Paper AD14

Building a large scale clinical trial IT system with in-house software and OpenClinica

Sonja Grotjahn, OxPop Department of University of Oxford, Oxford, UK
Allen Young, OxPop Department of University of Oxford, Oxford,UK
James Thompson, OxPop Department of University of Oxford, Oxford, UK

ABSTRACT

For one of the large scale clinical trials run at Oxford Population Health department of Oxford University, the AscendPlus trial, we combined third-party cloud based platform OpenClinica® with our in-house built software. OpenClinica is a clinical data management and electronic data capture system which offers many features to enable easy and fast development of software for a clinical trial. However, we found that due to the nature and the scale of our study, there were several software components that needed to be built in-house, such as treatment supply management, clinics administration and handling postal and email communication with participants and responses. For the smooth running of the trial it was important to provide timely exchange and synchronization of data between OpenClinica and the in-house software. OpenClinica provides several methods for interacting programmatically with its in-the-cloud data. This paper reports on our experience on using some of those methods to perform integration with the rest of the system and issues we encountered using these methods on a large scale trial. We will describe how we used not just APIs provided for the OpenClinica main system, but also APIs of the separate reporting module Insight to achieve the needed level of control, monitoring and integration with the remaining software for the trial.

INTRODUCTION

The Clinical Trials Unit within the Oxford Population Health department of Oxford University specialises in running large scale phase IV trials involving thousands of participants. We are striving to conduct those trials as efficiently as possible – we often say our methodology is streamlined. Following on from one of our previous trials, the Ascend trial (ASCEND, 2021), in which we successfully recruited over 15000 participants and followed them up for around 7 years completely remotely from a single coordinating centre, we have designed one of our latest trials, the AscendPlus, trial to be run remotely too (AscendPlus protocol, 2022).

The AscendPlus trial is testing the possible effects of a drug called semaglutide in the prevention of cardiovascular diseases in people with type 2 diabetes. We aim to recruit 20,000 participants for this trial, in collaboration with the National Health Service (NHS). The communication with participants, including screening and follow-up will be done by post, by phone or via Internet. The drug will also be delivered to them by post.

In the past we would build the whole IT system for a trial within the department, but recently we started using off-the-shelf components like OpenClinica (OpenClinica, 2023). OpenClinca (also referred as OC in this paper) is a cloud based, open-source, clinical data management and electronic data capture system. The platform offers many features such as rapid visit forms design, scheduling, auditing, issues tracking, direct involvement of participants via the 'Participate' module and reporting via the 'Insight' module. Still, we found that there were areas where we needed to create additional software components for our study. In this paper we will explain why we complemented OpenClinica with our in-house software, we will describe various additional components we built and the way we achieved the needed level of control, monitoring and integration of OpenClinica with the remaining software for the trial. Special attention will be given to the components that exchange data between the in-house software and OpenClinica and what we did to improve the efficacy of the exchange.

OPENCLINICA – WHY WE USED IT AND WHY WE COMPLEMENTED IT WITH OUR SOFTWARE

OpenClinica enables fast building of forms for data capture including directly from participants. The platform offers many other features like scheduling, auditing, issues tracking and generating data extracts for regulatory submissions. There is an optional 'Randomize' module so the system can be used to randomise participants too. The main benefit we gained from using OpenClinica were:

- **rapid generation of visit forms** – this feature speeded up the development of software for this trial in spite of added time for learning the system,
- **direct involvement of participants via 'Participate' module,**
- **report generation using module 'Insight'** which allows querying of the Insight database which contains a copy of the clinical data, and
- **extensibility via Application Programming Interfaces (APIs).**

We identified several areas where we needed to complement OpenClinica with our own software:

Recruitment. We employed NHS to invite potential participants on our behalf by post. The study software then processed the returned forms and uploaded data for consented participant into OpenClinica.

Managing sites and participants' appointments. Although it is possible to schedule an appointment in OC, it is not yet possible to control and schedule clinics' time and availability.

Randomisation – we chose to use our own algorithm for this.

Managing participant statuses and progression through study. Support for participant timeline with rules for transitions through phases (epochs) is planned for a future version of OC, but is not available in the current version.

Handling Adverse Events. In the design of our study the local coordinators using OpenClinica wouldn't necessarily be medical professionals. Also the participants themselves will be able to fill the same forms. This information will be treated only as a potential adverse event. The clinicians within our department will use our in-house software to view and evaluate the data and record adverse events if appropriate.

Managing treatment supply. In our trial the treatment is sent to participants by post. Our trial managers coordinate supply of the treatment from to the participants using an in-house built software module.

***Email and SMS communication with participants*.** It is possible to send email or SMS to participants from OpenClinica, but we didn't have control over the text of the message and more importantly, the fact that the email has been sent is not recorded in the data available through APIs nor there was any information about the delivery status.

There are other components of the system, like in-house software user management, but in this paper we will give emphasis on the in-house built components that perform the exchange of data with OpenClinica. The features of the system that directly depend on those components are circled in red in the figure 1.

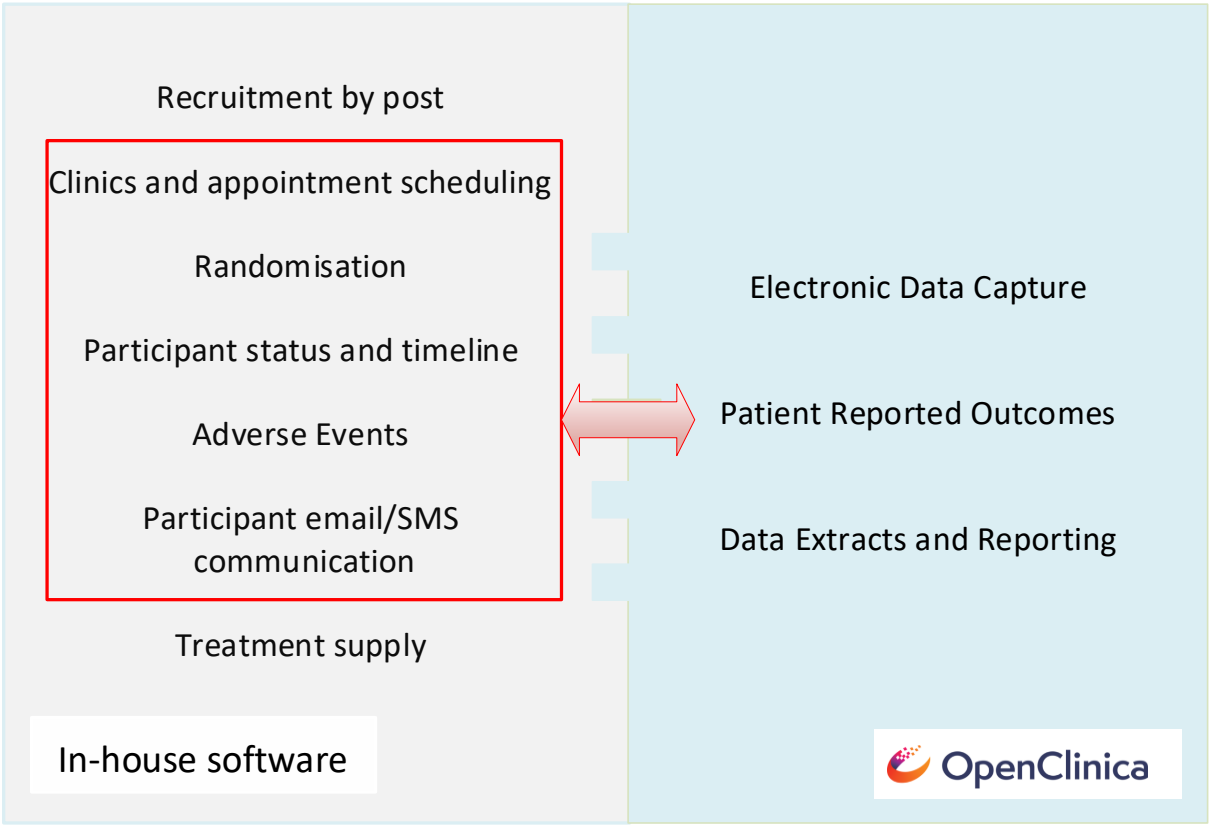

FIGURE 1 IN-HOUSE FUNCTIONALITY COMPLEMENTING OPENCLINICA

THE DIAGRAM OF THE SYSTEM

Figure 2 shows the basic configuration of the system. The main in-house built application, Aldrin, encompasses most of the additional functionality needed for the study, but there are several other smaller components in the system like the software handling recruitment forms and data, software for handling GP practices and various back-end components named Apollo. The in-house software stores data in a couple of in-house databases. The user applications are outlined in blue in the diagram and the back-end programs in red.

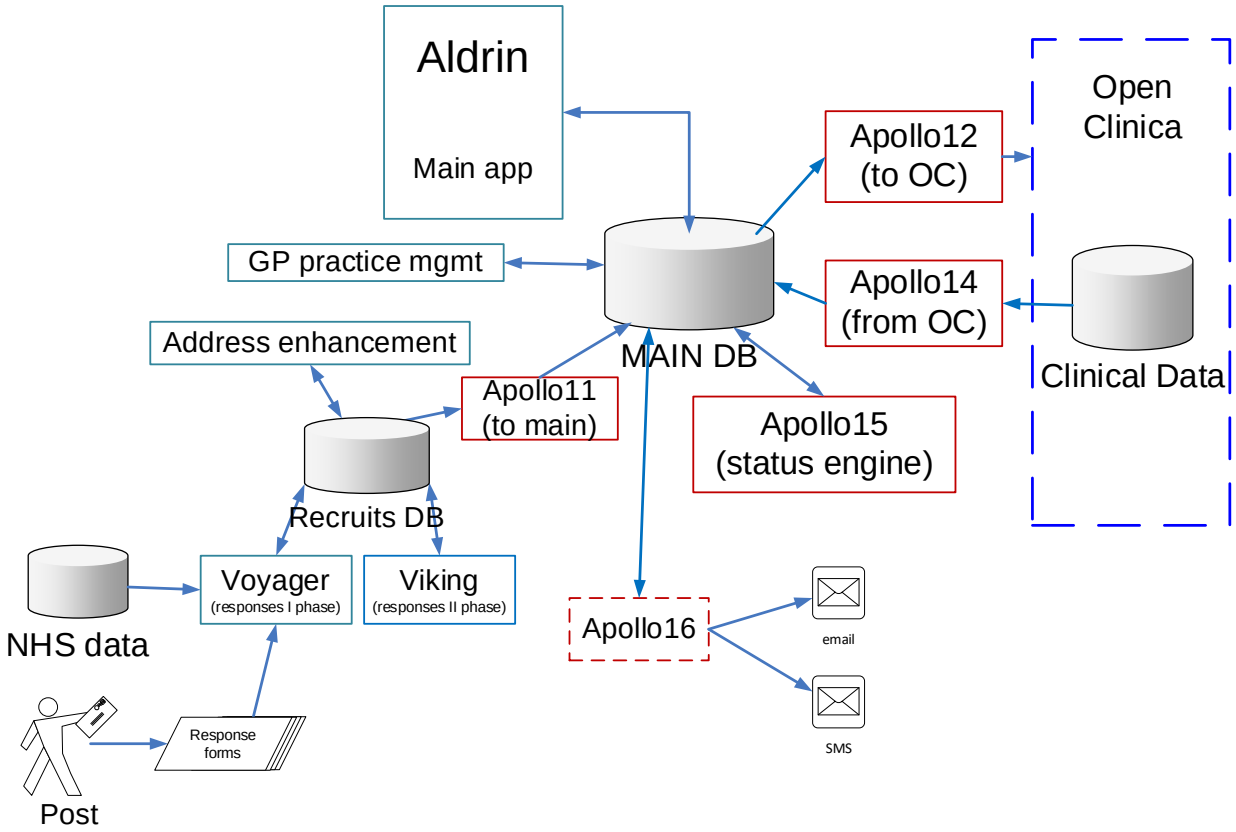

FIGURE 2 DIAGRAM OF THE SYSTEM

The main in-house application, Aldrin, provides the bulk of the user interaction with the system outside OC. Aldrin also implements our proprietary randomisation algorithm. Aldrin relies on clinical data from OpenClinica that has been transferred to the main database. There are two programs that provide the exchange of data between the main database and OC:

Apollo 12 – transfers data from the main study database to OC

Apollo 14 – transfers clinical data from OC to the main database

The accuracy and timeliness of the data exchange was one of the critical points in the system design.

OPTIONS FOR DATA EXCHANGE

There was more than one way to provide data exchange between in-house software and OpenClinica, but we have decided to use OpenClinica APIs to programmatically exchange data. The reasons will be explained below.

Transfer from the main database to OpenClinica

OpenClinica has an option for bulk-upload of participants. This feature is popular among OpenClinica users, but we wanted a more flexible approach so we opted for using APIs where we could update a single field for a single participant, for example the study phase that a participant is in.

Transfer from the OpenClinica to the main database

Besides the API access, OpenClinica provides the following means of accessing data:

Generating data extract using OpenClinica interface. The reasons for discarding this option were that
- the extract is accessed by clicking on the link from an email generated by OC which seemed impractical,
- It wasn't possible to run those extracts more than once per day,
- OpenClinica assigns names of fields at the moment of download and those don't' match field names nor IDs in the form design

***Downloading data from Insight database*.** The Insight database contains a copy of the OpenClinica data.

The data is stored in several tables which correspond to forms, item groups, lists definitions etc. We have discarded this option because:
- the refresh of the database is normally every hour and it is not guaranteed to succeed – this is a lower priority task in the system which might be abandoned in favour of more important ones,
- while the data downloaded directly from OpenClinica contained the metadata describing forms and fields, this information is not readily available through Insight.

In summary we have opted for the direct access to OpenClinica using its APIs as it was giving us:

- Better flexibility / control over the data
- Timeliness

OPENCLINICA APIS

We chose OpenClinica's RESTful APIs with OAuth 2.0 authentication, as our web-based applications use the same architecture. OpenClinica also provides SOAP APIs that we didn't use. Some of the notable RESTful APIs we used (OpenClinica, 2023):

- to get a case report form for a participant:
  `GET /OpenClinica/ClinicalData/{format}/{mode}/ODM_XML_PATH?OPTIONS`
- to register a new participant:
  `POST …/api/clinicaldata/studies/{studyOID}/participants`
- to create a new event for a participant:
  `POST …/api/clinicaldata/studies/{studyOID}/sites/<<SITEOID>>/events`

IMPLEMENTATION

We used a combination of web-based applications developed in Java using the Spring Boot framework (Spring, 2023), RESTful web services using the same technology and bash scripts and C++ programs for some back end operations. Our main back-end database was Ingres.

A PROBLEM WE ENCOUNTERED USING OPENCLINICA DOWNLOAD APIS

In our original plan, the Apollo14 program would download all of the clinical data from OpenClinica every night using the GET request above and update the main database with the data. When we started testing our software, we soon realised that with an increasing number of participants in the test environment, the time taken to download the data started increasing almost exponentially at one point. The following table shows some of the times measured in the test environment.

date	participants	time (minutes)
23-Jan	3	0.4
30-Jan	2504	73
15-Feb	5109	193
25-Feb	6186	240
13-Apr	6318	559

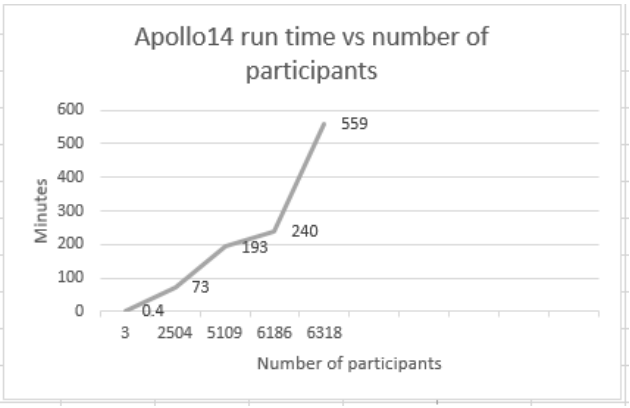

FIGURE 3 DURATION OF APOLLO14 RUN IN TEST ENVIRONMENT VS NUMBER OF PARTICIPANTS

THE SOLUTION

We needed a way to limit the number of API calls, ideally to get only the new data. The OpenClinica APIs were not offering this so we decided to complement the system with a tool that would obtain the information on what data is new from the Insight reporting module. This was achieved in the following way:
1, A new report was created in Insight: the last updated time for a participant report;
2. A new program called Kittyhawk was created which downloads data from this report using Insight APIs and then loads the data into the main database;
3. Apollo14 was amended so that it first compares the last updated time for a participant with its own last run time before downloading only those participants whose updated time is newer.
By implementing the transfer of data from OpenClinica as a combination of OpenClinica APIs and Insight APIs, we

achieved the desired level of accuracy, timeliness and completeness without sacrificing OpenClinica performance for users.

Some details about the implementation of Kittyhawk

This software component consists of two parts:
aplusreporter - a java command-line application built with Spring framework which actually downloads the report data. This application calls the Insight API:

```
POST /api/card/:card-id/query/:export-format
```

where *card_id* is the unique ID of the report and the *export_format* field was set to 'CSV'.
kityhawk bash script retrieves the card-id for each of the reports we needed from the configuration stored in our database, calls aplusreporter to get the report data and then loads the data into relevant tables in the database

FURTHER IMPROVEMENTS TO HANDLING OPENCLINICA DATA

The format of the downloaded clinical data is XML. Apart from the clinical data, each downloaded XML document contains the OpenClinica metadata. The sizes of the free text fields are up to 4000 characters. Apollo14 stores the whole XML document in a single field in the database. To enable easier access to form items to the main user of this data, program Aldrin, we have created another program that extracts the forms data from the XML document and stores it in database tables, one row per a form item. The program also extracts events, forms and items metadata information into appropriate tables to enable easier referencing of item data. We called the new helper program – Littlejoe, like the booster rocket used in NASA project Mercury. Littlejoe is a C++ command line program which uses the tinyxml library for XML parsing and Ingres database libraries for database access.

The updated configuration of the system

The diagram in Figure 4 shows updated configuration of the system with the above discussed additions highlighted and elements not relevant for the exchange of data with OpenClinica shaded. These additions enabled the in-house software components to run entirely on the in-house study databases with all the relevant data made available by two additional programs kittyhawk and littlejoe.

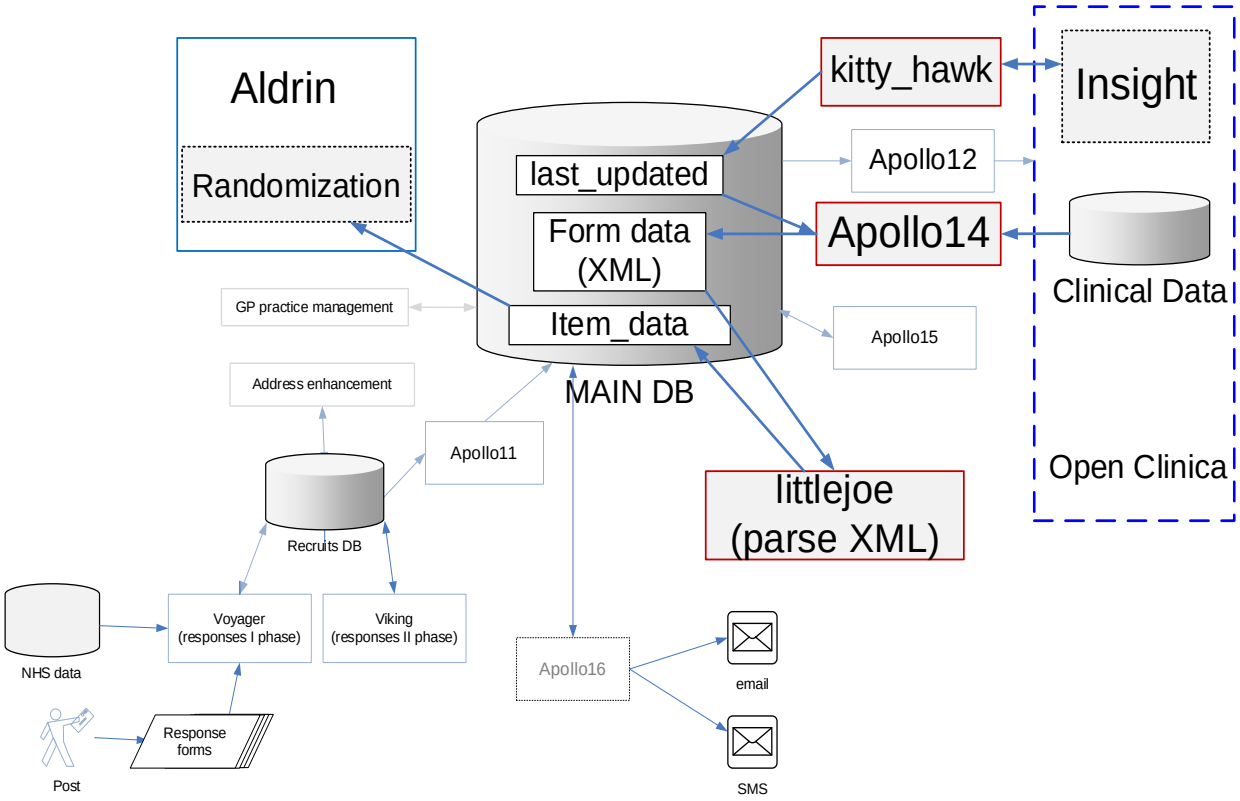

FIGURE 4 THE UPDATED CONFIGURATION OF THE SYSTEM

The additional software components brought a large improvement in the speed of execution of Apollo14 as it is

shown on Figure 5. The execution time is now shown to depend only on the number of new or updated forms rather than the number of participants. On a day with an increased activity and with 34 forms downloaded, the 9th of June, the duration was 1.43 minutes which was more than acceptable compared to the previous lengths measured in hours.

date	participants	time (minutes)
17-Apr	6432	0.23
27-Apr	6439	0.28
02-May	6439	0.23
17-May	6439	0.18
01-Jun	6443	0.23
09-Jun	6454	1.43
22-Jun	6509	0.53

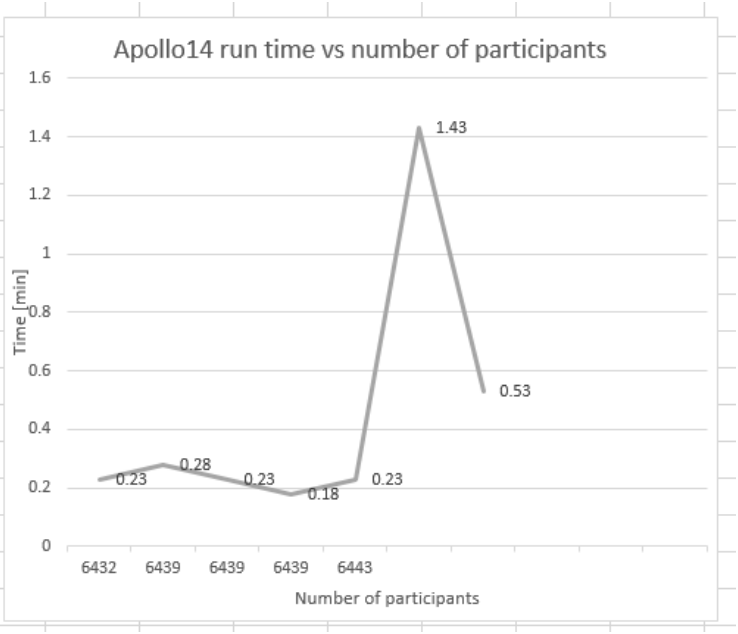

FIGURE 5 DURATION OF APOLLO14 RUN IN TEST ENVIRONMENT AFTER THE IMPROVEMENTS

CONCLUSION

An off-the-shelf cloud based clinical trials software like OpenClinica increased the speed of the trial software delivery but for a large scale trial like AscendPlus, with 20,000 participants, there were many other components needed to make the trial software system fulfil all our requirements. We complemented OpenClinica with several in-house built components. We described in this paper the basic configuration of the system, some of the challenges that were encountered and additional components that were built to overcome those challenges to produce an efficient system. An important if not critical step in making the system complete was the exchange of data between OpenClinica and in-house software and databases and for that reason this paper's emphasis was on the components that enabled the exchange. The trial is still in an early phase and software is still being changed and developed and we might still find areas for further improvements.

REFERENCES

ASCEND. (2021, November 15). *Long-term, low-dose aspirin did not affect risk of dementia in adults with type 2 diabetes*. Retrieved from A Study of Cardiovascular Events in Diabetes: https://ascend.medsci.ox.ac.uk/news/long-term-low-dose-aspirin-did-not-affect-risk-of-dementia-in-adults-with-type-2-diabetes

AscendPlus protocol. (2022, September 20). Retrieved from AscendPlus Trial: https://www.ascend-plus-trial.org/files/4-ascend-plus-protocol-v1-8-20-sept-2022-clean.pdf

OpenClinica. (2023). Retrieved from Openclinica: https://www.openclinica.com/

OpenClinica. (2023). *16 How and When to Use APIs*. Retrieved from OpenClinica Documentation and Training Site: https://docs.openclinica.com/oc4/how-and-when-to-use-apis/

Spring. (2023). Retrieved from Spring: https://spring.io/

ACKNOWLEDGMENTS

The author would like to express her gratitude to all colleagues involved in the project and particularly to James Thompson, the main developer responsible for Apollo14 and John Nolan for providing the first diagram of the system

CONTACT INFORMATION

Sonja Grotjahn
Oxford Population Health
Old Road Campus,
Richard Doll Building
Oxford, OX3 7LF
Email: sonja.grotjahn@ndph.ox.ac.uk
Web: https://www.ndph.ox.ac.uk/