Towards Collaboration

Dave Iberson-Hurst, 2021

Change History

- 12th September 2021 – Initial Draft
- 4th October 2021 – Second draft after some initial comments

Contents

Introduction

When I think about the current situation within our industry, I see barriers, barriers to collaboration, barriers to better data. We speak the same language (CDISC) but different dialects. The bar to deploying the CDISC standards is set high, and tools are not readily available, especially to the academic community. Where tools exist, integration is difficult. The ability to pool data without a significant effort is elusive and we struggle to make use of such data for a variety of secondary uses and share it with others.

I was out walking while thinking about this article and I made some notes on my phone, sketched out some ideas. Later I airdropped the notes to my iPad and added some more info while watching some bad TV. I then dropped them onto my laptop. Why cannot I not do the same with my clinical study metadata and data? Why is it not possible to for sponsor A to “airdrop” a definition to CDISC, to sponsor B? The direction of information sharing is one way, standards come out from CDISC, but we cannot share a new definition electronically easily in the other direction, collaboration is not a two-way street.

Why can I not quickly deploy tools that assist me in collecting data that is of sufficiently high-quality that it pools with data collected by someone I have yet to think about collaborating with?

In the summer of 2020, I wrote a paper [1] about the need to define better APIs to assist vendors in building better tools and the need to align such APIs with an Industry Knowledge Graph. This needs to be made real such that the potential can be demonstrated to a variety of stakeholders.

This paper proposes developing that idea via the use of an industry knowledge graph implemented via a series of open-source micro-services. Iterations can then be used to expand functionality and demonstrate how industry could use such an approach to increase collaboration and remove silos.

This is not about prototyping; it is about a considered development executed iteratively designed to a) prove that data can have more utility other than the primary use case of submission; b) provide tools to the community at large; and c) provide a foundation for the community to build capability on top.

This paper is rather blue sky and has some big aims. It is about being able to bring data together for the public good. But I have written it to seed ideas in others, to start a discussion. What emerges may be quite different than what is written below but I feel it is worth the conversation.

Problem Statement

We often dive into solutions without detaining the problem. The following statements provide, at the highest level, the problem to which we need a solution. They try to provide the vision, the big picture of the problem we are trying to address. They are what we can return to when we are down in the weeds and need to remember why we are undertaking the work and to provide necessary clarity and purpose in moments of crisis.

1. [DESIGN] Build the design for a research project in a consistent manner
2. [IMPORT] Import raw data from a variety of sources and varied formats into my research project, my research data
3. [POOL] Pool my research data with other research data
4. [EXPORT] Publish my research data easily (FAIR principles)

These requirements are expanded in the section Appendix: Requirements below.

Approach

I have long believed that we can work towards the above aims by building what we do around a core of Biomedical Concepts (BCs), focused on the data, that provides a solid foundation for automation – and help remove the dialects by providing consistency – rather than being so focused on the necessary submission formats. Experience has shown that BCs, combined with high-quality, machine readable, study designs can be used to automate the generation of the range of outputs we need while allowing for greater utility from our data.

For this we need:

1. A foundation of Controlled Terminology (CT)
2. A layer of BCs linked to the CT
3. Operational structures that allow for
    a. The definition of project designs linked to BCs (and thus CT)
    b. the capture of data based on the design
4. Ability to bring the design and data together in one coherent location to allow for the generation of desired outputs from those designs and data

For further information please see [2].

Therefore, the suggested approach is to:

1. Use recent experience to develop an industry Knowledge Graph (KG)
2. Use the KG to develop a set of open-source micro-services to that, as a set, provide an implementation to industry employing BCs. It may seem odd to select an approach like micro-services, but I want to investigate it in the light of an

incremental approach and allow users to build capability. Micro-services would enable this.
3. Provide some simple User Interfaces (UIs) to demonstrate key facets of the implementation and key capabilities
4. Iterate and build capability implementing increasing coverage of the KG
5. Provide deployment capabilities to make it easy for those wishing to deploy the services

We should be guided by FAIR principles, see the section at the end of the document

Knowledge Graph

The key to ensuring that we remove silos from our world is to have a single representation of our world that reflects that world but provides a practical implementation of it. BRIDG does this but is not easy to deploy and experience has shown that a knowledge graph can provide a solid base for implementation.

There is a need to provide a sound base for that KG and there is a case for a metamodel to be in place to ensure consistency across the KG. The ISO 11179 metadata standard can provide useful pieces but I feel it constrains the benefits provided by a KG so we need to mix with caution. There are many good semantic standards such as SKOS that can be leveraged.

The KG can provide an Industry knowledge graph that all can share and improve with well-defined sections with good interfaces that lend themselves to a microservices architecture.

The KG should incorporate knowledge from both BRIDG and Transcelerate's DDF project. The KG should be dedicated to the task of being able to build and execute studies, automate the production involved in such and facilitate the rapid pooling of data from such studies.

One very important aspect of this suggestion is the knowledge graph. It provides for the single source of truth and everything we do is but a view of this graph. The service approach allows us to build useable components focused on one aspect and to build a solution by having the services cooperate.

Services

Overview

Once the KG has been developed to a sufficient level the graph can be implemented as a set of micro-services. The services should

- focus on one section of the graph
- implement the necessary business logic

- present a consistent interface via an API to consumers that could be an UI or another service
- be self-aware and allow for collaboration
- be able to load content from sources and thus front-end existing content
- users should be able to deploy one or more services easily

Focus

Each service should be focus on one job and one job only. It should be self-contained with its own database and only collaborate via APIs

Business Logic

The service should implement a common set of services and enough business logic that supports the ecosystem. This logic can be expanded over time through an iterative approach.

API

The API presented should conform to good design rules. The API will need not only to support the busines functionality required but also common functions to support such things as deployment, status monitoring as well as common functions such as CRUD operations

Self-aware

A service should be self-aware, i.e. it should be able to communicate with another instance of itself. For example, a sponsor's terminology micro-service should be able to send a code list to another sponsor's terminology micro service if appropriate permission is granted.

Load

A service should be able to load content from sources of data and allow for new loaders to be incorporated. This will allow services to "front-end" existing content. For example, a sponsor may have an MDR that contains its terminology. The terminology micro-service should be able to incorporate a new loader such that the content can be loaded into the micro-service. This will then allow that content to be shared within the ecosystem. This isolates the sponsor's system from change but allows the community to move forward and protects the sponsors existing investment.

Deployment

Micro-services should be easy to deploy without technical knowledge. See the section below.

Technology Neutral

Services should be technology neutral, they could be implemented in Java, Python, Ruby etc. In theory databases for services could be relational or graph but I think it would be wise, because of FAIR, to use a semantic approach and allocate URIs to items that persist through time. The KG drives us in this direction.

Services should be designed to make use of common libraries for the common functionality expected to be implemented within services. This will allow services to be implemented quickly.

APIs

Every service type needs to provide a public API that is published to allow tool builders to build against or use that API within their own tools. The API should provide:

- Common API services – functions that are common to all services and that all services should provide
- Role Specific API – functions specific to the service

It would make sense to build upon the existing CDISC Library APIs and build them in a common style etc. But there may need to be some updates to allow the support of FAIR principles etc.

APIs need to be freely available and in the public domain and come with no restrictions regarding their use.

Services should also support import and export in specific formats such as ODM, Define.xml etc.

User Interface

User Interfaces can make or break an implementation and are very important, but they are also expensive to build. It is suggested, initially, to build any UIs using R Shiny and make open source. As a principle it might be nice to have a simple open and free UIs such that any organization can quickly get up to speed.

With that in mind, experience has shown that people still love MS Excel. It is flexible and everyone seem to have it on their desktop “for free”. It would be interesting to see if we can we drive a service from Excel. I have had some initial thoughts on the challenges, and it might be possible in most cases. This may be of interest to individual users and smaller organizations. It may seem odd to be advocating the use of Excel but it is a reality in our industry and people like it. With a little structure (simple sheet formats) we can help a lot of users.

User Interfaces, like the services, should be offered as open-source software with a single service focus. A set of UIs with a common look and feel will allow users to use the functionality immediately without cost. However, this does not prevent complex, expansive, and complete applications being built by vendors with vendors free to charge for those UIs.

Deployment

As services become stable it would be beneficial to the community to be able to deploy them quickly and select associated user interfaces. It would be ideal if there was a portal whereby a user could select a set of services could be quickly selected, configured, and deployed on servers. This process should be made as simple as possible and would be of interest to smaller sponsors and academia.

Imagine a world were you, as an individual researcher, could log on to cdisc.org, view a single web page listing a set of services, pick the CT one, enter minimal configuration details, and have that CT service up and running within minutes somewhere in the cloud. Coming with the backend should be a free UI ready to do the basics. Other than the costs of the cloud hosting, it should not cost. You should then, via the simple UI, be able to add your CT easily, access the CDISC CT, search the community to see if there is a VT you don't have meeting the needs of your next study. It should be easy.

You should be able to add further services as you need them. The web page should also offer the "study set", the set of services you need for an entire study to be deployed as easily. The services should integrate with vendor tools such that a study design can be readily deployed, the data easily received back.

Services Envisaged

Below is a list of envisaged services. The list is not definitive, depends on design issues etc, but worth noting to give a flavour of what might be possible.

- Terminology (CDISC)
- Other Terminology
    - LOINC
    - MedDRA
    - Others
- Terminology Equivalence
    - CDISC <-> LOINC
    - Others
- BCs
    - Connect to BC mining. It would be possible to feed the BC service from the mining tool thus allowing a library of BCs to be quickly shared.
- Forms
- SDTM
    - CDISC core and Sponsor implementations of domains
- Collections
    - Base for TAs (a combination of domains, BCs and forms)
- Study
    - Study definitions such as endpoints.
    - Study Build resulting in a complete design.
    - Study Data attached to the appropriate design. This is the important service in that it is the one that allows the ecosystem to share data.
    - EHR / FHIR Interface. Take a study design and fulfil the study contract with data from an EHR, see [3].
    - Automate SDTM from Study Data, see [3].
    - Conformity. Given SDTM is autogenerated the nature of the checking will change from post to pre-validation. May want to integrate with existing tooling?

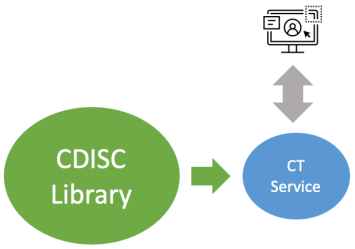
Scenarios

General

The few scenarios described below have been included to give a flavour of what is possible and what could be achieved. They are not definitive and do not cover everything that could be done, they are purely for illustrative purposes.

"Air Drop" Terminology

Imagine we have a single service. It knows about the structure of CDISC terminology, has a CDISC library API loader such that content is up to date and has a simple browsing and search user interface. We could spin up a service that sits in front of the CDISC library and allow people to look at the CDISC CT. Not really a step forward.



Now, if that service implemented a few more features, say allowed better relationships between code lists such as RACE and RACE AS COLLECTED then it offers an improvement, albeit small. But it allows CDISC to provide richer content without disruption to the main library development.

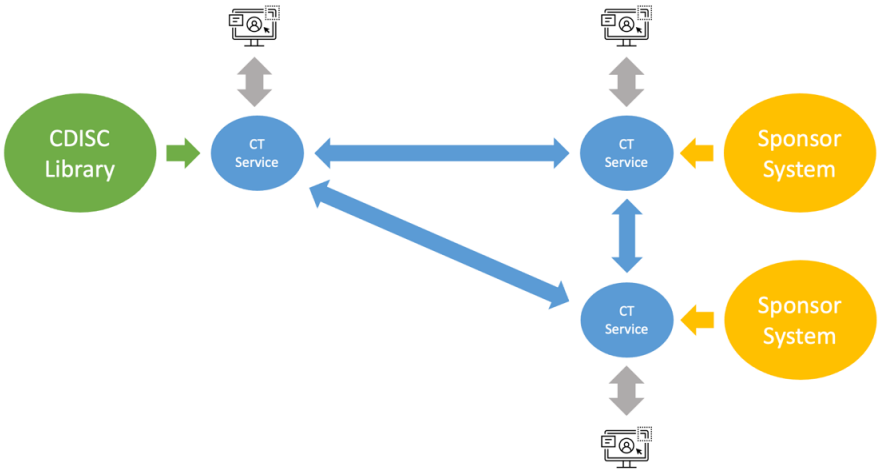
Now a sponsor writes a loader that can take its terminology that is used in conjunction with the CDISC CT and populates an instance of the service. The same UI allows a user to browse and search its own CT.

However, if the services are self-aware then any search or browsing could include the content from the other service if that content has been made visible. Obviously the CDISC content would be, but the sponsors would probably not be.

Items (code lists) in the sponsor service could be made public and thus become visible to the outside world. CDISC could then, if it so chose, request a copy of an item (code list) as a draft for new CDISC official content.

Sponsors will be able to see draft content a lot easier if such content was placed into the service allowing for review etc.

Now a second sponsor runs up a service. The sponsors could, if they so choose, share some selected content by making the item(s) public. CDISC then has the option to pull that interesting content for the basis of new CDISC content after appropriate review.

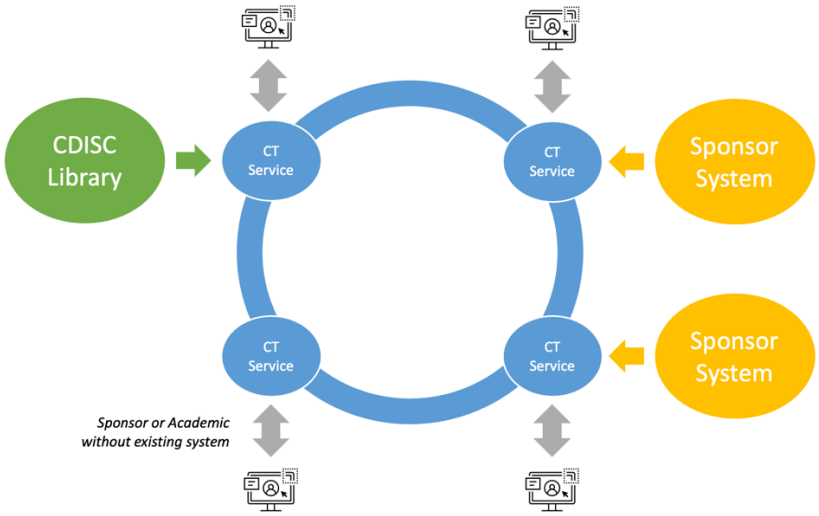


We now have “air drop” capability and we have a terminology ecosystem.

More and more users can deploy a terminology service, federated searches are possible; “does anyone have content for X”, the conversation is two-way, the community is collaborating, and the cost of entry is lowered.

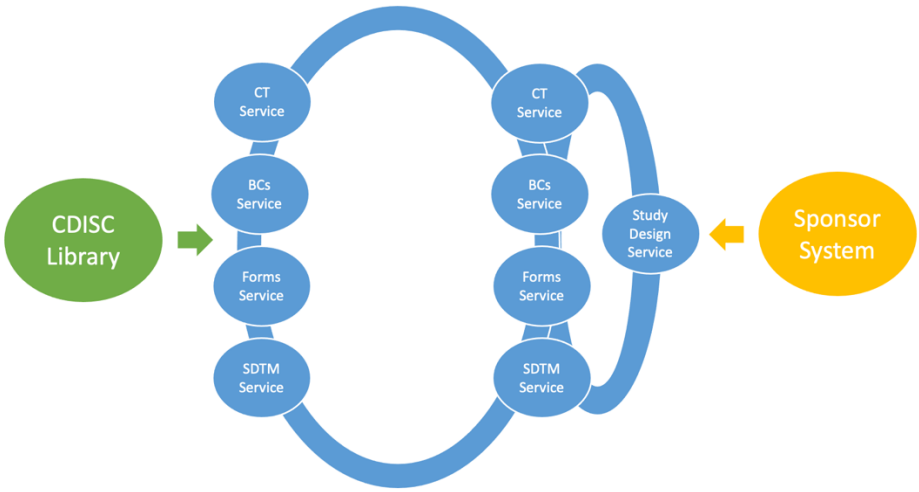
The question organisations will ask is “what is in it for me”. One obvious use case is the auto submission of existing local code lists to CDISC for incorporation into the standard. Similar will be needed for BCs. This point does need serious consideration.

Note that an organisation may not have an existing system and thus can just deploy the micro-service and a UI to drive it but join the collaborative world with little investment.



Study Build

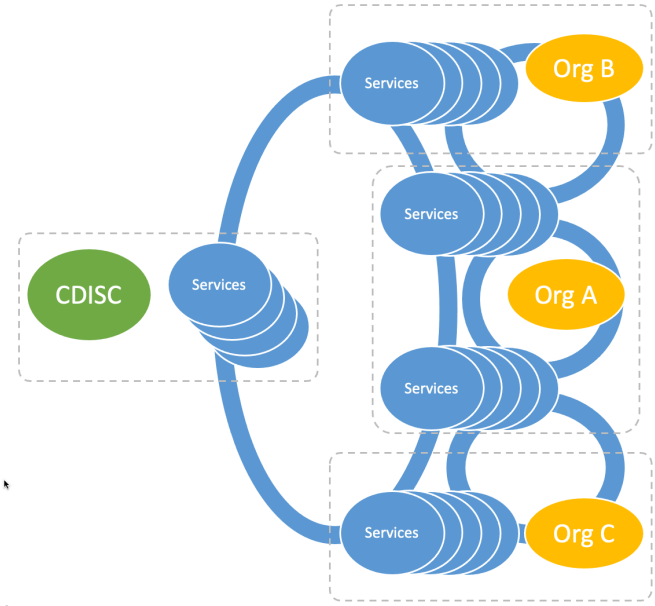
This scenario is just here to illustrate a more complicated example of multiple services being deployed. Let us assume that CDISC has services for terminology, BCs, Forms and SDTM. A sponsor has the same set, but these are providing additional sponsor BCs, forms and sponsor configured SDTM domains. If we add a Study Build service, we have everything needed to build a study. Here we have two ecosystems as illustrated, the one sharing definitions between CDISC and the sponsor as above and then another for the sponsor and its studies. Of course, a second sponsor can have its own study design service and linked to the first sponsor's if they wish to collaborate but CDISC isn't concerned with study builds, just the tools and definitions to get there.



Organisation Partners

This scenario illustrates an organisation working with partners on different developments; an example would be a CRO with sponsor customers. One large

ecosystem allows all the organisations to link to CDISC while keeping other work separate from the global community and shared only with the appropriate organisations. Such a model might also be employed within large organisations to enforce separation when it is desired.



Ecosystems

As can be seen from the scenarios described above, as services are added, and those services grow in capability, you start to build an ecosystem. Such ecosystems might be of interest to those users in the academic community who need simple and quick access to the use of the CDISC standards. The ecosystem may also be useful to demonstrate capabilities and potential for such use cases as the Transcelerate DDF project.

In addition, by providing the services you open the door to the development of third-party user interfaces that go beyond the basic open-source ones provided.

Iterate

Not everything can be built from day one. It is suggested that the initial focus be to demonstrate BCs and Study artefacts first. It would then be wise to expand and provide a constant flow of releases, add new services, and expand those contributing to the work.

SOLID

While writing this paper I discovered SOLID, https://solidproject.org/. I found the notion interesting; it has a lot of the technical aspects that a clinical study ecosystem would need (user authentication, roles etc.). It also has the notion of the pod. I have not had the time to go into all the details in depth, but a pod seems to be a personal data store. Two thoughts are currently rattling around my head. Could a pod be used for a subject's study data? Could the notion of a pod be used for the data from a single study? Those are quite different approaches, but I think worth thinking about.

And the But

Of course, there is always a "but"! What about existing tools and systems, EDC, IVRS, ePRO and the multitude of other system currently humming away doing their jobs. Well, it will be slow, first the acceptance of the idea, a slow move to integration, using services to perform some key actions (e.g. terminology) until slowly acceptance emerges. It will take time and much discussion.

Risks

Not everything in the garden is rosy! A few risks:

- KG, getting there, acceptance of it.
- Encourage sponsors to share their definitions (CT, BCs etc). Mindset change.
- Existing tools and integration.

Issues

During the development the following issues will need to be resolved. They are noted here such that they are not forgotten:

- Resolvable URIs and FAIR usage
- Authentication, use industry standard mechanisms
- User Roles
- Service Location and Discovery
- Organizations and management thereof (ISO11179 related)
- Status of items using ISO11179, keep it simple. Align with CDISC Library.

References

[1] Models, API and FHIT, A New World, Iberson-Hurst, July 2020

[2] FAIR Principles.
See https://www.go-fair.org/fair-principles/

[3] Into the fire, Linking CDISC and FHIR, 2018.
See https://www.lexjansen.com/phuse/2018/si/SI12.pdf

[4] Removing Silos: Placing Data at the Centre
See https://www.lexjansen.com/phuse-us/2019/si/SI13.pdf

Appendix: Requirements

High Level Statements

1. [DESIGN] Build the design for a research project in a consistent manner
2. [IMPORT] Import raw data from a variety of sources in varied formats into my research project, my research data
3. [POOL] Pool my research data with other research data
4. [EXPORT] Publish my research data easily (FAIR principles)

DESIGN

1. Must be able to create the design for a research project
2. Must be able to use a design in subsequent stages of the research project
3. Must be able to reuse a design in another research project

IMPORT

1. Must be able to import data matching a research project design
    a. Must be able to import Excel data and link to the design
    b. Must be able to import data captured by systems and link to the design
2. Should be able to import SDTM data to create a design and link the data as a project

POOL

1. Must be able to query across one or more research projects
2. Must be able to search across one or more research projects
3. Must permit third party tools to access one or more research projects

EXPORT

1. Export a research project conforming to a specified SDTM version

Notes

1. Research project is used as a generic term for a clinical trial/study, longitudinal study etc.

Appendix: FAIR

The following is reproduced from [2]. Inserted here for completeness.

Findable

The first step in (re)using data is to find them. Metadata and data should be easy to find for both humans and computers. Machine-readable metadata are essential for automatic discovery of datasets and services, so this is an essential component of the FAIRification process.

- F1. (Meta)data are assigned a globally unique and persistent identifier
- F2. Data are described with rich metadata (defined by R1 below)
- F3. Metadata clearly and explicitly include the identifier of the data they describe
- F4. (Meta)data are registered or indexed in a searchable resource

Accessible

Once the user finds the required data, she/he/they need to know how can they be accessed, possibly including authentication and authorisation.

- A1. (Meta)data are retrievable by their identifier using a standardised communications protocol
    - A1.1 The protocol is open, free, and universally implementable
    - A1.2 The protocol allows for an authentication and authorisation procedure, where necessary
- A2. Metadata are accessible, even when the data are no longer available

Interoperable

The data usually need to be integrated with other data. In addition, the data need to interoperate with applications or workflows for analysis, storage, and processing.

- I1. (Meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.
- I2. (Meta)data use vocabularies that follow FAIR principles
- I3. (Meta)data include qualified references to other (meta)data

Reusable

The ultimate goal of FAIR is to optimise the reuse of data. To achieve this, metadata and data should be well-described so that they can be replicated and/or combined in different settings.

- R1. (Meta)data are richly described with a plurality of accurate and relevant attributes
    - R1.1. (Meta)data are released with a clear and accessible data usage license

- R1.2. (Meta)data are associated with detailed provenance
- R1.3. (Meta)data meet domain-relevant community standards