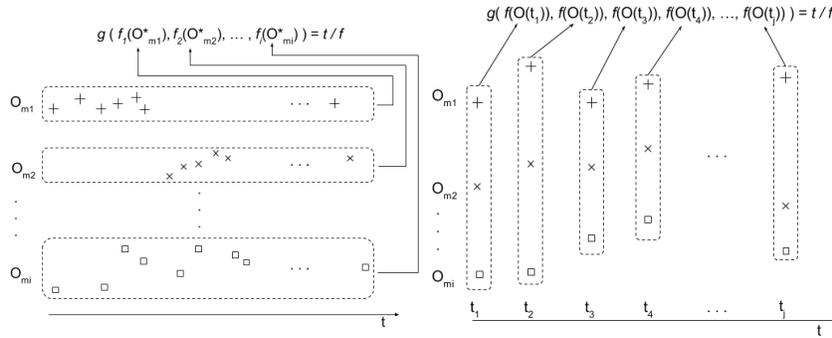
$$\text{trigger}(s', t) = \{o'_1, \dots, o'_K\} = \left\{ \langle t, s', m'_1, v'_1 \rangle, \dots, \langle t, s, m'_K, v'_K \rangle \right\} \quad (2)$$

To better understand the implications of this, Fig. 2a depicts two examples of compound sensors: an *in-sync* sensor s_6 wraps two storage sensors s_4 and s_5 , which monitor two blocks SB_1 and SB_2 ; a *not-in-sync* sensor s_3 wraps s_1 and s_2 , which monitor two sequential blocks PB_1 and PB_2 respectively. When an in-sync compound sensor wraps two SBs as s_6 in Fig. 2a it has the opportunity to synchronize the observations produced by them. If O_{m_i} is the time series of observations produced by the metrics m_i of s_4 and s_5 , metrics m'_k in s_6 generate a new compound value $v'_k = m'_k(O_{m_{1k}}(t), \dots, O_{m_{I_k}}(t))$ – where $O_{m_i}(t)$ “reads” the time series O_{m_i} at time t . In this case, a compound sensor triggered at time t produces a new derived observation for every defined m'_k by combining into a function the values produced by the enclosed metrics of interest at time t .

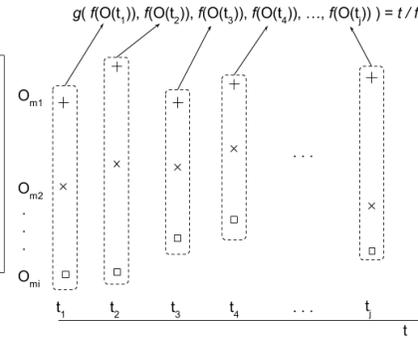
A not-in-sync compound sensor cannot synchronize the enclosed metrics (e.g. s_3 in Fig. 2a). Hence, new compound value v'_k is obtained as a function of partial evaluations, one for each O_{m_i} of interest for m'_k , i.e. $v'_k = m'_k(O_{m_{1k}}, \dots, O_{m_{I_k}}) = g_{m'_k}(f_{m'_k,1}(O_{m_{1k}}), \dots, f_{m'_k,I_k}(O_{m_{I_k}}))$, where $\{f_{m'_k,1}, \dots, f_{m'_k,I_k}\}$ is a set of functions dedicated to process one O_{m_i} each and produce partial evaluations, while $g_{m'_k}$ is the function combining such partial evaluations.



(a) Wrapping example



(b) Not synchronized sensors



(c) Synchronized sensors

Fig. 2: Types of sensor and relative controls

Controls Once sensors and metrics are defined, the admin can configure the *controls* she wants to enforce over time against its sensors. Controls depend on the scope (possibly inherited) of the sensors to which they are anchored and their relative time ordering, hence DataQ can guide the user in their definition. More precisely, a control c is defined in terms of one or more *selectors* x_i and one *analyzer* a as a function $c(x_1, \dots, x_K, a) = \{true, false\}$.

Given a set of observations of a metric O_m and two predicates p_t, p_v on observations' time and value, a selector x returns a subset O_m^* of O_m satisfying the predicates:

$$x(O_m, p_t, p_v) = \{o_1^*, \dots, o_I^*\} = O_m^* \subseteq O_m \quad (3)$$

With *analyzer* we refer to a comparing function a that accepts one or more groups of selected observation $O_{m_j}^*$ as input and returns a binary result $\{true/false\}$ reflecting whether the controls has been passed or not. Two possible types of analyzers are provided, one working with not synchronized observations (Fig. 2b)

$$a(O_{m_1}^*, \dots, O_{m_J}^*, f_1, \dots, f_J, g) = g(f_1(O_{m_1}^*), \dots, f_J(O_{m_J}^*)) = \{true, false\} \quad (4)$$

and one working with synchronized ones (Fig. 2c).

$$a_{sync}(O_{m_1}^*, \dots, O_{m_J}^*, f, g) = g(f(O^*(t_1)), \dots, f(O^*(t_i))) = \{true, false\} \quad (5)$$

where $O^* = \bigcup_{j=1}^J O_{m_j}^*$ and $O^*(t) = \{O_{m_1}^*(t), \dots, O_{m_J}^*(t)\}$.

Actuators Finally, DataQ can provide feedback to the infrastructure thanks to an *actuator* component. The actuator can be used for driving the data infrastructure behavior in order to correct automatically or at least compensate an issue revealed by failed controls. For example, a data source could be automatically graylisted when the quality of the exported data drops down a certain level and whitelisted when the the quality rises back to the expected value. The actuator runs within the ADI and waits for “stimuli” from DataQ in order to take user-defined countermeasures which it has been designed for.

3.3 Overview of the DataQ architecture

DataQ, whose architecture is reported in Fig. 3, is designed as a client-server application. The infrastructure source code needs to be instrumented in order to put sensors and actuators in place, plug their respective behaviours properly and deal with communication to the server counterpart. Currently DataQ can be integrated with the ADI in two ways: (i) a core Java module can be imported and used by the infrastructure's code in order to focus just on the implementation of metrics' core logic, leaving the rest of low-level abstraction to the framework (e.g. communication layer, sensor scaffolding, etc.); (ii) a low-level REST API, which has to be called by the ADI components in order to provide observations to DataQ server component. For the sake of configurability and extensibility, when importing DataQ's Java libraries, a catalog of common metrics (e.g. for XML inspection) is provided off-the-shelf from the framework.

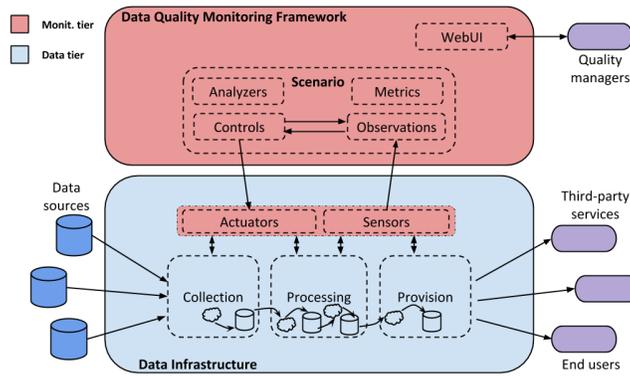


Fig. 3: DataQ Architecture

The DataQ *server* component is a stand-alone web application that enables the infrastructure manager to create, via a user friendly WebUI, one or more *monitoring scenarios*, each one designed to monitor a particular functional area or aspect of the ADI. Within a monitoring scenario, the infrastructure manager is able to model the data flows of the ADI as seen in Section 3.1, the deployed sensors and actuators, and the controls to be enforced over time against the generated metrics. Similarly to sensors, analyzers used by controls come with off-the-shelf implementations for most common comparison functions such as equality, upper and lower bound, peak/valley detection, threshold, percent variation, and so forth. In any case, analyzers can be customized and extended at need by the user requiring more specific behaviours. The DataQ server will separate the stream of incoming observations into different streams, each one related to a specific metric, and will store them as points of time series. In this way, observations can be queried and examined either as charts or tabular data via WebUI.

Furthermore, DataQ enables the generation of an exhaustive *report* about the defined metrics and controls providing insights, via the WebUI, in a quick glance about key features and potential issues present in the infrastructure. Given a set of controls, the monitoring service also takes care of raising *alerts* and *notifications* informing the infrastructure manager about the status of the infrastructure and its operation.

Finally, regarding actuators, the communication between DataQ and the actuators deployed in the ADI takes place through standard Web API; the developer using DataQ has the option (i) to use a Java ready-to-use actuator scaffolding which has only to be specialized with custom business logic, or (ii) take charge of low-level implementation in order to detect and react to such feedback.

4 The OpenAIRE use case

An implementation of DataQ has been realized and deployed in the OpenAIRE production infrastructure to monitor the “provision workflow” described in Section 2. For the sake of space, we selected a representative use case for demonstration (Fig. 4). Here, the metric “publications”, has been extracted from two storage blocks (a Solr index and a Redis key-value DB instances) thanks to

two sensors extracting the total number of publications collected by OpenAIRE. As we can see, though in recent history, the values of the two time series are perfectly matching, hence the control passes (the green box on the right). The second control checks whether the last three observations taken on Redis are monotonic increasing with no “bumps” (meaning the percent variation between two subsequent observations should not exceed 15%); as this condition is verified, the other control passes too (the green box on the left).



Fig. 4: Screenshot of the DataQ dashboard for the metric “Publications”

5 Related work

Data quality has been recently investigated in the field of eScience; in [21, 22], the authors proposed a framework for the evaluation of the quality of information during Finite Element Method (FEM) simulations carried out on a machine availing of Workflow Management Systems (WfMSs), a situation in which poor data quality generally yields a waste of time, especially during long running experiments and simulations.

In [17, 19], the authors developed a framework able to manage quality attributes in the context of eScience based on formal Information Quality ontology and proposed a case study in the proteomics field WfMS with experiments based on Taverna⁹.

Other works have been presented in the field of (heterogeneous) data integration systems [15, 16, 1, 18, 6, 8]. In [9, 11], the authors described the QBox platform, a system capable of assessing quality in data integration systems and enabling mediation among several quality evaluation and data profiling tools.

None of these approaches has tackled monitoring with the idea of providing the tools to model data flows, sensors, metrics, and controls in a conceptual sense, trying to push complexity of the monitoring framework away from ADI admins, who should only focus on the business logic of the metrics. To our knowledge,

⁹Taverna, <http://www.taverna.org.uk>

DataQ is the first framework of this kind, its first implementation being already in use in the production system of OpenAIRE.

6 Conclusions

In this paper we introduced DataQ a system for Data Flow Quality Monitoring of Aggregative Data Infrastructure (ADI). The proposed solution poses its foundation on the seminal work presented in [3, 10, 28, 4] to model such data infrastructures as Information Manufacturing System (IMS) leveraging a subset of the construct blocks available in IP-MAP [4, 24].

DataQ enables the automatic extraction of quality features from data thanks to specially devised software sensors. Such observations, persisted as time series, can be inspected and automatically queried in order to check for used-defined controls about the expected behavior of the infrastructure.

The infrastructure manager is provided with alerts and notifications when the infrastructure as soon as the ADI encounters anomalies. We also foresee a mechanism for providing feedback to the monitored infrastructure in order to trigger a prompt reaction to anomalies and attempt, to some extent, an automatic correction of the misbehaviour.

An implementation of DataQ is currently deployed in the OpenAIRE infrastructure [13] and helps the infrastructure managers to spot anomalies and prevent the dissemination of erroneous data and statistic to the general public and to EU commission.

References

1. Akoka, J., Berti-Équille, L., Boucelma, O., Bouzeghoub, M., Comyn-Wattiau, I., Cosquer, M., Goasdoué-Thion, V., Kedad, Z., Nugier, S., Peralta, V., Sisaid-Cherfi, S.: A framework for quality evaluation in data integration systems. 9th International Conference on Enterprise Information Systems, ICEIS (2007)
2. Artini, M., Bardi, A., Biagini, F., Debole, F., Bruzzo, S.L., Manghi, P., Mikulicic, M., Savino, P., Zoppi, F.: The creation of the European Film Archive : achieving interoperability and data quality. In: 8th Italian Research Conference on Digital Libraries, IRCDL. pp. 1–12 (2012)
3. Ballou, D.P., Pazer, H.L.: Modeling Data and Process Quality in Multi-Input, Multi-Output Information Systems. *Management Science* 31(2), 150–162 (1985)
4. Ballou, D., Wang, R., Pazer, H., Kumar.Tayi, G.: Modeling information manufacturing systems to determine information product quality. *Management Science* 44(4), 462–484 (1998)
5. Bardi, A., Manghi, P., Zoppi, F.: Aggregative data infrastructures for the cultural heritage. *Communications in Computer and Information Science, CCIS* 343 (2012)
6. Batini, C., Barone, D., Cabitza, F., Grega, S.: A Data Quality Methodology for Heterogeneous Data. *International Journal of Database Management Systems* 3(1), 60–79 (2011)
7. Batini, C., Cappiello, C., Francalanci, C., Maurino, A.: Methodologies for data quality assessment and improvement. *ACM Computing Surveys* 41(3) (2009)

8. Boufares, F., Ben Salem, A.: Heterogeneous data-integration and data quality: Overview of conflicts. 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications, SETIT pp. 867–874 (2012)
9. González, L., Peralta, V., Bouzeghoub, M., Ruggia, R.: Qbox-services: Towards a service-oriented quality platform. In: Lecture Notes in Computer Science. vol. 5833 LNCS, pp. 232–242 (2009)
10. Huh, Y., Keller, F., Redman, T., Watkins, A.: Data quality. *Information and Software Technology* 32(8), 559–565 (oct 1990)
11. Lemos, F., Bouadjeneq, M.R., Bouzeghoub, M., Kedad, Z.: Using the QBox platform to assess quality in data integration systems. *Ingenierie des systemes d'information* 15(6), 105–124 (2010)
12. Manghi, P., Artini, M., Atzori, C., Bardi, A., Mannocci, A., La Bruzzo, S., Candela, L., Castelli, D., Pagano, P.: The D-NET software toolkit. *Program* 48(4) (2014)
13. Manghi, P., Bolikowski, L., Manola, N., Schirrwagen, J., Smith, T.: OpenAIREplus: The European scholarly communication data infrastructure. *D-Lib Magazine* 18(9-10) (2012)
14. Mannocci, A., Casarosa, V., Manghi, P., Zoppi, F.: The Europeana Network of Ancient Greek and Latin Epigraphy Data Infrastructure. In: *Metadata and Semantics Research: 8th Research Conference, MTSR 2014*. pp. 286–300 (2014)
15. Marotta, A., Ruggia, R.: *Quality Management in Multi-Source Information Systems*. Quality (2002)
16. Marotta, A., Ruggia, R.: Managing source quality changes in a data integration system. *CEUR Workshop Proceedings* 263, 1168–1176 (2006)
17. Missier, P., Preece, A., Embury, S., Jin, B., Greenwood, M., Stead, D., Brown, A.: Managing information quality in e-Science: A case study in proteomics. *Lecture Notes in Computer Science* 3770, 423–432 (2005)
18. Peralta, V., Ruggia, R., Kedad, Z., Bouzeghoub, M.: A Framework for Data Quality Evaluation in a Data Integration System. In: *SBBD*. pp. 134–147 (2004)
19. Preece, A.D., Jin, B., Pignotti, E., Missier, P., Embury, S.M., Stead, D., Brown, A.: Managing Information Quality in e-Science Using Semantic Web Technology. *Procs. ESWC* pp. 472–486 (2006)
20. Redman, T.C.: The Impact of Poor Data Quality on the Typical Enterprise. *Communications of the ACM* 41(2), 79–82 (1998)
21. Reiter, M., Breitenbücher, U., Dustdar, S., Karastoyanova, D., Leymann, F., Truong, H.L.: A novel framework for monitoring and analyzing quality of data in simulation workflows. In: *IEEE 7th International Conference on E-Science*. pp. 105–112 (2011)
22. Reiter, M., Breitenbücher, U., Kopp, O., Karastoyanova, D.: Quality of data driven simulation workflows. *Journal of Systems Integration* 5(1), 3–29 (2014)
23. Scannapieco, M., Missier, P., Batini, C.: Data Quality at a Glance. *Datenbank-Spektrum* 14(January), 6–14 (2005)
24. Shankaranarayanan, G., Wang, R.Y., Ziad, M.: IP-MAP: Representing the Manufacture of an Information Product. *Proceedings of the 2000 Conference on Information Quality* pp. 1–16 (2000)
25. Strong, D.M., Lee, Y.W., Wang, R.Y.: Data quality in context. *Communications of the ACM* 40(5), 103–110 (may 1997)
26. Tani, A., Candela, L., Castelli, D.: Dealing with metadata quality: The legacy of digital library efforts. *Information Processing and Management* 49(6) (2013)
27. Tayi, G.K., Ballou, D.P.: Examining data quality. *Communications of the ACM* 41(2), 54–57 (1998)
28. Wang, R., Storey, V., Firth, C.: A framework for analysis of data quality research. *IEEE Transactions on Knowledge and Data Engineering* 7(4), 623–640 (1995)