When is Data Ready for Data-driven Science?

Giridhar Manepalli, Suchitra Manepalli
gmanepalli, smanepalli@cnri.reston.va.us
Copyright © 2021. Corporation for National Research Initiatives

Abstract

Science is increasingly driving medical, economic, and geopolitical policies and practices. It is therefore critical to ensure data used in scientific research is ready for use. While we assume research efforts consider the accuracy, completeness, and other quality attributes of data prior to its use, we draw attention to the possibility that data used in research is often derivative in nature; and the assumptions made in source data during its production are important factors to consider prior to the use of data in scientific research.

We hypothesize that there are specific domain-neutral questions, seven at the minimum, related to data that researchers should get answers to, in addition to any domain-specific and use-case-specific considerations, to deem data ready for use. Only a subset of those questions is currently answered by systems in most cases. We discuss those seven questions, provide an analytical approach for answering them, and describe a prototype system that can augment information managed by existing systems to answer such questions.

1. Introduction

Data is often described as lifeblood, oil, or rising tide in various expressions to convey the importance of its contributory role in certain applications. Unintentionally, though, in those expressions, data is relegated to the status of a replaceable commodity. But data is a differentiated product by the very fact that each instance of data is different from other instances in terms of the information it represents.

That data is differentiated is sometimes acknowledged via ongoing efforts to define and apprise data quality and embrace FAIR principles [1]. However, data is not a product in isolation, but often a result of being derived from other data that probably originated elsewhere. There are no standard, domain neutral, practices that evaluate data for its merits and that takes evolution of the data into consideration.

We argue that an assessment of data readiness should not only involve inspection of the data at hand, but its forebears. To this end, readiness evaluations of data should consider elements such as its identity, time of inception, and dependencies. While it might be difficult for a given system to make such information about data to be incorporated for prior generations of data all the way to its origin, especially as data crosses several organizational boundaries, each data
management system should bear the responsibility of assessing the information for the portion of the data it is making accessible.

The approach described in this paper provides a pathway for derivative data to rely on ancillary information made available by its data source. This, in turn, relies on information made available by its data source, thus enabling the possibility to deduce a deep nested dependency graph with sufficient provenance information for each node in the graph, and aiding data quality evaluators to assess the merits and accuracy of data in a more substantive way than before. To this end, in the next few sections, we define and propose seven questions that, at the minimum, a data management system should meaningfully address for each unit of data it is making available.

To demonstrate the ideas presented in this paper, we have designed and developed an open source extension to CNRI’s Cordra software, in prototype form, called Collab. Collab software, in conjunction with Cordra, can be used as a standalone data management system. However, the intended use case for the integrated system (other than for demonstration purposes) is to augment existing data management systems with a capability to answer data readiness questions with minimal overhead. As will be described later in this paper, Collab software can be used to repeat and reproduce complex scientific workflows using just their workflow identifiers, even when the users are unaware of any input that the workflows need to begin. This is a design pattern that creators of scientific workflow systems will hopefully embrace.

The ideas presented in this paper are especially meaningful when data is synthesized from data that is sourced from external systems (to demonstrate the cross-organizational sharing of data). To this end, we drew inspiration from a specific data mashup scenario that we worked on several years ago, and simplified it for our demonstration here. The modified scenario is to be able to produce a dataset consisting of the average household income along with the number of retailers participating in a nutrition assistance program, for each county in a specified state. The idea is that these datasets would, hopefully, reveal if there are insufficient number of such retailers in a county who can assist impoverished households at times of emergencies.

Small software components have been developed by CNRI, and integrated with a CNRI Collab prototype instance, which would allow the generation of datasets of the aforementioned kind, for each such data source in the scenario. Such a dataset will be derived from data made available from public sites on the Internet. Collab captures and makes available data readiness responses for not only the final dataset generated for a given request, but for all intermediate snapshots of data with links going back between different snapshots to emphasize the derivative nature of the data.

Although there are several efforts that define the term “data quality” using objective qualitative and quantitative metrics, we are not aware of any effort that specifically addresses “data readiness” in a formal domain-neutral way, while keeping in mind the derivative nature of data.
This paper is likely to provide useful points of reference for scientific discourse on the topic of data readiness. We also anticipate this paper to influence policy makers and research sponsors in drawing up data management plans that emphasize “data readiness” to enable data that is fit for use in emerging artificial intelligence and automation capabilities to be used in ways that are likely to shape our society in a productive way.

The next sections discuss our key contribution, data readiness questions and responses, and prototype software, along with screenshots and links to the specific scenario described above.

2. Our Contribution

Our key contribution is bringing the attention of scientific community to the topic of data readiness in science, by way of:

- defining specific questions that researchers from different domains should ask about data to evaluate its readiness,
- defining three API calls that data consumers can benefit from if implemented by data management systems,
- making available in open source form to the public prototype software, called Collab, (which when integrated with Cordra software can augment existing data management systems) to answer data readiness questions, and
- describing a method in Collab for users to repeat and reproduce complex workflows using workflow identifiers, even though the underlying programs might need input that the users are unaware of.

The Collab Prototype software referenced above, and discussed in detail below, leverages a novel data model for capturing data quality claims, proofs, and pre-verifications. That data model is described in a different paper [4].

3. Data Readiness

Science builds on past discoveries. When science is data-driven, data that contributed to a discovery will have lasting effects on all subsequent discoveries that depend on it. This implies that arbitrary data cannot be used for science.

Data readiness, per our definition, is that property of data which removes arbitrariness from data by being transparent about its shape, inception, sourcing, quality, repeatability, accessibility, and cite-ability.

Specifically, we hypothesize, the answers to the following seven questions will provide researchers from any domain an opportunity to reason about data readiness:
1. What identifier(s) is the data associated with?

An identifier is a named reference to data, as allotted by a service. Multiple services may allot identifiers to the same data. The listing of those identifiers (and references to services that allotted those identifiers) that a data provider believes to remain relevant for long periods is considered an appropriate response to this question.

2. How to access data?

The method (or methods) to gain access to data based on its identifier is (are) expected to be stated here. This response enables downstream researchers an opportunity to access data that has been used for a given scientific discovery.

The network coordinates to access data may vary over time, and the response to this question should be adjusted when data moves, but the data that is returned by the access method should remain fixed. Preservation principles argue for format/type migration, but the resulting data from any migration exercise should be identified differently.

3. What is the data type?

Uninterpretable data is useless or even dangerous given the possibility of misinterpretation. The use of “Type”, per our definition, makes data interpretable. At its simplest, a Type could be just a MIME type that informs researchers which software applications can visualize the data for a human review. A Type could also describe ontology definitions that aid machines to reason about data using logic, or could inform the exact computational methods that can be applied on data to produce a synthesized form of data or visualization or explanation that is interpretable.

Which exact kind of Type should be used from the spectrum of aforementioned possibilities is considered out of scope. Collab uses a notion of Type that represents the schema to which the data conforms.

The Type identifier itself should remain fixed, but the referenced Type definition may evolve purely to add clarity.

4. When is the data known to exist?

The data provider should reference the earliest time when the data is known to exist by using a verifiable timestamp for this purpose. This timestamp informs downstream users about the data inception, relevancy, and versioning.
5. What is the data derived from?

Data is often synthesized and normalized for use in a context that is different from where it is generated. Downstream researchers should be able to know the source/original data to evaluate the data readiness. References to source data should be made. In cases where the system making available the source data does not associate answers to questions discussed in this paper, the source data should also be managed along with answers by parties that produced derivative data.

This response should remain fixed for as long as the data is made accessible.

6. How to reproduce the data?

In cases where data production is a result of a complex computation or derived from other data, data providers may enable users to repeat the process and regenerate the data. If repetition should result in the production of an identical version, then the workflow is said to be reproducible. Data providers should indicate, as part of their response to this question, whether or not the workflow can be repeated, and if so, how (by providing a reference to the corresponding user or programming interface). They may also indicate if the repetition is expected to result in an exact reproduction. Finally, they may also provide access to relevant source code and documentation to enable users to replicate the workflow environment for repetition or reproduction of data in a different environment.

This response may change over time, as long as the reason for that change is to embrace a newer implementation of the reproducibility enabling system.

7. What assertions are made about data by producers that consumers should consider prior to data use?

While answers to the above questions inform downstream users of data access, typing, and provenance information, there may be other attributes of data such as its accuracy and completeness as evaluated using criteria outside the scope of the conveyed data. Those quality attributes may also be conveyed by the data provider to downstream users (as a response to this question). We have defined a data model for how to convey such claims, and in fact how to convey proofs for those claims, without placing the burden on the user to verify those proofs. Collab software leverages that data model, but that data model itself is described in here [4].

The claims themselves can evolve over time, as long as the newer set of claims are based on the original data.
4. Collab Prototype

We define three Application Programming Interface (API) calls that we argue every data management system should enable for users to consume data in a way that is common across systems and domains; and we showcase those API calls by way of the scenario described in the Introduction section.

Here are the three calls:

1. Given a data identifier, the data in question is returned.
2. Given a data identifier, the data readiness answers are returned.
3. Given a type identifier, the type description or definition is returned.

Call 1 is expected to return the same information over time. Calls 2 and 3 may result in different responses over time, but the reason for such changes is not expected to violate the conditions discussed in the Data Readiness section.

Figure 1 illustrates the resulting data (for Call 1) from the scenario; it is an example workflow that pertains to the state of Virginia. (Many rows of data in the Figure are deleted for brevity; full data can be accessed by following this link: https://hdl.handle.net/20.5000.1080/a76b412c0a880dfb47b67651c55f1eb9?locatt=view:dataUI.

Table 1 illustrates the data readiness answers (for Call 2) for the same example. Those answers can be accessed (alternatively) by following this link: https://hdl.handle.net/20.5000.1080/a76b412c0a880dfb47b67651c55f1eb9?locatt=view:claimsUI.

Listing 1 illustrates the type description (a JSON schema along with some code) (in response to Call 3) that describes the data that resulted from Call 1. The type record can be accessed (alternatively) by following this link: https://hdl.handle.net/20.5000.1080/8fa5e2f6902357489c3393399df52842.
Figure 1: Partial Resulting Data from the Example Workflow

<table>
<thead>
<tr>
<th>Claim</th>
<th>Value</th>
<th>Proof Type</th>
<th>Verifier</th>
<th>Source code:</th>
<th>Public Key:</th>
<th>Public Key:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identifier</td>
<td>hdl: 20.5000.1080/a76b412c0a880dfb47b67651c55f1eb9</td>
<td>Attested</td>
<td>N/A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>hdl: 20.5000.1080/8fa5e2f6902357489c3393399df52842</td>
<td>Reproducible</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No-later-than</td>
<td>16 Dec 2020, 2:56:44 pm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 1: Data Readiness Answers for the Example Workflow

<table>
<thead>
<tr>
<th>Dependencies</th>
<th>hdl: 20.5000.1080/181b25f7ddab22ae1ddd833aa227498f</th>
<th>Attested</th>
<th>N/A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>hdl: 20.5000.1080/aa366bb40cff03d9686a76380dd2c036</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>hdl: 20.5000.1080/c05a3a8132d39e8c649f98201615f6a1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>hdl: 20.5000.1080/7c52629b4ad066b28153e6f6b4886a7b</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Workflow       | hdl: 20.5000.1080/37114977ec9a9da3e082f77e134a27bb | Reproducible | Source code: https://collab.cordra.org/#objects/20.5000.1080/a7a5cb440a6b1bc308aeccb6e86b9a14 |
|                |                                             |            | Public Key: https://hdl.handle.net/20.5000.1080/cordra?index=1300&noredirect |

| Source         | hdl: 20.5000.1080/7ffbb3e190cfe1b631714cfd4e4c67fe | JWT        | Source code: https://collab.cordra.org/#objects/20.5000.1080/b0ea2e054a918234cfa774e6101c8478 |
|                |                                             |            | Public Key: https://hdl.handle.net/20.5000.1080/cordra?index=1300&noredirect |

| Binding        | hdl: 700:20.5000.1080/a76b412c0a880d747b67651c55f1eb9 | Attested  | N/A |


Listing 1: Type Record that includes the JSON schema and code that defines the structure and serialization of data resulting from the Example Workflow
There are additional calls that are discussed in the Collab section next. They enable consumers to repeat and reproduce a workflow and data providers to setup such workflows, but these calls are not standardized in this paper given the complexity inherent to the very notion of workflows and the related advancements made by sophisticated workflow systems already. This is a good area for future research.

5. Collab Software

Collab is an open source based prototype software extension to Cordra software [5]. Collab, when integrated with Cordra, can be used along with existing data management systems to augment those systems with the following capabilities:

- allot digital object identifiers to data,
- store snapshots of such data to provide original data (in case the source data mutates),
- answer data readiness questions for snapshots, and
- implement API calls proposed in Section 4.

Collab’s key ability is to accept registrations of operations (in the form of executable software) and stitch those operations together by applying the logic in the corresponding software at runtime, thereby resulting in repeatable and reproducible workflows. Collab executes the constituent operations of a workflow to generate, compute, or fetch data from any number of external sources with bespoke APIs. In fact, Collab’s ability to work with existing data management systems is a specific case of this generic ability to interact with external data sources. We discuss Collab software internals next.

Collab is an extension to the Cordra software. Cordra software is a broadly useful digital object architecture based system that not only enables create, retrieve, update, delete, and search operations on digital objects, but also provides the functionality to add and invoke custom operations. Collab shapes how those custom operations can be deposited into a Cordra instance. Collab further stipulates specific attributes for different types of data structures used to support the aforementioned capabilities: snapshots, operations, and invocations; all of those are managed as digital objects, each associated with its own unique resolvable identifier.

At its core, Collab (because of its reliance on Cordra) has a code execution environment that executes operations, the results of which are stored as snapshot digital objects. Each snapshot is timestamped, typed, and hashed; and contains the identifier of the operation that produced it. Users can create a workflow as a chain of operations. When the workflow is executed, Collab stores snapshots produced at each stage. The provenance contained in each of the snapshots along with the knowledge of the chain of operations that constituted a workflow makes it possible for others to repeat and verify reproducibility. Collab supports reuse: the snapshots from one workflow may be used as source data in future workflows, and types and operations can also be reused.
Workflow and Operation

A workflow is a chain of operations (along with necessary input), where such chain is determined at runtime as codified in the logic of the executable code in those operations. In other words, an operation determines what other operations to run depending on the state of data an operation is processing.

An operation is a uniquely identified digital object that includes ‘code’, a designation of the programming language of the code (default is JavaScript), the ‘type’ of data the operation produces, a timestamp when the digital object is created, and the hash of the information from the digital object. Once created, changes to the operation digital object are disallowed. The identifier itself is derived from the hash; any changes made to the digital object can therefore be detected. The identifier is excluded from the input to the hash function, so the identifier itself could be derived from the hash.

Collab provides a (JavaScript) library that the code (of an operation) can use to invoke other operations to create a dynamic chain of such operations. Code can be written to interact with external systems (by implementing to the API of the external systems). In any event, the objective of the code (and therefore the operation) is to gain access to contributed data, and/or to generate appropriate data, and process that data to produce a snapshot conforming to the type specified in the operation data structure.

A user can execute a workflow by requesting Collab to execute the first operation of the chain: Collab will execute other operations as necessary to complete the first operation. As part of the execution request, and for each anticipated operation in the chain, the user can specify whether Collab should run the operation again or use existing snapshot data instead. Based on the request, Collab either re-runs the code from the chain of operations using its code execution environment or uses snapshots at appropriate stages of the execution. Whenever an operation is run (or re-run), Collab stores snapshots that correspond to the results from the execution of that operation. In any event, Collab returns to the user the resulting snapshot from the first (and main) operation.

Snapshot

A snapshot is a uniquely identified digital object that has a data portion capturing the results from the execution of an operation, the type of the data (which matches the type specified in the corresponding operation object), the timestamp at which the snapshot is created, the identifier of the operation digital object that produced the snapshot, and the hash of the snapshot itself. Like operations, the identifier of the snapshot is based on its hash making it possible to detect modifications, which are disallowed.
Code pertaining to an operation can be written either for a specific objective or in a generic way enabling reuse, where input parameters passed to the operation will bring runtime specificity. A special input parameter, in the form of a query, can be passed to an operation to specify that Collab could bypass running the operation and, instead, use existing snapshot data (possibly from such operation’s prior runs). This query parameter that targets properties from the snapshot to select one snapshot possibly from among multiples. An example query would be to indicate that the latest of the snapshots produced during a specific timestamp range should be selected if present (instead of running the operation again).

**Invocation**

An *invocation* is a uniquely identified digital object that contains the identifier of an operation and the input parameters that were passed at runtime to that operation. Collab creates an invocation digital object automatically when it observes a new combination of parameters along with an operation request. Like operations and snapshots, invocations too are hashed. The identifier of the invocation becomes a shorthand way to refer to an operation with a specific combination of parameters. In other words, invocation is a parameterized operation. A workflow, in Collab, therefore is a chain of *invocations* determined at runtime.

**Repeatability and Reproducibility**

A key enabling functionality of Collab is to allow users to repeat a workflow easily and verify if that workflow can be reproduced across runs. A workflow can be run or re-run by using its invocation identifier. Collab displays a visualization of not only the result from the workflow, but also the intermediate snapshots that are produced as a result of the invocation of the workflow. That visualization, which we refer to as dependency graph, shows which snapshots are derived from which other snapshots.

Figure 2 shows a (partial) dependency graph for an example workflow. This can be accessed by following this link:

https://collab.cordra.org/collab/objects/20.5000.1080/a76b412c0a880dfb47b67651c55f1eb9.

When a workflow is re-run, Collab detects if the result or any intermediates deviated from the original run. Collab provides users an option to select the result from the original run to encapsulate the intermediate snapshot that deviated, to see if a re-run with the use of original snapshot results in a reproduction. This will help users understand, objectively, the contributing factor for the reproducibility failure. This feature is not illustrated in this paper, but is available on the deployed system.
We have defined a model for associating a set of claims with data, where such claims carry not only the proof, but also that such proofs are pre-verified. Claims in this context are any attributes about data, such as those that characterize data quality. We described that model here [4].

Collab software includes an implementation of that claims manifest model. Collab stores the data readiness responses in the claims manifest, and associates such manifest with each snapshot that is produced. Users can retrieve the manifest, ensure that the manifest is not tampered, and accept the data readiness responses.

Figure 3 shows a visualization of the claims manifest for the example workflow. The visualization can also be accessed by following this link: https://hdl.handle.net/20.5000.1080/a76b412c0a880dfb47b67651c55f1eb9?locatt=view:claimsUI.
Collab supports the following APIs:

1. Given a snapshot identifier, retrieve the snapshot (Call 1, defined in the Prototype section).
2. Given a snapshot identifier, retrieve data readiness answers (i.e., claims manifest) associated with the snapshot (Call 2).
3. Given a type identifier, retrieve the type digital object consisting of the type description (Call 3).
4. Given a snapshot identifier, retrieve its dependencies.
5. Create an operation digital object.
6. Given an operation identifier, retrieve the operation digital object.
7. Create an invocation digital object, for the combination of an operation identifier and parameters.

Figure 3: Claims Manifest of the Example Workflow
8. Given an invocation identifier, run or re-run the invocation (i.e., workflow) to produce a snapshot (along with any intermediary snapshots).

Collab software includes a user interface (UI). The UI displays a landing page for a given snapshot, using which users can access data, its type, its dependencies, and claims manifest as well as provide a way to repeat and reproduce workflows.

The landing page includes both API links as well as UI links. Users can bookmark or share the landing page link with other users.

Figure 4 illustrates the landing page of a snapshot produced for the example workflow. The landing page can be accessed by following this link: https://hdl.handle.net/20.5000.1080/a76b412c0a880dfb47b67651c55f1eb9?locatt=view:collabUI.
6. Conclusion

Organizations have recognized the value of driving decisions primarily based on data, but there are no standard practices around ensuring only fit data is used for decision-making. We presented a strawman for what constitutes fit data in the form of data readiness questions and provided an analytical approach to reason about those questions (in a domain neutral way). We explored the value of those questions with the help of a practical example and via a prototype software called Collab, which we plan to release to the public in open source form. We anticipate taking further actions by initiating a conversation in the scientific community on the topic of data readiness based on this paper and related work. In particular, we expect to include not only researchers in that conversation, but also research sponsors and policy makers because they are likely to play an important role going forward and also because of the negative impacts we may have to bear in the long run if unfit data is used for making decisions.

7. Acknowledgements

We would like to thank the technical staff at CNRI for helping us conduct the relevant research and produce this paper. Ian Little (from CNRI) deserves a special thanks; he implemented some key portions of the project in software and, in the process, helped us identify and resolve issues in some of the design elements of the project.

This material is based upon work supported by the National Science Foundation under Award No: 1838981.

References

[4] A Data Model for Pre-Verified Data Quality Claims: https://hdl.handle.net/20.1000/127