A Distributed Analytics Platform to Execute FHIR-based Phenotyping Algorithms

Md. Rezaul Karim[1,2], Binh-Phi Nguyen[2], Lukas Zimmermann[3], Toralf Kirsten[4,5], Matthias Löbe[4], Frank Meineke[4], Holger Stenzhorn[3], Oliver Kohlbacher[3,6,7,8,9], Stefan Decker[1,2], and Oya Beyan[2,1]

[1] Fraunhofer FIT, Sankt Augustin, Germany
[2] Informatik 5, RWTH Aachen University, Aachen, Germany
[3] Institute for Translational Bioinformatics, University Hospital Tübingen, Germany
[4] University of Leipzig, Germany
[5] University of Applied Sciences Mittweida, Germany
[6] Dept. of Computer Science, University of Tübingen, Germany
[7] Quantitative Biology Center, University of Tübingen, Germany
[8] Center for Bioinformatics Tübingen, University of Tübingen, Germany
[9] Biomolecular Interactions, Max Planck Institute for Developmental Biology, Tübingen, Germany

Abstract. despite the benefits of reusing health data collected in routine care, sharing datasets outside of the organizational boundaries is not always possible due to the legal and ethical restrictions. The Personal Health Train (PHT) is a novel privacy-preserving approach to execute analytics tasks at distributed data repositories, without sharing the data itself. In this work, we report a proof-of-concept implementation of the PHT by using FHIR data standards and Clinical Query Language (CQL). The Semantic Web and containerization technologies have been utilized to move computations to the data. We developed tools to design phenotyping algorithms on the data consumer side, implemented an infrastructure to transfer and execute Docker containers at the data centers, and to return results to the consumers. We experimented the evaluated PHT infrastructure and tools by designing a phenotyping algorithm for diabetes mellitus and prostate cancer risk case-control study and executed it at three distributed FHIR repositories.

Keywords: Distributed analytics, Data reuse, Personal Health Train, Phenotyping, HL7 CQL, FHIR.

1 Introduction

Electronic Health Records (EHR), which is used in hospitals for routine health care offer a great potential and value for secondary uses such as research, quality of care, patient safety, and data-driven medicine. The reuse of EHRs is already widely accepted in observational studies (e.g. drug utilization, epidemiology) as well as in regulatory studies (e.g safety surveillance, pharmacovigilance), and partially accepted in clinical research (e.g. hypothesis generation, guideline adherence).

Emerging fields like patient recruitment, comparative effectiveness, and pragmatic trials also benefit from reusing EHRs [1]. Despite the traditional longitudinal studies with precise protocols, EHRs serve as a low-cost rich data source to build open cohorts in which patients may enter and leave at any time [2]. The conventional way of re-purposing EHR data is on-demand sharing of research data sets. However, since the curation process is quite labor-intensive and costly, eventually it impedes researchers to exploit the full benefit of EHRs. Moreover, the sensitive nature of the health data and EU General Data Protection Regulation (GDPR) [3] enforcements obstruct naive data sharing approaches.

Novel distributed learning approaches, such as the Personal Health Train (PHT)[1], provide an alternative to conventional data sharing. It introduces the concept of sharing analytical algorithms rather than data by executing tasks at the data source in a tightly regulated manner without revealing the primary data directly to the requesting data consumer [4]. In this train metaphor, stations are the data-containing repositories which can be discoverable and reusable, trains are the analytical task which could be queries, statistical models or algorithms, and tracks are the rules of interaction and underlying exchange infrastructure.

Recently, the German Federal Ministry of Education and Research has initiated the Medical Informatics Initiative (MII) funding concept[2] to support digitalization in medicine, and funded four large consortia (HiGHmed, DIFUTURE, MIRACUM, and SMITH). Each consortium develops an infrastructure to support cross-institution data exchange and implements a set of use cases to demonstrate data reuse for various purposes including research and data-driven medicine. The core element of the concept is the establishment of Data Integration Centers (DIC) at university hospitals and innovative ways to link data, information, and knowledge from health care, clinical, and biomedical research across the boundaries of sites[3]. Authors are the members of two of the funded MII consortia: SMITH [5] and DIFUTURE [6]. We developed an architecture and a reference implementation of the PHT approach to execute distributed analytics over the distributed data will be located in the DICs of the German MII. In this paper, we report a specific use case of the reference implementation, which focuses on exchanging the phenotyping algorithms between the DICs and perform statistical analysis for the selected cohorts. This implementation is based on HL7 standards[4] using Fast Healthcare Interoperable Resources (FHIR) resources and the Clinical Quality Language (CQL)[5].

In our research, we developed a FHIR standard-based distributed analytics platform tool by utilizing Semantic Web and containerization technologies. The developed platform has a user-friendly interface to generate CQL for further use in phenotyping algorithms, provides the infrastructure to send them to multiple DIC and interact with the FHIR resources to compute the requested analytics and return the outcomes to the data consumer.

The rest of the paper is structured as follows: section 2 briefly discuss related works. Section 3 describes the phenotyping algorithms. Section 4 presents the main PHT concepts, and Section 5 describes the proposed approach and the implementation. Section 6 outlines some initial evaluation results based on a sample use case. Finally, limitations of the study along with some future works is discussed before concluding the paper in section 7.

2 Related Work

In order to facilitate knowledge discovery for both humans and machines, FAIR data principles [7] have been proposed, which suggest a set of guiding principles

[1] `https://www.dtls.nl/fair-data/personal-health-train/`

[2] `http://www.medizininformatik-initiative.de/en/start`

[3] `https://www.bmbf.de/de/medizininformatik-3342.html`

[4] `http://www.hl7.org/implement/standards/`

[5] `http://www.hl7.org/implement/standards/product_brief.cfm?product_id`

to make research data Findable, Accessible, Interoperable, and Re-usable. These guiding principles are promising in the discovery, access, integration, and analysis of task-appropriate scientific data and associated algorithms and workflows.

The GoFAIR PHT implementation network initiative, which is an adoption of FAIR, aims to increase the reuse of existing biomedical data for research for personalized healthcare, preventive medicine, and value-based healthcare. Recently, Jochems et al. [8] and Deist et al. [9] proposed the PHT approach based on Semantic Web technologies. The underlying information system architecture enables learning from privacy-sensitive data without the data ever crossing organizational boundaries, maintaining control over the data, preserving data privacy and thereby overcoming legal and ethical issues common to other forms of data exchanges. The key concept in PHT is bring algorithms to the data rather than bringing data to a central repository, which gives controlled access to heterogeneous data sources while ensuring maximum privacy protection and engagement of individual patients.

Core to realizing both PHT and FAIR are Semantic Web technologies [10], which provides a framework for data sharing and reuse by making the semantics of data machine interpretable. On the other hand, as an HL7 specification, CQL aims to provide a rule-based way to define clinical quality measures and decision support rules. One of the key features of using CQL is that it makes logic expressions independent of any specific data models [11].

3 Use Case: Phenotyping Algorithms

Phenotyping algorithms used for identifying cohorts – a group of patients with certain characteristics [12] for purposes such as testing of novel therapies or recruitment for clinical trials. They can be described as rule sets, mostly represented as decision trees. These rule sets typically describe detailed patient characteristics and health conditions, and may include various data types, such as structured data, molecular data, and machine learning algorithms, such as text mining [13].

A number of studies have been published describing automated phenotyping techniques employed by medical organizations. However, owing to the sensitive nature of the data, most institutions have developed their own systems. The PHT approach can help to share and execute phenotyping algorithms at multiple data centers without sharing such privacy-sensitive data required for identifying cohorts.

In this study, we leveraged HL7 CQL to model and exchange phenotyping algorithms across DICs. The CQL provides a standard representation of rules and execution of phenotype algorithms required for machine-interpretable and exchangeable representation of phenotyping, which can be executed with FHIR resources. We provided a user-friendly web-based application to write phenotyping algorithms as CQL queries. With this tool, users can design their phenotype definitions on the fly, and specify any previously implemented algorithms or statistical models to be executed at the selected cohort.

4 Concepts

In this section, we present the main concepts of PHT and it's different components in the proposed architecture. The PHT approach is based on the principle that data does not leave the origin, rather than the analytics are transferred to the data sources by a gateway. Once the data analytics task is dispatched, the owner of the task is detached from the process, the gateway controls routing to the DICs and collecting the outcomes of the tasks. DICs, which are coupled with computation power, runs the tasks and returns the trained models or result to requester via the gateway. We define three core entities, namely Curator (Station), Consumer (Train) and Gateway (Handling station) as main components of the proposed architecture.

Curators provide data sets, metadata and required computational power to execute analytics task. They are conceptualized as *the Train Stations*. They integrate the privacy sensitive data from multiple sources in a secure storage that can be accessed by authenticated and authorized parties. They act as FAIR data points, by publishing schemas, metadata and access protocols. Our assumption is that the data is privately hosted across stations but the schema and metadata are public.

Train Stations play three main roles: (i) publishing the metadata and schemas, which are required for the discoverability of the data and for the definition of input parameters of the analytics task; (ii) providing an access mechanism and an interface to evaluate and execute the data queries, either to negotiate the availability of data sets or to extract the data to feed the analytics tasks; (iii) providing a secure enclave to execute Dokerized computations on the extracted data during the computation phase. In our implementation, DICs play the role of the Train Stations.

Consumers are real world persons or services who aim to access potentially privacy sensitive data resides at multiple repositories to execute analytics task e.g. statistical analysis, machine learning, cohort identification, record linkage. Consumers are responsible from defining their data requirements by formulating queries, implementing and containerizing their analytics tasks and specifying how the different results generated by each curator will be aggregated. Consumers build *Trains* and send them to the gateway.

A Train has four main components: (i) a Query, which will be executed at the Train Station to retrieved the data input for the analytics task; (ii) an Analytics algorithm, which encapsulates the main task in a container; (iii) an Aggregator, which defines the methods to aggregate and post process the results; and (iv) a Metadata, which describes and keeps track of a range of information, e.g. owner and purpose of the task, access rights, execution provenance.

The Gateway is a point of trust. It provides common interfaces to describe, transfer and execute trains, as well as supervise the transmission of them. In our implementation, the *Handling Station* acts as a Gateway between consumers and curators. The Handling Station provides protocols to communicate with consumers and stations. It receives and forwards trains, keeps a privacy preserving index of stations and provides a uniform and trustful execution interface to the stations. During the training process, it keeps the provenance of trains e.g. to which stations it has been forwarded, execution status. The handling station also has the role of a broker. It publishes aggregated data schemas and vocabularies to describe the data in the stations, and directs trains to the relevant stations.

5 Proposed Approach and Implementation

Figure 1 illustrates workflow of the proposed architecture and communications between the PHT components. Consumers build trains as Docker containers. Each container contains metadata about the train, a query (CQL) to be executed over FHIR resources and an algorithm written in a programming language such as Python or R. These three components are encapsulated in a Docker image and pushed to the Train Registry.

Train registry stores and publish Trains as images in different versions. When a new Train arrives, the Registry notifies the Train Router. The Router maintains the list of the Stations in the system, and dispatches the docker image to the stations. Stations pull the trains, execute them and push them back to Train Registry. Stations interact with trains via the standardized Train API, such as checking the requirements of the train and controlling the life cycle of the execution, e.g. timeouts, observation of machine resource consumption. Once all the scheduled training is completed, the consumer is notified and receives the outcome of the tasks. At the current implementation, aggregation of results has been performed at the consumer site.

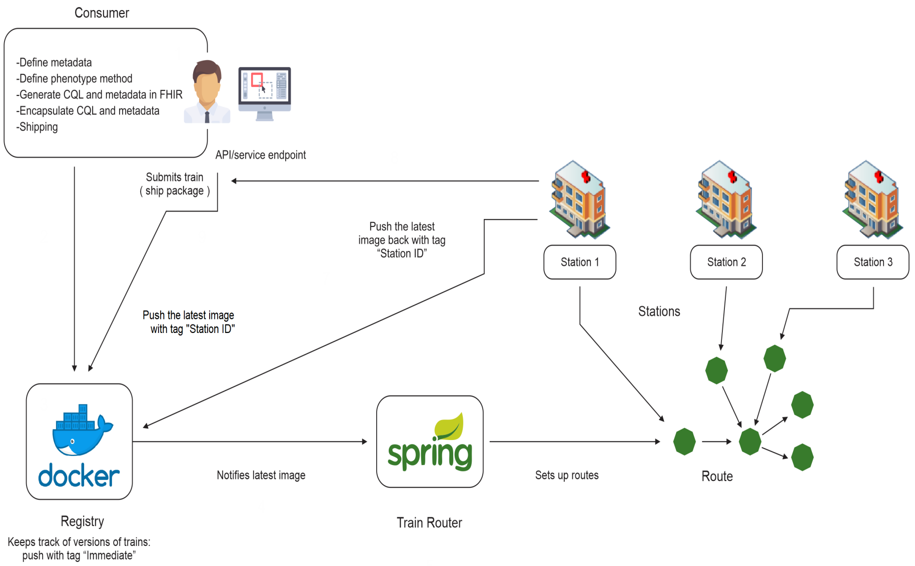
Consumer
-Define metadata
-Define phenotype method
-Generate CQL and metadata in FHIR
-Encapsulate CQL and metadata
-Shipping
API/service endpoint
Submits train
(ship package)
Push the latest image back with tag "Station ID"
Station 1
Station 2
Station 3
Stations
Push the latest image with tag "Station ID"
docker
Notifies latest image
spring
Sets up routes
Route
Registry
Keeps track of versions of trains: push with tag "Immediate"
Train Router

Fig. 1. Workflow of the proposed architecture

The current implementation is focused on methods to build trains and establishing protocols for exchanging them. It does not cover how data will be curated in each station and how various data formats will be transformed into agreed, consumable data standards. We assume there will be a FHIR server and CQL evaluation engine service in each data station, and data will be available as FHIR resources in JSON format.

We assume that DICs are trusted parties and they use FHIR as data representation standard. The current prof-of-concept implementation does not audit the containers, nor does control them to check if they leave with any privacy sensitive information. The following subsections will present the implementation details.

5.1 Data and metadata preparation

We generate synthetic data of about 2,500 patients using Synthea tool[1]. This enables us using the data without concern of legal or privacy restrictions. Patient records are based on a set of de-identified data recommended by the clinicians and real-world statistics collected by the CDC, NIH, and other sources.

Each patient is simulated independently from birth to present day and their diseases, conditions and medical care describing a progression of states and the

[1] https://synthetichealth.github.io/synthea/

transitions between them. Thus, for each synthetic patient, the data contains a complete medical history, including medications, medical encounters, and social determinants of health. We also introduced minor biases to patient observations -e.g. weight, height, blood pressure, heart rate etc. The condition and observation encoded with LOINC[1] and SNOMED[2], respectively. We convert patient data into a set of FHIR resources bundles based on observation and condition value sets and distribute them into three DICs to simulate different hospitals.

Metadata scheme is based on the condition[3] and observation[4] resource's examples provided by FHIR, the basic metadata information is collected by looking up corresponding examples in `Meaningful Use Value Sets` from the `United States Health Information Knowledge Base` [5]. The schema is first extracted inspired by the literature [4]. Then the resulting schema was further encoded into FHIR standard and publicly exposed using a dedicated FHIR server.

5.2 Train building

Consumers build trains by using a phenotype design client. This client provides a Web user interface to specify phenotyping algorithms, to generate CQL queries, and to create docker images containing the metadata, query and an algorithm. Query builder accesses the metadata deployed to a publicly accessible schema endpoint. Users then interact with this metadata and write their queries as CQL.

For the phenotyping artifacts, FHIR model version 3.0.0 is used as the primary data model to support for the FHIR STU3 standard within the library: all the artifact logic using CQL are wrapped in a library in the FHIR client. The data model supports a subset of resources including MedicationRequest, Observation, and Condition.Once the CQL is generated, it is sent to the CQL evaluation engine for syntactic validation before encapsulating and shipping.

Then consumers then generate a package containing the metadata of the phenotype algorithm, query and the script that specified the phenotype computation mechanism, and send to handler. During the training process, real-time progress can be tracked using the train monitor module and the resulting service API is activated, which listens to the changes, refresh the status and notify the users when there is a result returned from the station.

5.3 Train dispatching and routing

Once a package containing the metadata and CQL queries is shipped, it is wrapped into a Docker image, which is then pushed into a known Docker registry instance. This instance is known to all stations and the train issuer. The Docker tag designates the station that should pull the image next. We currently use the syntax `station.<id>`, where `<id>` is an integer that unambiguously identifies a station.

Each station continuously monitors the Docker registry to see whether it is supposed to pull an image, which should be running locally (which means executing the phenotyping algorithm using the local FHIR resources). Stations can be configured to scan a particular namespace of the Docker registry for new images. Currently, we use the Portus[6] authorization service and Docker registry

[1] `https://loinc.org/`

[2] `https://www.snomed.org/`

[3] `https://www.hl7.org/fhir/condition-examples.html`

[4] `https://www.hl7.org/fhir/observation-examples.html`

[5] `https://ushik.ahrq.gov/ValueSets?system=mu`

[6] `http://port.us.org/`

frontend to manage the authentication for the station. The User Interface (UI) of Portus also allows for monitoring the train images that are produced.

Routing is currently delegated to a simple Spring Boot Framework, which knows all present stations and hence can tag the Docker images accordingly. Once a station has processed an image, the Station pushes the newly created image containing the updated model to the Docker registry, which then sends a notification to the routing service. This service then determines the next station the train is supposed to visit and tags the pushed Docker image accordingly. The next station will then be able to find the image, which it should execute next.

5.4 Distributed phenotyping

Based on the scheduling suggested by the routing module, the train travels to different DICs, where the CQL query is executed inside a Docker container. First, the queries are evaluated and validated. Then the request is scheduled for processing within a secure enclave, where the query and algorithm are executed. Query returns a FHIR resource bundle from the FHIR servers of DICs. Once the data is available, phenotyping algorithms are executed on the data specified by the algorithm script inside the same Docker container.

The station is also responsible for updating train states depending on events and repeatedly status updates to the issuer by invoking the result service endpoint as `HTTP POST` requests without ever directly granting access to the data. After the computation is finalized, phenotype results and workflows are further pushed back to Docker registry, version of the Docker image is updated in the Docker registry, and station invokes the result service API and posts the results to the issue.

6 Evaluation

We have evaluated our developed approach by simulating a population based case control study from literature reported in [14] partially, which examines the risk of prostate cancer (PCa) among men with Type 2 Diabetes Mellitus (T2DM) [15]. In this study, prostate cancer risk categories among men with T2DM carefully characterized regarding glucose-lowering therapy, duration of disease, body mass index (BMI), and circulating levels of glycated hemoglobine (HbA1c).

This study showed a reduced risk of being diagnosed with PC among men with T2DM –especially for low risk tumors. Obese diabetic men (BMI¿30 kg/m2) showed a reduced risk compared to men without diabetes. We used this study as inspiring use case, and designed phenotyping algorithms to specify PCa and T2DM cohorts and calculated the BMI for the queried subpopulation at the each stations. We used the synthetic data created as FHIR resources and distributed into three stations (DICs) hosted at the University of Tübingen, RWTH Aachen University and an AWS EC2 cloud, respectively. Data contains approximately 400 PCa, and 120 T2DM cases. HAPI FHIR server used to host resources.

We asked users to define PCa and T2DM phenotypes by using diagnostic codes, observations such as HbA1C, and medication data. Via phenotype design client four phenotypes has been created and named as four arms of the study as follows: (i) PCa positive and T2DM negative; (ii) PCa positive and T2DM positive; (i)PCa negative and T2DM negative; (ii) PCa negative and T2DM positive. Each named phenotype is generated as CQL query and validated with the CQL engine service. Then in order to identify the cohort, for each arm, CQL is used to query the characteristics of each patient from FHIR server and then the service of CQL engine collects all the patient's information that satisfies the condition.

BMI is calculated for the arms, which includes T2DM positive cases. The BMI calculation algorithm is implemented in Python and selected by user via the web interface. Once the BMI algorithm is selected the required FHIR resources to query height and weight is included to the CQL query. Additionally, for each arm a count algorithm is included to return the number of patients selected for the specified phenotype.

```
library PhenotypeLibrary_Arm1 version ’0.0.1’

using FHIR version ’3.0.0’

codesystem "SNOMED": ’http://snomed.info/sct’
codesystem "LOINC": ’http://loinc.org’
codesystem "RXNORM": ’http://www.nlm.nih.gov/rxnorm’

valueset "Weight": ’vs-weight’
valueset "Height": ’vs-height’

code "CarcinomaOfProstate":’254900004’ from "SNOMED"

code "DiabetesMellitusType2":’44054006’ from "SNOMED"
code "DiabetesMellitusType1":’46635009’ from "SNOMED"

code "Insulin": ’139825’ from "RXNORM"
code "FastingGlucose": ’1558-6’ from "LOINC"
code "HemoglobinA1C": ’4548-4’ from "LOINC"

context Patient
```

Listing 1.1. Part-I: CQL query showing library, data model, context and terminology definition (code system, value sets, codes)

An example of generated CQL is shown in listing 1.1 and 1.2 As shown in the CQL, as a default and must-have statement to measure the population, the `InDemographic` statement is defined as a condition to characterize the disease in terms of patient characteristics observable from clinical data. Once user defined and validated the phenotyping algorithms, a Docker image has been build and published at the Docker registry. Each of the three stations monitoring the Docker registry pulls the image, performs the computing over the retrieved FHIR bundles, updates the image and pushes it back.

Table 1. Distribution of patients based on BMI weight status and counts

	BMI distribution	**Patient count**	
Arm 1	Underweight	15	
	Normal	23	**3.21%**
	Obese	35	
	Overweight	4	
Arm 2	Count	1913	**79.71%**
Arm 3	Underweight	3	
	Normal	5	**0.75%**
	Obese	6	
	Overweight	4	
Arm 4	Count	382	**15.92%**

Results and the status updates are then received through the result in service endpoint and aggregated there. Table 1 shows the distribution of the patients into different groups based on BMI.

```
define "InDemographic":
  "ProstateCancerPositive" and "Type2DMPositive"

define "ProstateCancerPositive": exists (
      [Condition] C where ToCode(C.code.coding)~"
         CarcinomaOfProstate")

define "Type2DMPositive": exists (
      [Condition] C where ToCode(C.code.coding) ~ "
         DiabetesMellitusType2") and not exists (
      [Condition] C where ToCode(C.code.coding) ~ "
         DiabetesMellitus1") or exists ([
         MedicationRequest] MR where ToCode(MR.
         medication.coding[0]) ~ "Insulin")
  and ToQuantity (Last ([Observation] O where ToCode(O.
     code.coding) ~ "FastingGlucose"
          sort by effective.value).value as Quantity
    ).value > 200 and ToQuantity (Last ([Observation] O
          where ToCode(O.code.coding) ~ "HemoglobinA1C"
          sort by effective.value
      ).value as Quantity).value >= 6.5
```

Listing 1.2. Part-II: CQL query showing the InDemographic and statement definitions for population and condition

7 Conclusion and Outlook

In this work, we presented a FHIR standard-based approach for distributed phenotyping to reuse EHRs for research purposes by using Semantic Web and Docker technologies. We developed a proof-of-concept implementation for the PHT approach to exchange computations, instead of sharing the data. We briefly presented the system architecture, concepts, implementation choices and the overall workflow. From the users perspective, our approach enables query formulation against privacy sensitive data sources and successive evaluation of that request in a secure enclave at the data provider's end.

Initial experiments for the distributed phenotyping on T2DM and PCa risk case-control use case show that our approach using CQL, FHIR, and Docker can overcome the reliance of previous approaches on agreeing upon shared schema and encoding a priori in favor of more flexible schema extraction based on FHIR standards. Further, we specified the main concepts of the PHT, such as train, station and gateway. Our current PHT implementation is limited to specify basic protocols to define trains and exchange them between stations. The first results are promising, however, requires extensions to achieve full power of PHT approach. Additional functionalities such as authentication and authorization, intelligent routing, trust services planned to be added in future.

Another limitation of the current work is the absence of the privacy preserving technologies such as differential privacy, encryption, secure computation. This proposed architecture could be extended with selected approach. In our work, we did not focus on the curation of data at the train stations. We assumed there is a data described with FHIR standards. There are numerous challenges to create the FHIR resources from the operational systems in hospitals. Additionally, there are various standards applied by other communities, such OMOP, OpenEHR and FHIR. Communication between these standards remains as another challenge to explore.

Acknowledgements

This work was supported by the German Ministry for Research and Education (BMBF) as part of the SMITH consortium (MRK, OB, and SD grant no. 01ZZ1803K; TK, ML, and FM grant no. 01ZZ1609A) and the DIFUTURE consortium (OK, HS, and LZ, grant no. 01ZZ1804D).

This work was conducted jointly by RWTH Aachen University, Tubingen University and Fraunhofer FIT as part of the PHT and GoFAIR implementation network, which aims to develop a proof of concept information system to address current data reusability challenges.

References

1. Cowie, M.R., Blomster, J.I., Curtis, Lesley H, e.a.: Electronic health records to facilitate clinical research. Clinical Research in Cardiology **106**(1) (2017) 1–9
2. Shields, C.L., Alset, A.E., Boal, Nina S, e.a.: Conjunctival tumors in 5002 cases. comparative analysis of benign versus malignant counterparts. American journal of ophthalmology **173** (2017) 106–133
3. Chassang, G.: The impact of the eu general data protection regulation on scientific research. ecancermedicalscience **11** (2017)
4. Gleim, L.C., Karim, M.R., Zimmermann, L., Kohlbacher, O., Stenzhorn, H., Decker, S., Beyan, O.: Schema extraction for privacy preserving processing of sensitive data. life sciences **1**(39) 48
5. Winter, A., Stäubert, S., Ammon, D., Aiche, S., Beyan, O., Bischoff, V., Daumke, P., Decker, S., Funkat, G., Gewehr, J.E., et al.: Smart medical information technology for healthcare (smith). Methods of information in medicine **57**(S 01) (2018) e92–e105
6. Prasser, F., Kohlbacher, O., Mansmann, U., Bauer, B., Kuhn, K.A.: Data integration for future medicine (difuture). Methods of information in medicine **57**(S 01) (2018) e57–e65
7. Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.W., da Silva Santos, L.B., Bourne, P.E., et al.: The fair guiding principles for scientific data management and stewardship. Scientific data **3** (2016)
8. Jochems, A., Deist, T.M., van Soest, e.a.: Distributed learning: Developing a predictive model based on data from multiple hospitals without data leaving the hospital – A real life proof of concept. Radiotherapy and Oncology **121**(3) (2016) 459–467
9. Deist, T.M., Jochems, A., van Soest, J., Nalbantov, G., Oberije, C., Walsh, S., Eble, M., Bulens, P., Coucke, P., Dries, W., Dekker, A., Lambin, P.: Infrastructure and distributed learning methodology for privacy-preserving multi-centric rapid learning health care: euroCAT. Clinical and Translational Radiation Oncology **4** (2017) 24–31
10. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific american **284**(5) (2001) 34–43
11. Jiang, G., Prud'Hommeaux, E., Xiao, G., Solbrig, H.R.: Developing a semantic web-based framework for executing the clinical quality language using fhir. CEUR-WS. org (2017)
12. Shivade, C., Raghavan, P., Fosler-Lussier, E.e.a.: A review of approaches to identifying patient phenotype cohorts using electronic health records. Journal of the American Medical Informatics Association **21**(2) (2013) 221–230
13. Löbe, M., Stäubert, S., Goldberg, C., Haffner, I., Winter, A.: Towards phenotyping of clinical trial eligibility criteria. Studies in health technology and informatics **248** (2018) 293–299
14. Pierce, B.L., Plymate, S., Ostrander, E.A., Stanford, J.L.: Diabetes mellitus and prostate cancer risk. The Prostate **68**(10) (2008) 1126–1132
15. Fall, K., Garmo, H., Gudbjornsdottir, S., Stattin, P., Zethelius, B.O.: Diabetes mellitus and prostate cancer risk; a nationwide case-control study within pcbase sweden. Cancer Epidemiology and Prevention Biomarkers (2013) cebp–1046