A Feasibility Study of an Agile and Data-Centric Method for Prototyping Services Based on Open Transport Data

Nicolas Ferry(✉), Aida Omerovic, and Marit Kjøsnes Natvig

SINTEF, Trondheim, Norway
{nicolas.ferry,aida.omerovic,Marit.K.Natvig}@sintef.no

Abstract. Data under open licenses and in reusable formats, often referred to as "open data", is increasingly being made accessible by both public and private actors. Government institutions, municipalities, private companies and entrepreneurs are among the stakeholders either having visions of new open data-based services, or just looking for new ideas on potential innovations based on open data. It is, however, in both cases, often unclear to the service developers how the open data actually can be utilized. A main reason is that the data needs to be retrieved from multiple sources, understood, quality checked and processed. While gaining insights on possible services that can be created on the top of open data, a service developer has to undergo an iterative "trying and failing" exercise of service prototyping. In order to be practically feasible, such a process needs to be agile and efficient. Open data from the transport sector is in this study particularly focused on and used as a case. The open transport data are characterized by many challenges common for open data in general, but also a few specific ones. One of those challenges is the need for combining (often real-time) data from rather many sources in order to create a new service. This paper is an extension of our earlier research, which introduced a novel data-centric and agile approach to early service prototyping based on open transport data. In particular, we present a refinement of the initial approach and its evaluation in a significantly extended trial and discussion about the lessons learned from it.

Keywords: Service prototyping · Open transport data · DevOps

1 Introduction

During the past several years, increasingly many private and public actors all over the world have been actively releasing data under open licenses and often in reusable formats [2]. The goal is to foster creation of new and innovative digital services. The innovation and economic potential is becoming more and more visible, as documented by a European study [4], thus attracting governments, municipalities, companies and entrepreneurs to take part in the ecosystem of the data provision and creation of innovations on the top of open data. Once the data are released and announced through a public catalogue, a developer needs to understand its format and content, evaluate its quality and then (at least partially) create a new service through several iterations. This

D. Ferguson et al. (Eds.): CLOSER 2019, CCIS 1218, pp. 87–100, 2020.
https://doi.org/10.1007/978-3-030-49432-2_5

process is necessary in order to try out the ideas and evaluate feasibility of the envisioned service. Such a creative process of "trying and failing" to develop new services needs to be highly agile and efficient. The process is however slowed down since the data openly available online frequently consist of rather unstructured information [10], which makes service prototyping difficult and expensive [16]. It is also a challenge that the quality of the dataset descriptions and the meta data announced might not be good enough to give the developer with the information needed [3,12].

This paper is an extension of our earlier research [6], which introduced a novel data-centric and agile approach to early service prototyping based on open transport data. In particular, we present a refinement of the initial approach and its evaluation in a significantly extended trial and discussion about the lessons learned from it

The approach is novel in the sense that it is data-centric and focuses on how to develop an idea into a prototype rather than how to implement a solution. The approach is motivated by the identified challenges as well as experiences gained from the "Open Transport Data"[1] (OTD) research and innovation project, as well as from applying the data which has been harvested into an open catalogue by the project. We exemplify our approach on an open transport data service and discuss the lessons learned so far. We also outline a roadmap for the forthcoming research towards a comprehensive approach for agile prototyping of open transport data-based services.

Section 2 discusses the challenges related to the use of open data. Section 3 gives an overview of the approach. Section 4 reports on trial of the approach conducted by prototyping a service based on real-life open transport data, and Sect. 5 summarizes the related works, the lessons learned in this trial, and discusses the threats to validity of the results. We also propose the priorities for future work which aims to provide a comprehensive approach for agile prototyping of open transport data-based services.

The work builds upon and is an extension of our earlier research [6]. Parts of this paper are therefore re-used from the publication of the previous research, in order to ensure completeness and readability of this publication. Main extensions of this paper include: (i) a description of how we identified the main challenges that a method for service prototyping based on open data should address (Sect. 2), (ii) an extension of the state of the art (see Sect. 5), (iii) a refinement of the initial approach (see Sect. 3), (iv) a significant extension of the trial (see Sect. 4), and (v) a detailed elaboration of the experiences and lessons learned from the trial (see Sect. 5).

2 Challenges

We have through the above mentioned OTD project, which gathers some of the major public and private actors from the transport sector in Norway, addressed service prototyping in the context of open data from the transport domain. The project has conducted several use cases, and the insights from those were used to design a survey and semi-structured interviews, in order to identify the main challenges that a method for service prototyping should address. The survey was distributed through several channels, the main of which were the network members of "Intelligent Transport Systems"

[1] https://www.sintef.no/en/open-transport-data/.

(ITS) Norway (gathering organisations and companies in the transport sector); a meetup group on open data in Oslo region in Norway; participants at hackathons; and the networks of governmental organisations providing open data, among others the Norwegian Public Road Administration and the Norwegian Mapping Authority. Google forms were used for all channels except in interaction with the participants at hackathons. During the hackatons, the respondents received a paper version of the survey.

To get more in-depth details on the use of open data and related challenges, the overall questions asked that triggered the responses were:

1. What is your background with respect to education?
2. What is your experience with use of open data?
3. Which data have you used?
4. Which data have you searched for, but not been able to find?
5. What are the most important problems you have experienced regarding use of open data?
6. How has the use of open data influenced product and functionality ideas?

In addition, the researchers also carried out semi structured interviews with participants at the hackathons to enable the respondents to provide additional information. During the interviews, one researcher asked the questions while another one observed, recorded the interview and addressed missing issues. Some of the interviews were also carried out via telephone upon agreements at the hackathons.

The following list includes the main challenges that have been identified, *i.e.*, the challenges that a developer faces when prototyping services on the top of open data:

- Discovery of relevant datasets through metadata search and visualisation of datasets to better understand the data content. Public catalogues and data portals are still not comprehensive and metadata for describing the contents are only to a limited degree standardized and available.
- Understanding and using varying application programming interfaces (APIs) for data retrieval. Even though API description standards exist (*e.g.*, OpenAPI), they are not commonly used, and APIs are not documented in a standardised way.
- Combining multiple sources of open data, in order to create value added services. Travel planners will for example need information on addresses, stop points, route plans and position data from several transport service operators, maps, etc.
- Accessing real-time data from IoT and sensors. The amount of such data will increase, and new services will use real-time data streams on, for example, the conditions at locations and the movement of people, vehicles and goods.
- Handling of large volumes of data, which is possibly unstructured.
- Handling proprietary data formats. For example, standards exist for data on public transport, but for other transport types (*e.g.*, car sharing, city bikes, ride sharing) there are no standards, and proprietary data formats are used.
- Understanding the data. In many cases, domain knowledge is required in order to sufficiently understand the data contents. This is a challenge due to lack of documentation and metadata, as described above.

Clearly, these characteristics impose requirements to the approach followed for prototyping the services based on open transport data. Our goal is that a service developer

(*e.g.*, an entrepreneur with limited programming background) can incrementally explore the possibilities and ideas while creating a service prototype. To that end, the approach has to be highly iterative, comprehensible to non-expert developers and cost-efficient. To the best of our knowledge regarding state of the art (summarized in Sect. 3), there is currently no approach which sufficiently meets the above mentioned needs and challenges. In particular, the existing approaches fail to be sufficiently agile, scalable and comprehensible in order to fit for gradual prototyping through consolidation of many data sources through multiple iterations.

3 Overview of the Approach

In this section we introduce our approach for the iterative prototyping of services based on open data. We propose a seven-step prototyping process for the development of services based on open transport data, as depicted in Fig. 1.

Fig. 1. Data oriented early prototyping process [6].

In the following we describe the details of each of the steps depicted in Figure 1. Some of the steps have been extended with additional information compared to [6].

1. **Search Data:** The developer needs to identify the data sources and datasets which the forthcoming prototyping iteration will be based upon. Catalogue, data repositories, and search engines can help finding the relevant datasets. When datasets are found, the developer wants to quickly judge the relevance of the dataset. However, this task can be tedious as datasets typically lack proper description and meta-data

and since open data is typically released with terminologies and structures dependent on the domain they originate from. Due to its open nature, the data is not prepared for a specific application and can be used in many different contexts which were not necessarily anticipated at release time. In addition, as stated in [13]: "*it can be difficult to determine not only the source of the dataset that has the information that you are looking for, but also the veracity or provenance of that information*".

2. **Access and Understand Data:** Once found, the data needs to be accessed and understood by the developers. For this, they typically need to manipulate and test the data. Indeed, in many cases, only looking at the documentation of the data (when available) is not enough, as documentation typically fails to represent aspects such as the missing data, data accuracy, etc. This process consists first in understanding how the identified datasets or data streams can be accessed, second in actually accessing the data, and finally in looking at different samples of the data in order to properly understand its contents, structure, etc. These activities are often done in an ad-hoc manner as the APIs to retrieve data are typically not following API description standards.
3. **Identify Added Value:** From this stage, the developer can identify the potential usage area for the data that will enable the creation of new added value services. This step requires looking into the details of the data in order to understand its contents and to identify which parts of it are relevant for our service. It is important at this stage to evaluate several samples of data in order to establish the overall quality of the data - *e.g.*, data accuracy and the missing data.
4. **Specify Capabilities:** At this stage, the developer can start specifying the features that will be offered by the prototype. This activity will be affected by the availability of data and its identified added value.
5. **Prepare Data:** Once the capabilities of the service are identified, and before its implementation, the developers need to manage and prepare the data for further analysis and processing as part of the service business logic. This includes the following activities: data characterization, data organization, data filtering, restructuring and compression. At the end of this stage, the data should be ready to be consumed by the business logic of the service. In addition, it should fit its needs and requirements.
6. **Prototype Service:** This stage consists in the actual development, delivery and deployment of a prototype that implements the business logic of the service specified at step 4.
7. **Identify Missing Data:** At the end of a prototyping iteration, once a new set of features have been added, the developer identifies which features should be added to the prototype in the forthcoming iteration, as well as which data are required.

In case additional data is required to deliver the service with the desired capabilities, developers can enter a new iteration of the prototyping process. If not, the prototype can then be used in other stages of the product life cycle such as code and deployment stages, for instance when part of its implementation needs to be re-developed to meet the production requirements (*e.g.*, specific framework needs to be used), or to the testing stage.

4 Trial of the Approach

We tried out our approach in the context of the OTD project, where we developed a service aiming at *(i)* supporting user (the citizens) in planning public transport trips in Oslo, as well as *(ii)* counting all the ongoing deviations within the public transport (*e.g.*, tram delays, problems with a bus). More precisely, this service first loads a map of Oslo and displays all the stops in the city. A user can then plan a travel by clicking on two of these stops. A route, including details about the stops, is then proposed to the user as a path on the map (see Fig. 2). In addition, the service displays statistics about the number of deviations in the city. The scope of the trial were open data available for the public transportation within the city of Oslo, Norway.

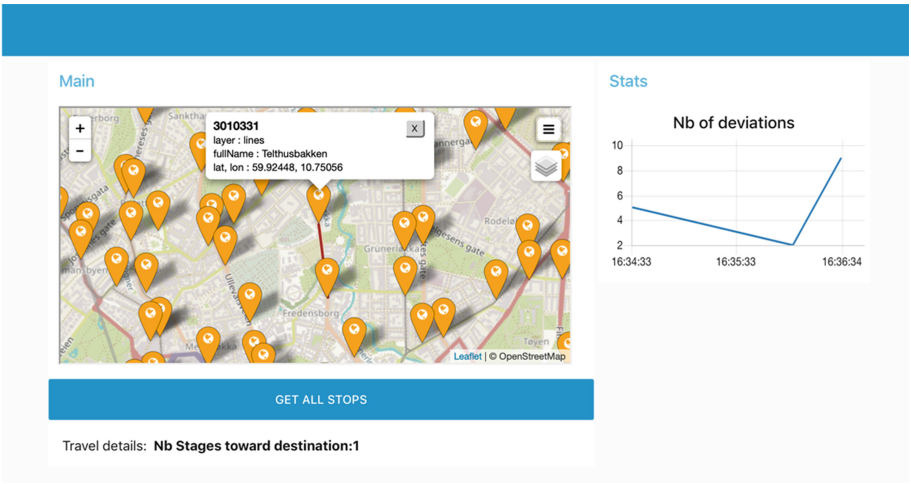

Fig. 2. A Simple Travel Planner as a trial of the approach.

During the trial we instantiated all the steps of the approach presented in Sect. 3. In the following, as described in [6], we recall the activities we performed in each of the steps.

Search Data. We first searched for data in the Open Transport Data CKAN catalogue[2] (see Fig. 3) using "transport" and "Oslo" as keywords but we could not find relevant data. By contrast, when using the "Ruter" keyword (Ruter is the public transport authority for Oslo), we found the API of a "route planning" service.

Access and Understand Data. We first selected the Ruter Sirisx API[3] which allowed us to retrieve, for one stop (*i.e.*, buses, tram, and subway stops), the list of ongoing

[2] http://78.91.98.234:5000/.
[3] https://sirisx.ruter.no.

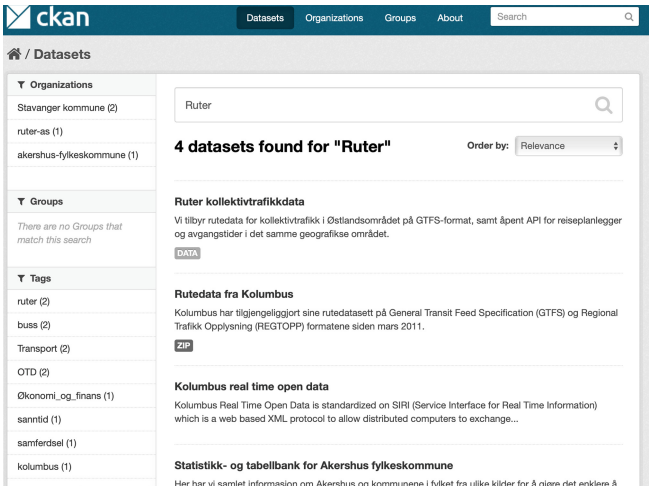

Fig. 3. Open Transport Data CKAN [6].

deviations in all the lines using this stop. It is exposed as a REST API and can be accessed using classical tools such as "curl" or a "REST console". However, the API is little documented and we identified that we could not use exactly this service as it requires a JSON object containing the identifier of the stop of interest, as input. We thus searched again in the catalogue for another API providing such information, and we selected the Ruter Reise API[4] as it provides details about all the public transportation stops in Oslo, regardless of the transportation mode. We verified that the information between the two services was matching semantically - *i.e.*, we stored identifiers of a few stops from the Ruter Reise API service and thereafter we called the Sirisx API using these identifiers.

Identify Added Value and Specify Capabilities. We analyzed the data from both the Ruter Reise and the Sirisx APIs. We could easily find the relevant information and in general the data was accurate even though the textual description of a deviation was sometimes incomplete or missing. Using these APIs we could retrieve and provide users with live information about the deviations associated to one or several stops. We also decided to retrieve and store this information on a regular basis to compute the average number of deviations over a week in the whole city.

Prepare Data. We prepared the data in two ways. First, by filtering it to only manipulate the part relevant for our service. Second, we prepared the data for further analysis. The data from the Reise API describing the stops was obtained in the form of a JSON object stringified. Unfortunately, the JSON obtained was not properly formatted as it used single quotes instead of doubles. In addition, some Norwegian language

[4] https://reisapi.ruter.no/help.

characters where not properly encoded. We thus implemented a mechanism to fix this issue before transforming the string into a proper JSON object.

Prototype Service. We implemented our service using the Node-RED platform[5], an open source project by IBM that uses a visual dataflow programming model for building applications and services. Using Node-RED, an application takes the form of a set of *nodes* (*i.e.*, software components) wired with *links* that are encapsulated in a *flow*. A flow can easily be exposed as a service using specific Node-RED nodes. Thanks to the large community behind Node-RED, a large set of nodes are available off-the-shelf and for free, making it rather easy to implement new applications and services. We had to implement specific nodes for accessing the two APIs and for computing the average number of deviation over a week[6]. The final flow is depicted in Fig. 4.

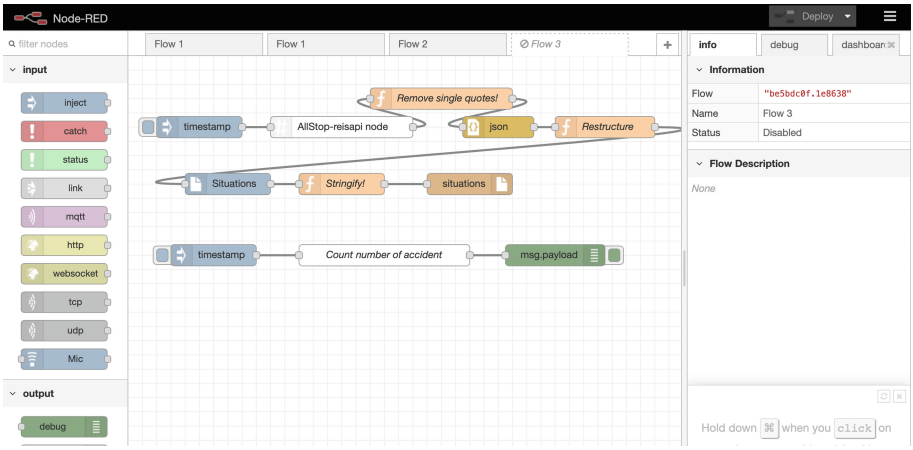

Fig. 4. Data preparation using Node-RED [6].

Identify Missing Data. We did not find it necessary to implement this step in the trial, as the prototype already covered the intended functionality.

5 Discussion and Related Work

This section briefly puts our work in the context of most essential related work in general, namely Agile Software Development. This is followed by a detailed discussion of the lessons learned from the trial, including how the specific steps relate to the related work which is particularly relevant to them. Finally, an elaboration on threats to validity and reliability, is provided.

[5] https://nodered.org.
[6] https://github.com/SINTEF-9012/OTD-components.

5.1 Related Work in General

The principle of iterative and incremental software development has already been advocated for many years by the Agile Software Development (ASD) manifesto and its principles. The ASD cycle [5] relies on the following six stages: plan, design, develop, test, review and release. Our approach is inspired by the ASD methodology and aims to be integrated as a part of the ASD process. It focuses on prototyping and aims to be used in the plan stage where a prototype would be used to prove feasibility of the service and to discuss future development activities. It could also be used in the design stage in order to understand and discuss the features to be offered, as well as in the development stage of the ASD process.

More recently, the DevOps principles are being widely adopted by the software industry. DevOps advocates a set of software engineering best practices and tools, to ensure Quality of Service whilst continuously evolving complex systems and foster agility, rapid innovation cycles, and ease of use [8]. In particular, DevOps put a lot of emphasis on automation and collaboration between development and operation activities with continuous feedback between Dev and Ops. The DevOps infinite loop consists of the following stages: plan, code, test, deploy, operate, and monitor[7]. As for the Agile methodology, our approach could take place as part of a DevOps process either in the planning or coding stages.

In 2018, Gartner introduced DataOps in the Hype Cycle for Data Management[8]. At the moment, DataOps, which is inspired by the DevOps movement, is still in its infancy. It strives to speed the design, implementation, and production of data processing and analytics applications. Similar to our approach, data is a first class concern in DataOps. However, it is mainly focusing on big data applications.

5.2 Lessons Learned from the Trial

This section summarizes the challenges we faced during the trial and the lessons learned with respect to each step of the approach.

Search Data. During our trial, we first observed that many catalogues of datasets (and data sources) are available on the web through data portals, but it was difficult to make sure that we were using the best candidate. Data portals leverage data catalogue systems to store, publish and discover datasets. CKAN, DKAN, and Socrata are amongst the most famous data catalogue systems used by data portals. CKAN[9] is an open-source data catalogue system that supports the publication, sharing, search and management of datasets in a domain-independent way. CKAN exposes a powerful RESTful JSON API to manage data catalogues. DKAN[10] offers similar features but it is based on Drupal whilst Socrata[11] is a commercial platform. In the context of our Open Transport Data

[7] Please note that the terminology and the number of stages change from one source to another.
[8] https://www.gartner.com/en/newsroom/press-releases/2018-09-11-gartner-hype-cycle-for-data-management-positions-three-technologies-in-the-innovation-trigger-phase-in-2018.
[9] http://ckan.org.
[10] https://getdkan.org.
[11] http://socrata.com.

project, CKAN has been adopted, as it is the solution powering major data portals such as the European data portal[12]. In terms of tooling, we observed that there may be a need for a cross-catalogue search engine (*i.e.*, an engine enabling searching on multiple catalogues).

In addition, as already presented in Sect. 3, searching the most relevant datasets or data sources for building a specific service is challenging due to the lack of metadata about *(i)* the datasets (or data sources) and *(ii)* the semantic overlaps between different datasets (or data sources). For example, it would be interesting to link datasets by means of automatic annotations with keywords that would form a domain specific ontology [9].

Similarly, once we selected our datasets or data sources, it was impossible to assess if these were the best candidates. However, in this case, it is worth noting that our agile approach, where we can start over again after trying to use the dataset, helps assessing the quality and value of different data sources.

Understand Data and Identify Value. Identifying the value of the datasets is also challenging as it can be difficult to evaluate the quality of the data. For instance, when dealing with large datasets or data streams, it is difficult to identify if some data is missing. As an example, in a large dataset with data recorded every second for a few months, it might be difficult to check if a few days or hours of recordings are missing. More generally, information about the reliability of a data source is typically not provided.

Prepare Data. The preparation of the data does not necessarily involve complicated tasks. Some tools and methods facilitating manipulation of open data do exist. For instance, the Linked Data Stack [1] is a software stack consisting of a number of loosely coupled tools, each capable of performing certain sets of operations on linked data, such as data extraction, storage, querying, linking, classification, and search. The LinDA project [7] developed a set of tools for linked data publishing, packaged into the LinDA Workbench. In the cases of both Linked Data Stack and LinDA, the complexity of provisioning resources and managing the web application rests on the service developer who must install the tools and maintain the infrastructure. The COMSODE project [14] provided a set of software tools and methodology for open data processing and publishing. COSMODE is not available as an online service, but rather as a set of tools that need to be individually managed, which implies additional burden on the developer. Datalift [17] is a software framework for linked data publishing. It is considered as an "expert tool" [17]. For example, it comes with no GUI to support data publishers in the data publication process. The Linked Data AppStore [15] is a Software-as-a-Service platform prototype for data integration on the web. Common for the mentioned tools and approaches is that they either only partially cover the prototyping process, or that they are too extensive and therefore unfit for a DevOps-driven agile approach. After a few steps of manipulation, it can be difficult to actually understand the status of the data being manipulated (*i.e.*, structure, format, or even the actual content of the data). In such a case, tools providing a means to visualize the data after each manipulation, would be highly beneficial. This applies not only to datasets but also to data streams.

[12] https://www.europeandataportal.eu.

Prototype Service. Our approach is meant to be used during the prototyping phase of the overall life-cycle management of a service. However, it appears that this prototyping phase, by itself, would benefit from using classical tools for the continuous and agile development and operation of services. For instance, once a prototype has been implemented, it typically has to be deployed and tested in an sandbox environment. Similarly, more advanced prototypes could undergo a canary testing - *i.e.*, routing a subset of users or requests to the prototype. A deep analysis of how our approach fits within the main Agile and DevOps processes, is required.

Moreover, the migration from a prototype to a service in production is challenging. In particular, the service will most likely need to be re-implemented using tools, languages, and frameworks adapted for production. For instance, a service that consumes data streams will rely on a stream processing framework. Such frameworks are typically designed for continuously processing data in real-time. The most prominent stream processing frameworks such as Apache Storm[13], Apache Flink[14], Heron [11] and Apache Spark[15], usually rely on the concepts of: data sources (*i.e.*, the entity producing the data), events (*i.e.*, the abstractions that encapsulate the data from the data source), data streams (*i.e.*, sequences of events), processing components (*i.e.*, the entities responsible for actually performing operations on the data streams), and data flows (*i.e.*, orchestrations/topologies of data streams and processing components). Even though prototyping platforms such as Node-RED, to some extent, share common concepts with these frameworks, the migration from one to another is not straightforward. To the best of our knowledge, there are no tools supporting such migration. This applies not only to the implementation of the service itself but also to the deployment, installation and configuration of the framework. For instance, in order to parallelize and distribute the processing activity, processing components and data streams are often executed on a cluster of machines managed by a coordination platform such as Zookeeper[16] or YARN[17]. This also applies for classical batch processing frameworks such as Hadoop[18].

5.3 Threats to Validity and Reliability

The validity of the results depends to a large extent on how well the threats to validity and reliability have been handled. This section discusses the essential aspects of such threats in our context. In the original study [6], we argued that several threats to validity and reliability were present. Majority of those threats also apply to this research, although the evaluation has been more comprehensive. In fact, the refinement of the approach and the extended evaluation have brought additional arguments regarding the presence of the validity threats. However, our recent results and extensions of the previous research have also partially addressed some of the weaknesses which were identified during the original study.

[13] http://storm.apache.org.
[14] https://flink.apache.org.
[15] https://spark.apache.org.
[16] https://zookeeper.apache.org.
[17] https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html.
[18] https://hadoop.apache.org.

In terms of validity, our trial is only to a limited degree representative for the contexts intended to be within the scope of our approach. The service prototyped has a specific, rather limited, functionality based on few data sets and involving only fictitious end-users. In a realistic setting, the functionality may have been far more comprehensive, relying on several larger, more different and distributed data sets. The quality of the data sets may also be at widely different levels. In addition, the number of the end users and the frequency of their requests may be far higher then what was the case in our trial – this may have impacted scalability and performance of the solution. None of these properties of the context were present in our trial, hence they have not been tested. A realistic setting would also involve prototyping and even integration of several services, thus introducing additional complexity that we did not cover in our evaluation.

The trial has, however, given strong indications of feasibility of the approach. No particular customizations of the approach (once the refined version was proposed and ready for evaluation) were needed for the trial. Thus, we have reason to believe that it should be possible to reapply our approach on new services.

Reliability is concerned with demonstrating that the empirical research can be repeated with the same results. Of course, a trial like the one we have conducted can not give solid repeatable evidence. There are several contextual factors influencing what happens, particularly the choices made by the researchers during the service development. As our main goal has been to propose a refined version of the approach and test its feasibility through the trial, performance evaluation of the approach itself was not addressed. Ideally, we should have exposes the method to several teams aiming to prototype both the same service as well as other services, under comparable and controlled conditions. Such a setting would provide more relevant evidence for reliability of the results.

It is, in terms of evaluation with respect to reliability, also a weakness that the researchers who tried out the approach also participated in design of the approach. As such, it is also a threat to reliability of the evaluation results, as we cannot know to what degree another service developer would have obtained the same results.

Hence, we do need to further evaluate the approach in more realistic settings. There is also a need for a baseline for comparing this approach with the alternative ones, in order to assess its characteristics such as usability, usefulness and cost-effectiveness. It should be a part of the future work. Further empirical evaluation is also needed for assessing scalability of our approach with respect to complexity and size of the services to be developed.

Overall, we have drawn useful experiences from developing and instantiating the approach in the example. Although the mentioned threats to validity and reliability are present in the study, we argue that the results indicate feasibility and suggest strengths and weaknesses of the approach.

6 Conclusions

This paper is an extension of our previous research [6] where an initial approach to early and continuous service prototyping based on open data, was proposed. The approach has been based on a the needs identified throughout the “Open Transport Data” research

and innovation project, and in particular the challenges identified through a survey and interviews. We have tried out feasibility of the approach on an open transport data service, and elaborated in detail on the results and the experiences. Main contributions of this paper include: refinement of the initial approach and refinement of the state of the art. Moreover, the trial has been significantly extended and presented in detail. We also provide a detailed elaboration of the experiences and lessons learned from the trial. In this paper we present an agile and data-centric approach for the early prototyping of services. The results of our feasibility study indicate the benefits and drawbacks of the approach. In particular, as main benefit, we argue that it fosters a "try and fail" development process where developers implementing services on top of open data can play, test, and understand the data while implementing a service. In future stage we will investigate how the approach could be seamlessly integrated in the overall development and operation process of a service.

Acknowledgement. This work has been funded by the Open Transport Data Project under Norwegian Research Council grant no. 257153, as well as by the H2020 programme under grant agreement no 780351 (ENACT).

References

1. Auer, S., et al.: Managing the life-cycle of linked data with the LOD2 stack. In: Cudré-Mauroux, P., et al. (eds.) ISWC 2012. LNCS, vol. 7650, pp. 1–16. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35173-0_1
2. Barometer, O.D.: Open data barometer global report. WWW Foundation (2015)
3. Beno, M., Figl, K., Umbrich, J., Polleres, A.: Open data hopes and fears: determining the barriers of open data. In: 2017 Conference for E-Democracy and Open Government (CeDEM), pp. 69–81. IEEE (2017)
4. Carrara, W., Chan, W., Fische, S., Steenbergen, E.V.: Creating value through open data: study on the impact of re-use of public data resources. European Commission (2015)
5. Cockburn, A.: Agile Software Development: The Cooperative Game. Pearson Education, London (2006)
6. Ferry, N., Omerovic, A., Natvig, M.K.: Towards early prototyping of services based on open transport data: a feasibility study. In: Proceedings of the 9th International Conference on Cloud Computing and Services Science, CLOSER 2019, Heraklion, Crete, Greece, 2–4 May 2019, pp. 257–262 (2019). https://doi.org/10.5220/0007675402570262
7. Hasapis, P., et al.: Business value creation from linked data analytics: the LinDA approach. In: 2014 Conference on eChallenges e-2014, pp. 1–10. IEEE (2014)
8. Humble, J., Farley, D.: Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Addison-Wesley Professional, Boston (2010)
9. Jiang, S., Hagelien, T.F., Natvig, M., Li, J.: Ontology-based semantic search for open government data. In: The proceedings of the IEEE 13th International Conference on Semantic Computing (ICSC), pp. 7–15. IEEE (2019)
10. Kim, G.H., Trimi, S., Chung, J.H.: Big-data applications in the government sector. Commun. ACM **57**(3), 78–85 (2014)
11. Kulkarni, S., et al.: Twitter heron: Stream processing at scale. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp. 239–250. ACM (2015)

12. Martin, S., Foulonneau, M., Turki, S., Ihadjadene, M.: Open data: barriers, risks and opportunities. In: Proceedings of the 13th European Conference on eGovernment (ECEG 2013), pp. 301–309. Academic Conferences and Publishing International Limited, Reading (2013)
13. Noy, N., Brickley, D.: Facilitating the discovery of public datasets (2017). https://ai.googleblog.com/2017/01/facilitating-discovery-of-public.html
14. Hanečák, P., Krchnavý, S.I.H.: COMSODE publication platform - open data node - final. Technical report, July 2015
15. Roman, D., et al.: The linked data AppStore. In: Prasath, R., O'Reilly, P., Kathirvalavakumar, T. (eds.) MIKE 2014. LNCS (LNAI), vol. 8891, pp. 382–396. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-13817-6_37
16. Rusu, O., et al.: Converting unstructured and semi-structured data into knowledge. In: 2013 11th Roedunet International Conference (RoEduNet), pp. 1–4. IEEE (2013)
17. Scharffe, F., et al.: Enabling linked data publication with the Datalift platform. In: AAAI Workshop on Semantic Cities (2012)