Working Paper

UNITED NATIONS
ECONOMIC COMMISSION FOR EUROPE

CONFERENCE OF EUROPEAN STATISTICIANS

Workshop on Statistical Data Editing
(Neuchâtel, Switzerland, 18-20 September 2018)

A GENERIC VALIDATION REPORT FOR THE ESS

Prepared by Olav ten Bosch and Mark van der Loo[1]
Statistics Netherlands

I. Introduction

1. Data validation is one of the cornerstones of the production of official statistics. Whether it is the input received from a survey, a dataset received from an administrative source or the result from some internet scraping, data must be checked against our expectations in order to process it and turn it into reliable statistics. Data needs to be *validated* and the outputs of the possibly numerous validation checks, *reported* in some way need to be studied and processed. Especially in the case of cross-organisation validation, where both a data producer and a data receiver check the same data against some commonly agreed validation rules, *standardisation* of reporting is crucial to prevent different interpretations of the results. Harmonizing validation reports improves mutual understanding between organizations.

2. Within the ESSnet project **Validat Integration** [ESS, 2017] Statistics Netherlands, working together with partner countries Germany, Lithuania, Poland, Portugal, Sweden and Eurostat, developed a *generic report structure* to express validation results. To maximize the applicability in the European Statistical System (ESS) it was designed in such way that it can be used as an output from *any validation tool*, using *any validation language*, on both *micro-data* as well as *aggregated data* and in *any stage of the statistical process*. This was achieved by concentrating on the core elements in a validation process: the data that were validated, the rules that were checked, properties of the validation procedure, and the validation results that came out.

3. The generic report is *extensible* to support use in different contexts such as reporting in a national editing system. In such contexts there may be a need to add additional, domain specific results. To facilitate the use of the report in a statistical production chain the validation report has an efficient implementation in the machine-readable JSON format [ECMA, 2013]. This format can easily be generated by validation tools and consumed by other tools. In addition, software has been built to view, filter, aggregate and export a validation report into a human-readable format.

4. For the conceptual design and the technical standard itself we refer to the technical deliverable of the ESSnet (van der Loo and ten Bosch [2017]). In this paper we explain the most important concepts and we demonstrate some of the prototype implementations. In addition we highlight the role the report structure can play in ESS standardisation, both for cross-organisation validation as well as for validation purposes within the core processes of a National Statistical Institute (NSI).

[1] obos@cbs.nl, mplo@cbs.nl

5. In Chapter II we touch upon other deliverables in the context of validation. In Chapter III we motivate the need for a machine-readable as well as human readable version of the report and we report on the demands we identified designing the report. Chapter IV contains an explanation of the basic concepts of a validation report, such as a validation event, and shows how the report looks in practise. Chapter V focuses on the pilot software that has been created to test and support the uses of generic validation reports, Finally, in chapter VI we summarize and conclude.

II. Validation principles and methodology

1. Recently two important deliverables have been written down in the area of validation in the ESS. One of them is the set of validation principles identified by the ESS validation task force, see ESS [2017, Chapter 4]. These principles:

(1) The sooner, the better
(2) Trust, but verify
(3) Well-documented and appropriately communicated validation rules
(4) Well-documented and appropriately communicated validation errors
(5) Comply or Explain
(6) Good enough is the new perfect

were used in the design of an international data validation architecture. Especially principle number (4) underlines the need for a generic validation report. In more detail this principle states that *the error messages related to the validation rules need to be clearly and unambiguously defined and documented, so that they can be communicated appropriately to ensure a common understanding on the result of the validation process.* It was recognized that for the ESS this principle implies the definition of a standard validation report structure that is expressive, clear and unambiguous.

2. Another important deliverable in the validation landscape is the methodological handbook on data validation Di Zio et al. [2015]. This work contains an explanation of data validation in general, the relationship with quality and many more specific subjects such as a formal typology of validation rules, some reasoning on the validation process life cycle and theories about properties of validation rules such as completeness, redundancy, feasibility and complexity. We will not repeat the methodological background described in this handbook and instead refer to this handbook for a conceptual discussion on data validation. The design of the report aligns as much as possible with the concepts in the methodological handbook.

III. Demands for a validation report

1. Having recognized the need for a generic validation report, the next questions to answer are, *how should it look like?* and *what should be in it?*. These questions are addressed in the following subsections.

A. Machine readable and human readable

1. For clear and unambiguous communication, a technical standard that defines how each information element is to be stored and in a data structure is indispensable. On one hand, standardizing the data transfer format facilitates reliable data transfer, automated processing, and application building. On the other hand, technical data storage formats are often cumbersome for humans to read.

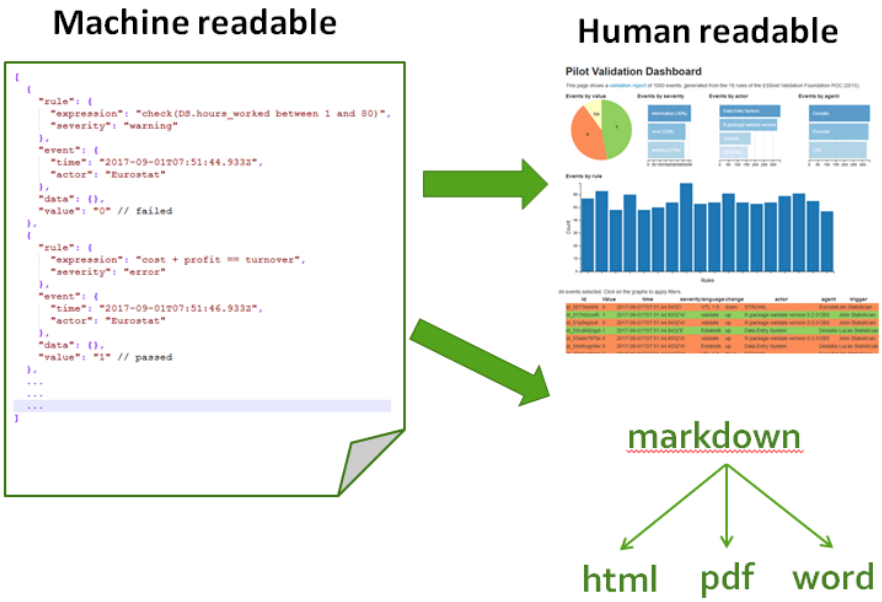

FIGURE 1. Machine readable and human readable formats.

2. To make the standard applicable in machine to machine communication contexts as well as in human contexts we designed a machine-readable format as well as two prototypes of a human readable format that can be automatically derived from the machine-readable format. Figure 1 shows the relationship between the machine-readable and the two prototype human readable formats. One of the prototypes is an interactive dashboard that can be used to aggregate, filter and inspect validation events. The other prototype is a translator to the popular markdown format which can be processed into other well know formats such as html, pdf and word. Both are publicly available and open source to encourage re-use.

B. Contents of a validation report

1. Ultimately, a validation report should communicate validation results in a way that is informative enough to lead to a targeted search for the cause of invalidities. To identify which information elements are needed for this, Figure 2 gives a high-level overview of a data validation procedure. At the input side, we find the data to be validated and the validation rules that the data are supposed to satisfy. At some point in time, the data are confronted with the rules and validation results are created. The procedure as a whole generates and processes a lot of information that can possibly be included in such a report. For example, the validation rules may be endowed with metadata such as descriptions and severity level and for the validation procedure it may be interesting to record a timestamp and the used software.

2. To make choices on the elements to be included in the validation report we took two approaches. The first approach was to collect a number of example validation reports from member states and Eurostat, and record what information was stored therein. This resulted in a long-list of validation report elements categorized into several classes such as rule metadata, process metadata, aggregates, and more. Figure 3 shows the validation report elements aggregated by type. Clearly, validation results, process meta and aggregates play an important role.

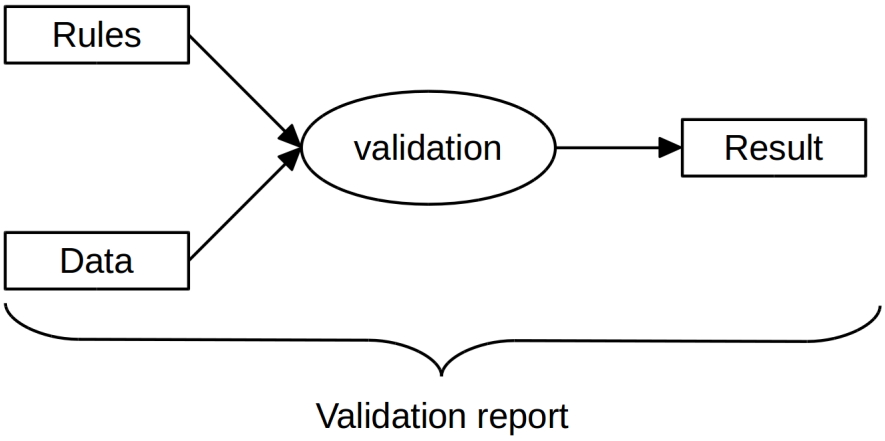

FIGURE 2. Information elements involved in creating a validation result, relevant for validation reports.

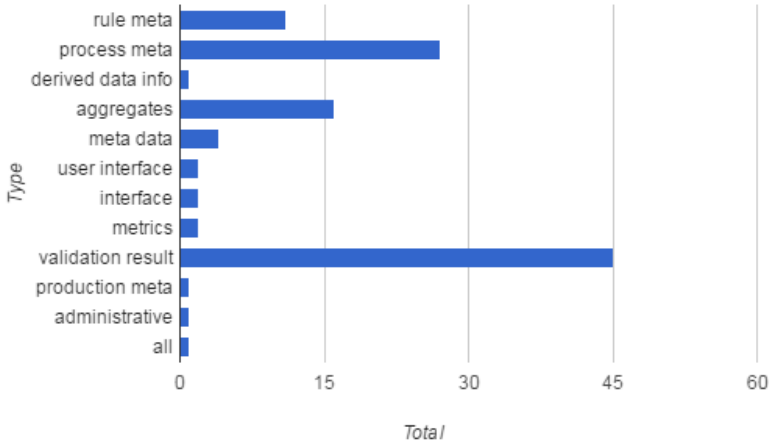

FIGURE 3. Validation report elements aggregated by type.

3. The second approach was to define a set of high-level demands that a technical format should satisfy in order to be acceptable as a report. The advantage of a set of formal demands (to be specified below) is that it becomes possible to reason about validation reports in a structured way, which increases understandability and programmability. Once the report was designed with high-level demands in mind, it was tested whether it was expressive enough to harbour the elements gathered from the examples in the long-list. This turned out to be the case in general, providing evidence that the resulting report structure is generic enough to be used in a wide range of validation contexts in many institutes but also flexible enough to be adopted in a regional validation context. Below, we provide details on the high-level technical demands (in condensed form) since they contain the core concepts behind the validation report structure.

4. One of the criteria for selecting fields came from the observation that a validation result that cannot be identified with the data and rules it pertains to is useless. If the reports are to be used for communication across organizations, or if they are to interpreted separately from the process that triggered the validation, the relation with the rules and data needs to be included. This is a direct consequence of the validation principle *Well-documented and appropriately communicated validation errors* [ESS, 2017] and leads us to the first demand.

TABLE 1. Example data, validation rules, and results for six validation events.

	Variables		Rules	
Nr	Age	hasjob	`Age >= 0`	`IF Age < 15 THEN hasjob == "no"`
1	36	`yes`	1	1
2	53	`NA`	1	NA
3	11	`yes`	1	0

Demand 1 (Identification)**.** *A validation report shall convey validation results such that they can be identified with the validation procedure, the validation rules used, and the validated data.*

5. A second demand came from the observation that a validation procedure usually involves multiple rules and each rule may concern different subsets of variables, records or reference datasets. Every confrontation of a rule with the data can be seen as a validation (sub)procedure yielding a validation report. To gather all results, validation reports should be able to be combined to an overall report.

Demand 2 (Closure under combination)**.** *Two validation reports shall be combinable in such a way that the result is again a validation report that includes all information that separate reports contained.*

This includes cases where multiple procedures are involved, possibly related to varying datasets, rule sets, and actors involved in the validation procedures.

6. Finally, depending on intended use, one may be interested in the details of each step in each validation procedure, or in a more aggregated view of one or more validation events. Indeed, a quick view on some example validation reports from multiple NSI's and multiple statistical domains shows that most of them contain some kind of aggregated results within the validation report itself. Therefore, we also demand the following.

Demand 3 (Closure under aggregation)**.** *A validation report can be aggregated such that the result is again a validation report.*

7. Many types of aggregation may be relevant. Obviously counting the number of passes and fails must be supported but also more complex metrics based on aggregation are supported. In this paper we do not dive any further into the support of aggregates into a report itself, we however come back to the creation of aggregates when discussing the dashboard in §V.

IV. Core concepts

1. To get an idea of the core concepts in the identification of a validation result, we give an example. Consider the data, validation rules and validation results shown in Table 1. When a dataset is confronted with a validation rule, there are three possible outcomes. A rule can be satisfied, yielding 1, a rule can be failed, yielding 0, or a rule cannot be evaluated because of missing data, yielding NA (Not Available).

2. In the example three records on age and work status are checked against two rules: age must be larger than or equal to zero, and persons under 15 years old cannot have a job. In each case the demand on age can be checked and each record passes this test, yielding 1 (True) as validation result. For the second rule, the first record passes the check since age equals 36 and the person has a job which is allowed by the rule. In the second case the job status is not available (NA). Hence the rule cannot

be checked and the returned value is NA as well. Finally, in the third record there is an 11-year old with a job which is a combination that is not allowed by the rule, yielding 0 (False).

3. One observation from this example is that the validation of the dataset of three records against two rules results in six validation results of either 0, 1 or NA. structure as well. A second observation is that we need the value NA as possible outcome of a validation. To support situations such as in this example where data can not always be validated we added not only the values $\{0, 1\}$ but also NA as a possible outcome of a validation result. Note that it is always possible for users of the report to afterwards interpret NA as False or True, depending on circumstances.

4. This simple example was based on in-record rules, yielding precisely one outcome per record. However we also need to support reporting on other classes of validation rules, including rules that result in a single value for the whole data set. For example, we may have a rule that states that the fraction of unemployment for people over 15 years old must be below 30%. In our generic report structure, a validation process is therefore seen as composed of possibly many fundamental validation events. In each event, a data set is confronted with a single rule yielding a single outcome. In the example of Table 1, there are six events. In the example on unemployment, there is a single event. All these results should fit in the validation report in an identifiable way.

5. Roughly speaking, the report describes validation events (who, what, when), the rule being evaluated, the data it was evaluated on, and the validation result. Schematically, an atomic record of the report can be represented as follows.

$$\begin{aligned}
\mathsf{validation} &:= \langle \mathsf{id}, \mathsf{type} = \texttt{"validation"}, \mathsf{event}, \mathsf{rule}, \mathsf{data}, \mathsf{value} \rangle \\
\mathsf{event} &:= \langle \mathsf{time}, \mathsf{actor}, \mathit{agent}, \mathit{trigger} \rangle \\
\mathsf{rule} &:= \langle \mathsf{language}, \mathsf{expression}, \mathsf{severity}, \mathit{status}, \mathit{decription} \rangle \\
\mathsf{data} &:= \langle \mathsf{source}, \mathsf{target}, \mathit{description} \rangle,
\end{aligned}$$

Here id is a unique identifier for the record, type is used to separate *validation* atomic records from record *aggregates* (recall Demand 3). The record contains three data items. The first is event, which contains data on the time of evaluation and the actor (person or software) performing the evaluation. Slots in *italics* are optional and will not be described here. The second element concerns the rule. The report describes the language in which the rule is expressed, including version if necessary, the rule expression itself, and the severity level. The severity level aligns with the ESS validation business architecture [ESS, 2017] where rules are assigned an *information*, *warning*, or *error* severity level. The third item is called data, and describes the data used to evaluate the rule (source) as well as the data under scrutiny (target). The distinction is made for cases where a large dataset is used to essentially evaluate a single value, for example when comparing a single value with the mean of a column. To identify data items, we align with the Handbook on validation [Di Zio et al., 2015], and advise to specify four elements to identify data: the *population* under scrutiny, the *measurement event*, the *element of the population*, and the *variable* being measured. We reiterate that the generic report is extensible. One may add additional fields if necessary or useful and this is probably very useful to add identifiers to the data in the local system to the data part of the validation report. Finally, the fourth element is the validation value, one of 0, 1 or NA.

6. The above description serves as a quick reference data structure, independent of the implementation language. The technical format for exchanging data validation reports is in principle to choose freely, but we felt it is useful to make a default choice based on existing popular standards. There are several common standards allowing for implementation of structured data exchange, including textual formats such as XML [W3C consortium, 2008–2013], YAML [Ben-Kiki et al., 2001–2009], JSON

```
{
  "event": {
    "time": "20170518T105055+02",
    "actor": "R 3.4.0",
    "agent": null,
    "trigger": null
  },
  "rule": {
    "language": "R pkg validate 0.1.7",
    "expression": "income >= 0",
    "severity": "error",
    "description": "total income must be non-negative"
  },
  "data": [
    "Dutch inhabitants",
    "Household survey 2017",
    "8237193679",
    "Household Income"
  ],
  "value": "1"
}
```


FIGURE 4. Example of a machine-readable validation report.

[ECMA, 2013] and popular binary formats such as protobuf [Google, 2008–2017]. In principle, any of these formats can be used to serialize the validation and aggregation data structures that were defined in the previous subsection. Here, we use the JSON format because of simplicity, wide support in many languages, and the availability of a schema definition language [Galiegue et al., 2013]. Moreover, JSON strings parse straight into javascript objects, facilitating further processing and visualisation, for example in the popular `d3.js` framework [Bostock et al., 2011]. The JSON scheme is available from GitHub: https://github.com/data-cleaning/ValidatReport.

7. Figure 4 shows an example of the machine readable JSON report describing the result of one validation of the rule *income is large or equal to 0* with a severity level of *error*, executed by the *R package validate* on *18th May 2017*. The data being checked is identified by the 4 elements described above where the population element contains an ID (8237193679). This makes it possible to link the validation report to the record in a local system. The validation result is positive (1).

V. Applications

1. Inspecting the output of a validation run can be difficult. Especially if the validation involves multiple rules and possibly many different datasets the output might grow huge. This is no problem for the machine-readable validation report as it uses an efficient format, for a user however, inspecting the output and deciding on the data to be edited might be cumbersome. For this reason we created two implementations where the technical validation report structure is presented in a human-readable format.

2. The first implementation is an interactive validation dashboard, presented in Figure 5. This web-based application shows aggregated statistics on the validation events in the validation report and a number of individual validation events. The example shown here is based on synthetic data. This data was generated based on 18 validation rules from the ESSnet Validation Foundation project [ESS, 2015], which in their turn were derived from an analysis of validation systems in Germany, the

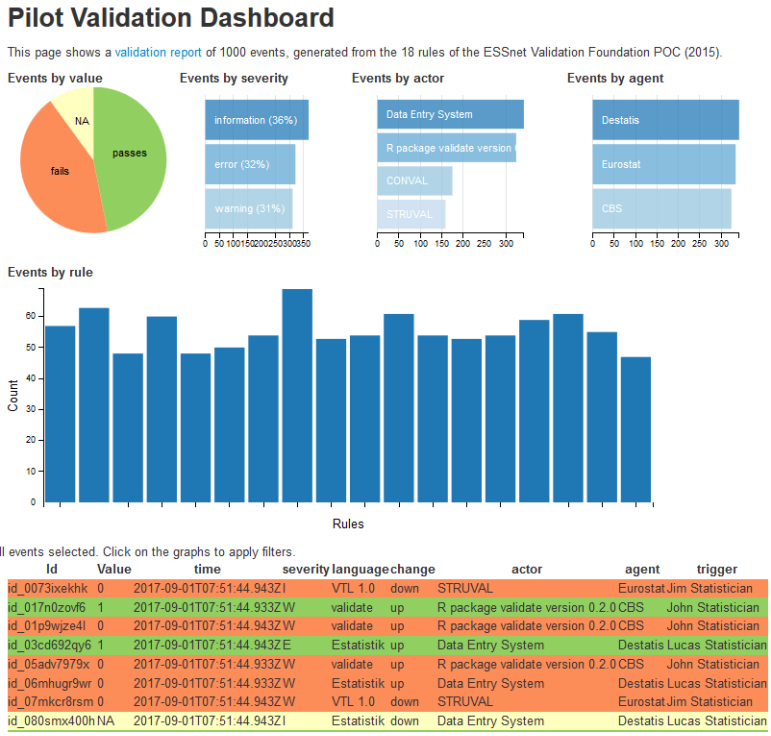

FIGURE 5. Validation dashboard based on crossfilter.

Netherlands and Eurostat. The dashboard shows a pie chart summarizing the number of passes (green), fails (red) and NAs (yellow) in the validation report. Next, are three bar charts showing summaries of events by severity, actor and agent. This is an example, the bar charts can instead show statistics on other variables from the validation report. The values shown depend on the selections the user makes interactively. On the second line the distribution of validation events per validation rule is shown. At the bottom part we see a number of validation events.

3. Each chart in the dashboard operates as a filter that is activated by clicking on it. Figure 6 shows the dashboard where *fails* of severity *information* generated by the *R package validate* are selected. Clearly there is another distribution of validation rules displayed. This way a user can drill down to the validation events of interest. If the validation report contains identifiers to the data involved, the dashboard could use the identifiers to guide the user directly to the right record in the data in an editing application. The dashboard itself is based on the dimensional charting library (`dc.js`) which works together with the crossfilter library for fast n-dimensional filtering and grouping of records (`crossfilter.js`).

4. Besides the interactive dashboard, we created a pilot implementation for static reports. In this case, the JSON technical report format is translated into a variant of markdown [Gruber, 2014]. Markdown is a text format that is aimed to be simple enough to be read and typed by humans, but strict enough to be automatically translated into attractive documents. An advantage of markdown is that by now it can be translated into many formats automatically, including pdf, HTML and Word, using freely available rendering software such as pandoc [Pandoc Development Team, 2016]. There is also extensive integration with popular tools including Python and R. The translating tool has been implemented in an R package called validatereport van der Loo and de Jonge [2018] which is currently available as a proof of concept on GitHub. At the moment this package is capable of reading and

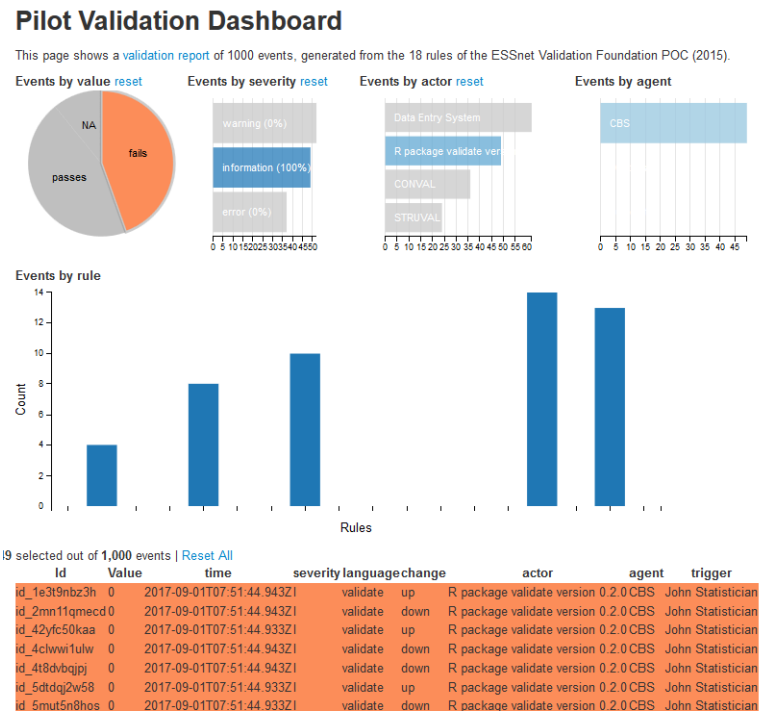

FIGURE 6. Validation dashboard with active filters.

writing the generic report structure and translating it into markdown. A typical example of markdown output may look like this (`[snip]` indicates we left a bit of code out for compactness).

```
- Event id        : 00175
      - type      : validation
      - actor     : R version 3.4.4 (2018-03-15) [snip] running on x86_64-pc-linux-gnu
      - agent     : Olav
      - trigger   : Task2_Metrics validation
- Value           : FALSE
- Source data     : 25
- Target data     : 25 [snip]
- Rule           :
    - language    : R package validate 0.2.0
    - severity    : error [snip]
- Rule expression:
```
```
   !(Age > 65) | (Working_hours = 0)
```
```

Using freely available tools such as pandoc, the already readable markdown format can be converted for example into HTML. An example is shown in Figure 7.

## VI. Conclusions

1. This paper highlights the design and use of a validation report, designed in the ESSnet ValiDat Integration. Based on the demands of *Identification*, *Closure under combination* and *Closure under aggregation* and a study of example validation reports from multiple countries a generic machine
```

FIGURE 7. Markdown, rendered to plain HTML, as shown by a browser.

readable format for validation reports was designed [van der Loo and ten Bosch, 2017]. This design is independent form the statistical domain, the validation language or the validation tool being used. The design is language-independent, but a full specification in the popular cross-platform JSON format has been developed in the form of a JSON schema.

2. The generic validation report has been implemented in an R-package *validatereport*. It works together with the popular R-package for data validation *validate* and is available on GitHub.

3. We also demonstrated that human-readable versions of the validation report can be generated automatically, either in the form of an interactive dashboard, or in the form of a static report in any popular format.

4. Although we labeled the current version of the validation report version 1.0.0 that does not mean it should not change any more. Successful standards evolve by using and refining them and we think and hope that it is the same here. The use of the generic validation report and related pilot software will generate new needs and it would be good to use a collaborative development strategy to extend and refine them. We encourage colleagues in any statistical institute to use the generic validation report in their validation processes and to (re-)use the pilot software such as the dashboard and translator(s) for their own purposes. In 2018 and 2019 Statistics Netherlands will further improve the R packages Validate, ValidatReport, and the other related software. Other statistical institutes are invited to join us in this adventure.

Acknowledgements

The authors are grateful to Tjalling Gelsema, Edwin de Jonge, Dick Windmeijer, and the partners of the ESSnet Validat Integration for comments and suggestions.

References

Oren Ben-Kiki, Clark Evans, and Ingy döt Net. *YAML Specification Index*, 2001–2009. website.

Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D3: Data-driven documents. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2011. website.

M Di Zio, N Fursova, T Gelsema, S Gießing, U Guarnera, J Ptrauskienė, L Quensel-von Kalben, M Scanu, K ten Bosch, M van der Loo, and K Walsdorfe. Methodology for data validation. Technical Report Deliverable No. 11/2, ESSNet on validation, 2015. pdf.

ECMA. The JSON data interchange format. Technical Report ECMA-404, ECMA International, 2013. pdf.

ESS, 2015. See https://github.com/data-cleaning/ValidatPoC for the rule sets.

ESS. Business architecture for ESS validation. Technical report, European Statistical System, 2017. pdf.

ESS. Essnet validatintegration, 2017.

Francis Galiegue, Kris Zyp, et al. Json schema: Core definitions and terminology. *Internet Engineering Task Force (IETF)*, page 32, 2013. json-schema.org.

Google. *Protocol buffers*, 2008–2017. website.

John Gruber. *Markdown*, 2014. http://www.daringfireball.net/markdown.

Pandoc Development Team. *Pandoc: the swiss-army knife for converting files from one markup format into another.*, 2016. URL http://pandoc.org.

Mark van der Loo and Edwin de Jonge. *validatereport: Report on Data Validation Results*, 2018. URL https://github.com/data-cleaning/validatereport. R package version 0.0.2.

Mark van der Loo and Olav ten Bosch. Design of a generic machine-readable validation report structure. Technical report, Statistics Netherlands, 2017. pdf.

W3C consortium. *XML*, 2008–2013. website.