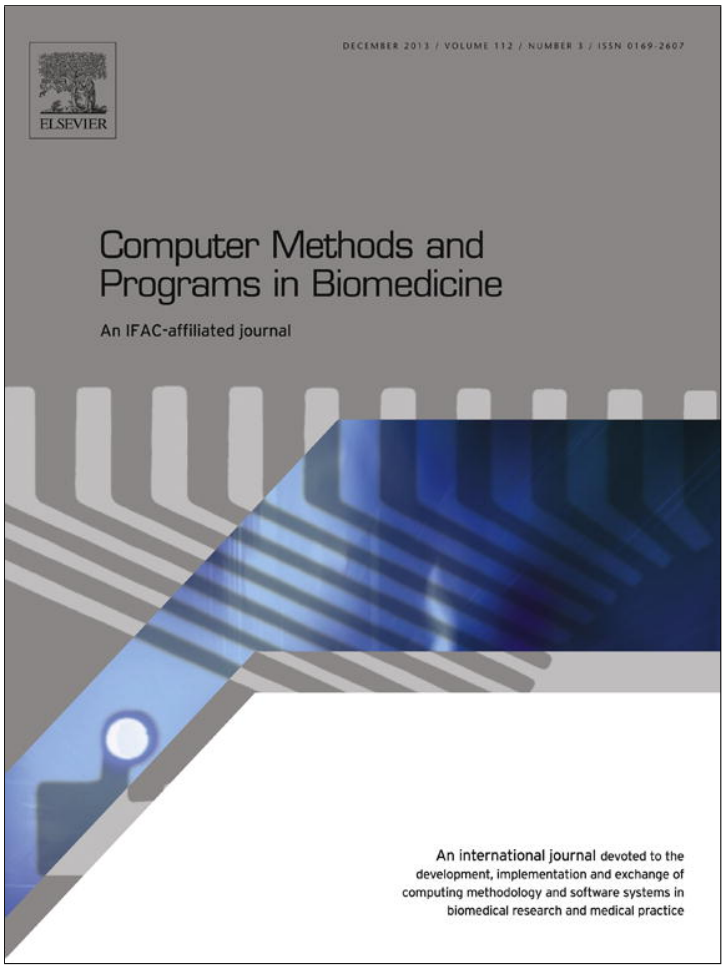


COMPUTER METHODS AND PROGRAMS IN BIOMEDICINE 112 (2013) 529–552

journal homepage: www.intl.elsevierhealth.com/journals/cmpb

# A new tool for sharing and querying of clinical documents modeled using HL7 Version 3 standard

*Vasil Slavov*[a], *Praveen Rao*[a,*], *Srivenu Paturi*[a], *Tivakar Komara Swami*[a], *Michael Barnes*[a], *Deepthi Rao*[b], *Raghuvarun Palvai*[a]

[a] *Department of Computer Science Electrical Engineering, University of Missouri-Kansas City, United States*
[b] *Department of Pathology and Laboratory Medicine, University of Kansas Medical Center, United States*

ARTICLE INFO

*Article history:*
Received 10 September 2012
Received in revised form
16 April 2013
Accepted 2 July 2013

*Keywords:*
HL7 v3
P2P
XML
Query processing
Security

ABSTRACT

We present a new software tool called CDN (Collaborative Data Network) for sharing and querying of clinical documents modeled using HL7 v3 standard (e.g., Clinical Document Architecture (CDA), Continuity of Care Document (CCD)). Similar to the caBIG initiative, CDN aims to foster innovations in cancer treatment and diagnosis through large-scale, sharing of clinical data. We focus on cancer because it is the second leading cause of deaths in the US. CDN is based on the synergistic combination of peer-to-peer technology and the extensible markup language XML and XQuery. Using CDN, a user can pose both structured queries and keyword queries on the HL7 v3 documents hosted by data providers. CDN is unique in its design – it supports *location oblivious queries* in a large-scale, network wherein a user does not explicitly provide the location of the data for a query. A location service in CDN discovers data of interest in the network at query time. CDN uses standard cryptographic techniques to provide security to data providers and protect the privacy of patients. Using CDN, a user can pose clinical queries pertaining to cancer containing aggregations and joins across data hosted by multiple data providers. CDN is implemented with open-source software for web application development and XML query processing. We ran CDN in a distributed environment using Amazon EC2 as a testbed. We report its performance on real and synthetic datasets of discharge summaries. We show that CDN can achieve good performance in a setup with large number of data providers and documents.



## 1. Introduction

Today, it is well agreed upon that through effective use of Information Technology (IT), health care costs can be reduced and better quality care can be delivered to patients. The US government is spending billions of dollars to promote the adoption of electronic health records and to develop Health Information Exchanges (HIEs) [46]. HIEs aim to enable "the electronic movement of health-related information across organizations according to nationally recognized standards" [46]. They are considered to be the building blocks for

* *Corresponding author at*: Department of Computer Science Electrical Engineering, University of Missouri-Kansas City, United States. Tel.: +1 816 235 2705.
E-mail addresses: vgslavov@mail.umkc.edu (V. Slavov), raopr@umkc.edu (P. Rao), sp895@mail.umkc.edu (S. Paturi), tk27f@mail.umkc.edu (T.K. Swami), mjb5a6@umkc.edu (M. Barnes), drao@kumc.edu (D. Rao), rp7x5@mail.umkc.edu (R. Palvai).
0169-2607/$ – see front matter © 2013 Elsevier Ireland Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cmpb.2013.07.002

Nationwide Health Information Network (NHIN) initiative [45] and are designed to achieving Institute of Medicine's (IOM) vision of a learning healthcare system [19]. Some of the established HIEs such as HealthBridge, CareSpark, Indiana Health Information Exchange, and MedVirginia serve up to few million patients and few thousand physicians, thereby, hosting large volumes of patient data [47].

Recently, "data sharing and collaboration" and "large scale management of health care data" have been identified as the key IT challenges to advance the nation's healthcare system [80]. This is because vast amounts of health-related information remain untapped due to the lack of suitable IT solutions. Personal health information resides in digital silos and healthcare systems do not easily share information with each other. However, by tearing down these silos, health-related information can be utilized by medical practitioners and researchers to provide efficient, quality, timely, and cost–effective care to patients.

The National Cancer Institute's caBIG is a nation-wide initiative, whose vision is to advance research on cancer and improve clinical outcomes for patients by connecting the members of the cancer community to share knowledge and data [32]. The caBIG community has more than 190 organizations [13]. Today, there are 124 participating institutes connected to caGrid – the underlying network infrastructure of caBIG. The community has shown great interest in sharing large amounts for biospecimen annotations, microarray data, cancer genome data (e.g., tissue samples), and so forth [14]. Such large-scale sharing of biomedical and clinical data is the first step towards collaborative e-science in the 21st century.

Achieving interoperability among applications processing clinical data has been a topic of interest for several years. Many advances have been made in developing standards for clinical data with regard to exchange/messaging, terminology, application, architecture, and so forth [53]. The standards from Health Level Seven International (HL7) have become popular for the exchange, integration, sharing and retrieval of electronic health information. HL7 standards are used by 90% of the hospitals in the US.[1] More recently, HL7 Version 3 standard was developed to enable *semantic interoperability* in healthcare data interchange [63]. (XML is used to encode the data.) The documents in HL7 v3 are derived from the Reference Information Model (RIM) and use terminologies such as SNOMED CT, LOINC, CPT and ICD-9. Software tools are available for modeling data using HL7 v3 standards (e.g., Model-Driven Health Tools [67], caAdapter [15], HL7 Tooling [66]).

We present a new software tool called CDN (Collaborative Data Network) for sharing and querying of clinical data modeled in HL7 v3 standard. Of particular interest to us are the HL7 CDA (Clinical Document Architecture) and CCD (Continuity of Care Document) standards. CDN is ideal tool for data providers (e.g., clinic, hospital, research lab) who wish to selectively enable data sharing and querying of HL7 v3 documents. While CDN is not restricted to a particular health condition, the GUI of CDN is designed for posing clinical queries related to cancer diagnosis and treatment. Cancer is the second most leading cause of deaths in the US. CDN *differs from the aim of HIEs in the sense that it is not designed for the electronic movement of health-related information across organizations.*

The remainder of the paper is organized as follows. Section 2 provides the background and motivations. Section 3 describes the novel architecture of CDN, the query processing approach, and security schemes in CDN. Section 4 describes the implementation and evaluation of CDN. We provide a discussion in Section 5 and conclude in Section 6.

This work has previously appeared in the conference proceedings of the 2010 and 2012 ACM SIGHIT International Health Informatics Symposium [68,71]).

## 2. Background and motivations

### 2.1. *The peer-to-peer model of computing*

We have witnessed a huge success of the P2P model of computing in the last decade. This has culminated in the development of Internet-scale applications such as Kazaa, BitTorrent, and Skype. P2P computing has also become popular in ecommerce and ebusiness and has lead to the development of many Internet-scale systems. Innovations in P2P computing, most notably the concept of Distributed Hash Table (DHT) (e.g., Chord [81], Pastry [76], CAN [72], Tapestry [90], Kademlia [62]), has been embraced by key-value stores of production quality such as Dynamo [24], Cassandra [56], and Voldemort [58]. DHT-based systems have good scalability, fault-tolerance, and load balancing properties. Because of these useful properties, CDN employs a DHT for indexing HL7 v3 documents and locating relevant documents during query processing.

### 2.2. *XML and distributed XQuery*

The extensible markup language XML has become the de facto standard for information representation and interchange on the Internet. It is widely adopted in a variety of domains ranging from ecommerce to health informatics. XQuery is a popular query language for XML and is recommended by the W3C. It is a functional language that subsumes XPath – a query language for selecting qualifying nodes in an XML document. XQuery allows for the creation of new elements and attributes and the specification of their contents and relationships. Queries in XQuery can contain `for`, `let`, `where`, `order by`, and `return` clauses and are frequently called FLWOR expressions.

A great deal of work has been done in the area of distributed query processing [55]. There are three well-known approaches to processing a distributed query, namely, pure data shipping, pure query shipping, and hybrid shipping. Neither pure data shipping nor pure query shipping are the best choices in all scenarios in a distributed setting and a hybrid approach has shown to perform better [55]. Distributed XQuery processing [73,33,89,34,29,25,38] has been studied in recent years. The underlying principle is to ship portions of a query to remote servers which then execute them. Locations of remote servers are specified in the query. These previous solutions were not designed for a P2P network, where the locations of relevant data of interest may not be known apriori. In contrast, CDN

[1] http://www.itl.nist.gov/div897/docs/Message_Maker.html.

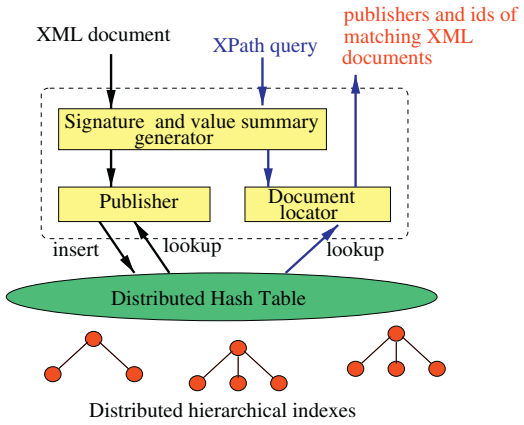



**Fig. 1 – Architecture of *psi*X.**

differs from previous techniques as it supports *location oblivious queries*.

Due to the popularity of P2P systems, several approaches were developed to find/locate relevant XML documents and their publishers in a P2P environment [37,54,4,20,70]. Of particular interest to us is the *psi*X system [70,69], which is used in CDN to process location oblivious queries. The *psi*X system is an Internet-scale location service for XML documents. Using *psi*X, participants can publish their XML documents so that others can query them. The original documents reside with the owners and summarized representations of documents are indexed in a distributed fashion using a DHT. Any participant can issue an XPath query and *psi*X will locate all the XML documents (and their publishers) that contain a match for the query.

Fig. 1 illustrates the architecture of *psi*X and the different components within it. The first main component is the Signature and Value Summary Generator. An XML document is mapped to its signature. This signature is essentially a bit string and can capture the document's structural properties and content. To compute the signature of a document, we summarize the document into a graph, and assign irreducible polynomials (from a finite field [8]) carefully to the edges of this graph. After that we compute the (algebraic) signature by taking the product of the assigned irreducible polynomials. (Irreducible polynomials are like prime numbers in a finite field). An XPath query is also mapped to its signature. A useful necessary condition of *psi*X's signature representation method is that *if a document contains a match for a query, then the query signature (algebraically) divides the document signature*.

The next main component is the Publisher. It builds and maintains distributed, hierarchical indexes on the signatures of published documents. By design, similar documents have similar signatures and can be grouped together into fewer index nodes to reduce the search time. The indexes allow us to efficiently test the aforementioned necessary condition to find matching documents. The nodes of the indexes are stored (as key-value pairs) by different peers in the underlying DHT network. This enables *psi*X to load balance and scale.

The last main component is the Document Locator. To locate relevant XML documents for a given XPath query, the signature of the query is used to traverse and search an index for matches. The traversal begins from the root of the index and is similar to accessing a hierarchical index such as an R-tree. Because the index nodes are distributed across peers, the peer where the query is issued uses DHT APIs to contact appropriate peers during index traversal. As *psi*X is built over a DHT, it inherits the scalability, fault-tolerance, and load balancing properties of the DHT. Note that the original XML documents are never stored by *psi*X; this provides complete control to the owners of the documents.

### 2.3. *Semantic interoperability in healthcare data interchange*

Achieving interoperability between healthcare systems has been one of hardest challenges in medical informatics. The HL7 v3 standard (e.g., CDA R2 [28]) aims to provide incremental semantic interoperability and therefore, HL7 v3 documents can evolve over time. An adopter can start with minimal structure in the HL7 v3 documents and over time add more structure to the documents and code content using standard terminologies such as SNOMED CT, ICD-9, and LOINC. Different XML schemas can be designed by the data providers to model their data as long as the schemas are derived from the Reference Information Model (RIM).

A data provider is free to choose a standard clinical terminology for coding clinical concepts. With the availability of methods to map codes between standard terminologies [3,2], it is possible to search for the same clinical condition coded using different terminologies. However, within the same standard terminology, the same concept can be coded in different ways [7,6]. For example, there is more than one way to represent concepts in SNOMED CT through pre-coordination or post-coordination of codes. While CDN is designed to enable sharing and querying of HL7 v3 clinical documents in a distributed environment, it is currently incapable of determining the equivalency between different codes (e.g., for pre-coordinated and post-coordinated concepts) during query processing.

### 2.4. *Data integration systems*

A survey has shown that many state-level HIEs are based on a federated database model [41]. BIRN [52] is one of the early initiatives for large-scale sharing and collaboration of biomedical data (e.g., neuroimaging data). SHRINE [86,85] is a federated querying tool for aggregating data from multiple sites. Currently, SHRINE does not support joins. FURTHeR [60,59] is a federated querying tool for heterogeneous data sources owned by multiple organizations. It emphasizes on OSGi-based development (http://www.osgi.org). FURTHeR has been used to query two live data sources and is integrated with the i2b2 web frontend. FURTHeR does not support data sources modeled using the HL7 v3 standard. Recently, the Cross-Institutional Clinical Translation project developed a federated query tool to facilitate clinical trial cohort discovery [5] across three academic medical centers. The tool was developed by adapting the software from the i2b2 project. The

aforementioned tools use a federated architecture, where only aggregate counts are returned by the data providers when queried, to comply with state and federal laws [85].

NCI's caBIG [32] also uses a federated database model. It's underlying network infrastructure, called caGrid [77], is a model-driven, service oriented architecture, and the data services are accessed via grid services that expose data sources in well-documented, interoperable form. To the best of our knowledge, among the aforementioned federated systems, only caGrid [77] supports XML queries over native XML data sources.

Recently, HIWAS [48,49] was developed for analyzing clinical data represented as complex XML documents (e.g., HL7 CDA). It summarizes clinical documents using the Semantic Data Guide to facilitates faster access to relevant information. CDN differs from HIWAS in the sense that the original HL7 v3 documents are queried directly by users. Moreover, CDN operates in a distributed environment.

### 2.5. Symmetric-key and public-key cryptography

Today, there are two classes of cryptographic techniques: symmetric-key and public-key cryptography. In symmetric-key cryptography, the same key is used to encrypt and decrypt a message. Thus, a sender and a receiver must share a common key to securely exchange messages. Advanced Encryption Standard (or AES) is a widely used symmetric-key cryptographic technique [22]. To overcome the requirement of sharing a common key between a sender and a receiver, public-key cryptographic techniques were developed [26,75]. The sender generates a public and private key pair. The private key is known only to the sender, but its public key is made public. The sender uses its private key to encrypt a message. The receiver uses the public key of the sender to decrypt its message. RSA is a widely used public-key cryptographic technique [75]. On large messages, RSA is slower than AES, and therefore, it is common to use a combination of RSA and AES on them [74]. We employ this strategy in CDN during query processing.

### 2.6. Motivations

The design of CDN is motivated by the following limitations of connecting heterogeneous XML data sources via a service-oriented architecture (e.g., as in caGrid): (a) the inability to express complex queries effectively using XQuery, and (b) the lack of fine-grained selection of data sources.

Consider a query in caGrid to *find all the expression data where there are at least 50 conditions for genes found in the vacuole* shown in Fig. 2. (This example is taken from Summary and Initial Recommendations draft available on the website of caBIG [91].) The query performs joins across data exposed by three data services `Gene`, `GeneOntology`, and `Microarray`. Suppose we want a query to access `Gene` and `Microarray` data from all possible data providers to perform the join. Then multiple queries should be posed – each one for a particular combination of `Gene` and `Microarray` data service – and therefore, will lead to poor scalability and performance when the number of data services grow.

```
FOR  $gene IN service
("http://cabio.osu.edu/GeneService.wsdl")/Gene,
$go IN service
("http://cabio.osu.edu/GeneOntologyService.wsdl")/GeneOntology,
$microarray IN service
("http://caarray.duke.edu/caArrayService.wsdl")/Microarray
LET $subject := $microarray/experiment/subject
WHERE
  $go/term='vacuole' AND $gene/goAcc=$go/acc AND
  $gene/gbAcc=$microarray/data/geneId AND
  count($microarray/data[geneId=$gene/$gbAcc]/condition)>50
RETURN
<subject>
  <subjectId>{ $subject/lsid }</subjectId>
  <species>{ $subject/species }</species>
  <microarrayData>
    { $microarray/data }
  </microarrayData>
</subject>
```

**Fig. 2 – An XQuery query in caGrid.**

CDN overcomes this limitation by constructing a *location oblivious query*. This is done by replacing `service` ("http://...GeneService.wsdl") with the phrase `collection` ("CDN") and replacing `service`("http://...MicroarrayService.wsdl") with the phrase `collection`("CDN") in the original query. Now the query specifies a join over multiple data sources without specifying the locations of `Gene` and `Microarray` documents distributed across a network of participating data providers. From a user's perspective, a single query is posed, rather than a potentially large number of queries with location information (or data services). From a system's perspective, fewer queries need to be processed.

Consider another query that performs aggregation over all or multiple data sources. Suppose the query has multiple selection predicates with a Boolean AND operation (e.g., gender = "female" AND smoker = "yes" AND age > 35). In a service-oriented environment like caGrid, the query will be shipped to each data source (assuming the data service name is known), but only a few may contain data that satisfies all the selection predicates. It is, therefore, effective to identify those data sources that contain matching data for all the selection predicates and to ship the query to only those data providers. CDN aims to achieve such *fine-grained selection of data sources* through the indexing power of *psi*X. The benefit is clear: the number of queries issued in the network can be reduced and resources such as network bandwidth can be saved.

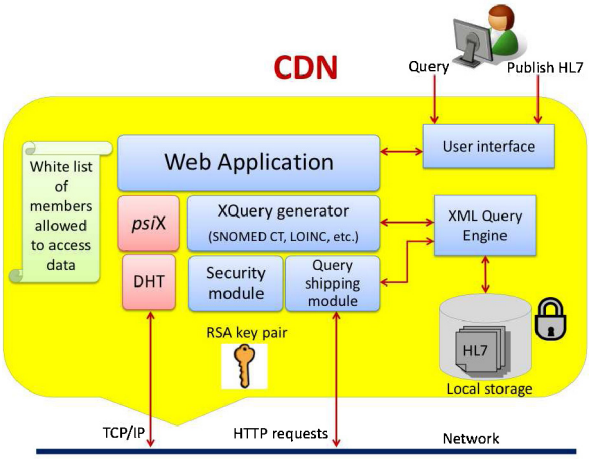



**Fig. 3 – Key components of CDN.**

```
Q1: How many male patients had colon cancer in the target population?

count (
 for $x in collection("CDN")/ClinicalDocument
 where $x/RecordTarget/PatientRole/Patient = "M" and
  ($x//observation/code[@codeSystem="2.16.840.1.113883.6.96"][@code = "315058005"] or
   $x//procedure/code[@codeSystem="2.16.840.1.113883.6.96"][@code = "315058005"])
 return $x/RecordTarget/PatientRole/ID
)
```

(a) XQuery query for Q1 over coded content in the HL7 v3 documents

```
/ClinicalDocument[RecordTarget/PatientRole/Patient = "M"][RecordTarget/PatientRole/ID]
         //observation/code[@codeSystem="2.16.840.1.113883.6.96"][@code = "315058005"]

/ClinicalDocument[RecordTarget/PatientRole/Patient = "M"][RecordTarget/PatientRole/ID]
         //procedure/code[@codeSystem="2.16.840.1.113883.6.96"][@code = "315058005"]
```

(b) Two maximal XPath expressions for Q1

```
(: ********** Query template A ********* :)
for $x in doc("...")/ClinicalDocument
   where $x/RecordTarget/PatientRole/Patient = "M" and
         $x//observation/code[@codeSystem="2.16.840.1.113883.6.96"][@code = "315058005"]
  return <res> {$x/RecordTarget/PatientRole/ID} </res>

(: ********** Query template B ********* :)
for $x in doc("...")/ClinicalDocument
   where $x/RecordTarget/PatientRole/Patient = "M" and
         $x//procedure/code[@codeSystem="2.16.840.1.113883.6.96"][@code = "315058005"]
  return <res> {$x/RecordTarget/PatientRole/ID} </res>
```

(c) Templates of queries shipped to data providers with matching documents

```
count ( distinct-values ( for $x in doc("results.xml")//ID return $x ) )
```

(d) Counting and duplicate elimination performed locally

**Fig. 4 – Step-by-step evaluation of a location oblivious XQuery query.**

While we emphasize on performance and efficiency, our work aligns with the need to manage semantic heterogeneity among clinical data sources. CDN promotes the sharing and retrieval of clinical documents modeled using HL7 v3 standards, and these standards are precisely designed to enable semantic interoperability between data sources.

## 3. The architecture of CDN

In this section, we present the novel architecture of CDN, which aims to achieve the following design goals:

- to enable scalable sharing and querying of HL7 v3 clinical documents in a distributed environment, where a data provider has complete control and ownership of its data like in a federated model,
- to leverage the power of XML technologies and the P2P model of computing to efficiently process a "location oblivious query" in CDN, wherein the locations of relevant data in the network (for a query) are discovered during query processing,
- to provide high level of security to data providers and protect the privacy of patients for HIPAA compliance using standard cryptographic techniques, and
- to leverage open-source software to develop CDN and facilitate easy deployment.

Occasionally, we use the terms "**data provider**," "**publisher**" and "**participant**" interchangeably to mean the same.

### 3.1. *System overview*

Given a network of data providers, each data provider runs a copy of the CDN software much like an Internet user who

installs and runs software such as Kazaa and Skype. The data providers are connected through a network such as the Internet or a Virtual Private Network (VPN). Each CDN software communicates with other CDNs in the network to process a user's request.

The key components of the CDN software are shown in Fig. 3. The user interface accepts requests from a user (authorized by the data provider) to either publish HL7 v3 documents or pose queries. Documents that are made available for sharing with other data providers are stored in a local database, which may be encrypted for security reasons. At the heart of the CDN software, is a web application containing the XQuery generator, the Query Shipping module, the Security Module, and the location service called *psi*X built atop a DHT. CDN employs a restricted form of the *hybrid shipping approach* [55] for processing queries. The Query Shipping Module is responsible for shipping the subqueries to relevant data providers and storing the returned results. An open source XQuery processor executes the queries. These queries are either subqueries shipped from other data providers or queries generated by the XQuery generator to process the results obtained from other data providers (e.g., join processing). The actual HL7 v3 documents are never transferred across the network. Each CDN maintains a RSA public/private key pair, which is used by the Security Module for authentication and secure communication during query processing. Each CDN maintains a white list of data providers/participants in the network that are allowed to access its local data. (One way is to maintain the public keys of allowed data providers in the white list.)

### 3.2. *Publishing HL7 V3 clinical documents*

CDN allows any valid HL7 v3 clinical document to be published by a data provider and therefore, a data provider is expected to do minimal standardization of the documents. A data provider may decide to share only the "Past Medical History" and "Physical Examination" from deidentified discharge summaries of some patients. (By using HL7 v3, we are in fact enforcing some standardization.) By "publishing a document", we mean that the data provider stores the document in its local database and the document becomes ready to be queried by other data providers. *How do other data providers become aware of this document?* The answer is through the location service called *psi*X [70,69], which is based on a novel, distributed XML indexing technique for DHT-based P2P networks. It is important to note that a document owned by a data provider resides locally and is never exchanged or transferred through the network. The data provider has full ownership and control of its data and can implement local access control policies similar to a federated system.

**Algorithm 1.** Publishing a HL7 v3 document
**proc** `publishDocument`(document *d*)
1: Store the document *d* in the local database
2: Compute the signature *s* for *d* as described in *psi*X [70]
3: Construct the *docid* for *d* by concatenating the hostname of the data provider, the local id of *d*, and the data provider's public key
4: Index (*s*, *docid*) using *s* as the key by invoking *psi*X
**endproc**

Algorithm 1 shows the sequence of steps involved in publishing a HL7 v3 document. First, the document is stored in the local database. Then the signature of the document is generated. The signature is indexed by invoking *psi*X and along with it the hostname of the data provider, the local id of the document, and the data provider's public key is stored. By knowing the hostname of a data provider and the local id of a document owned by that data provider, a participant in the network can ship a query to it for execution. The public key is necessary for secure communication during query processing. (The discussion of the security schemes employed by CDN is deferred until Section 3.4.)

### 3.3. *Processing XQuery queries*

Next, we describe the steps involved in processing an XQuery query. We use the term "query initiator" to refer to the data provider where query is posed by a user. In the context of sharing clinical data, we have developed a restricted form of hybrid shipping approach to ensure that effective security and privacy policies can be implemented for HIPAA compliance. There are some limitations of pure data shipping and pure query shipping. If pure data shipping were employed, then an entire HL7 v3 document would have to be transferred across the network to the query initiator and the query initiator would have complete access to the document. If pure query shipping were employed, then the participating data providers would have to exchange results of shipped queries amongst each other (e.g., in case of join operations). This may not be desirable. In the hybrid approach adopted by CDN, joins are always executed locally by the query initiator. The selection and projection operations in the query on a single document are always executed by the data provider owning the document. (Which parts of the data can be projected, will depend on what the data provider wishes to expose.) Aggregation and duplicate elimination can be done either by the query initiator or remotely by a data provider depending on the query.

**Algorithm 2.** Query processing at the query initiator

**proc** `processXQuery`(location oblivious XQuery query *q*)
1: Compute the maximal XPath expressions in *q* by analyzing the XPath expressions in the FOR, WHERE, and RETURN clauses of *q*
2: **foreach** *maximal XPath expression p in q* **do**
3: Send *p* to *psi*X to get the (docid, publisher) pairs for all documents that contain a match for *p*
4: **foreach** *publisher returned by* psi*X* **do**
5: Create one XQuery query per matching document to do selections and projections and ship the entire list of queries to the publisher
6: Merge the results from the publishers (after decryption) and store it in a single temporary XML document locally
7: Construct an XQuery query to operate on the temporary XML documents to perform operations such as joins, aggregation, and duplicate elimination
8: Return results to the user
**endproc**

Algorithm 2 shows the steps taken by the query initiator to process a location oblivious XQuery query. First, *maximal XPath*

*expressions* are extracted from the query by examining the XPath expressions in the FOR, WHERE, and RETURN clauses (Line 1). *We define a maximal XPath expression as the longest XPath expression that should be matched in an XML document to generate correct results.* For each maximal XPath expression, *psi*X is invoked to obtain the (docid, publisher) pairs for further processing (Line 4). Next, for each publisher identified by *psi*X, an XQuery query is created on one matching document owned by that publisher/data provider to do selections and projections. The entire list of such queries is sent all at once to that publisher (Line 5). The returned results are stored in temporary XML files (Line 6) and finally, local processing is done (Line 7).

Algorithm 3 shows the steps taken to process queries shipped to a data provider. First, the receiving data provider authenticates the query initiator using its white list containing public keys of authorized query initiators and public key cryptography. If the authentication succeeds, then the shipped queries are executed and the results are encrypted and returned to the query initiator. (The details of encryption and decryption steps during query processing is discussed in Section 3.4.)

**Algorithm 3.** Processing of a shipped query

**proc** `process Shipped Queries`(list of queries $\overline{q}$)

1: Authenticate the query initiator by checking the white list of allowed data providers
2: **if** *query initiator is authorized* **then**
3: Execute the queries $\overline{q}$ on the local HL7 database
4: Encrypt the results and return the results
**else**
5: Do not execute $\overline{q}$ and reject further processing
**endproc**

#### 3.3.1. *Basic aggregation queries*

We present a few examples of location oblivious XQuery queries with basic aggregation operations. These queries examine both coded content as well as textual content in the HL7 v3 documents. In the following discussions, we assume that the HL7 documents are modeled using Clinical Document Architecture (CDA) R2 over deidentified patient data. (In our evaluation, we used deidentified discharge summaries from the NLP research data sets available from the i2b2 project [84] and modeled them as CDA documents with sections, namely, History of Present Illness, Physical Examination, Past Medical History, Past Surgical History, Allergies, Hospital Course, Discharge Date, Discharge Diagnosis, and Discharge Disposition.)

Fig. 4(a) shows a location oblivious XQuery query for a clinical query Q1 *to count the number of male patients who have had colon cancer.* Two maximal XPath expressions are extracted from the query and are shown in Fig. 4(b). Based on these expressions, two different query templates are used to generate the queries shipped to the data providers containing matching documents for each of the maximal XPath expressions. These are shown in Fig. 4(c). (As an optimization, we ship a list of queries at once to a data provider, one each on a matching document from that data provider.) Finally, counting and duplicate elimination is applied at the query initiator on the results of the shipped queries. This is shown in Fig. 4(d).

A few more examples of clinical queries in `CDN` are shown in Figs. 5 and 6. Because HL7 v3 is designed for incremental semantic interoperability, it is necessary to have queries process both coded content as well as textual content for better coverage of the data. For instance, in the query shown in Fig. 5(a), the term "alopecia" is searched using the SNOMED CT code within `observation` as well as within the textual content under "Physical Examination." In the query shown in Fig. 6(a), "small cell lung cancer" is searched using the SNOMED CT code within `observation` and `procedure` and the term "smoker" is searched within the textual content under "History of Present Illness."

`CDN` also supports keyword queries over HL7 v3 documents. Fig. 7 shows a query that finds the number of patients who had past medical history of "anemia." A user is expected to provide the keyword and select the category wherein the textual content should be searched (e.g., "Past Medical History", "Hospital Course"). The XQuery query for Q4 is shown in Fig. 7(a). This query has one maximal XPath expression, which is shown in Fig. 7(b). Figs. 7(c) and (d) show the template of the query shipped to data providers containing relevant documents and the local XQuery query executed by the query initiator, respectively.

#### 3.3.2. *Aggregation queries with join operations*

Next, we present an aggregation query with a join operation that `CDN` can execute. Consider the query Q5 shown in Fig. 8(a) to *find the number of patients who were given medications during hospital course that have caused an allergy in one or more patients.* The join operation is on the attribute `ApplicationNumber` of the medications coded under "Allergies" and "Hospital Course" in different discharge summaries. The two maximal XPath expressions for the query are shown in Fig. 8(b). Two templates for the queries shipped by the query initiator to the data providers containing matching documents are shown in Fig. 8(c). (Query template A is for the first maximal XPath expression and template B is for the second maximal XPath expression.) The partial results from the shipped queries (generated from query template A) are concatenated and stored locally in a temporary file say `A.xml`. The partial results from the shipped queries (generated from query template B) are concatenated and stored locally in a temporary file say `B.xml`. Finally, the query initiator (locally) performs the join operation on the attribute `ApplicationNumber` in `A.xml` and `B.xml`, followed by duplicate elimination and counting. This is shown by the query in Fig. 8(d).

### 3.4. *Security module*

`CDN` provides high level of security to data providers and protects the privacy of patient data to ensure HIPAA compliance. `CDN` uses standard cryptographic techniques to achieve this. Similar to a federated database model, each data provider has complete control of its data and exposes only those that it wishes to share (e.g., portions of deidentified discharge summaries of certain patients). Each data provider running `CDN`

```
Q2: How many patients developed alopecia as a side-effect of chemotherapy in the target
population?
count (
 for $x in collection("CDN")/ClinicalDocument
 where $x//procedure/code[@code="150415003"][@codeSystem="2.16.840.1.113883.6.96"] and
  ($x//observation/code[@code="270504008"][@codeSystem="2.16.840.1.113883.6.96"] or
   $x//section[code/@code="29545-1"][code/@codeSystem="2.16.840.1.113883.6.1"]/text[contains(.,
   "alopecia")])
 return $x/RecordTarget/PatientRole/ID
)
```

(a) XQuery query for Q2

```
/ClinicalDocument[RecordTarget/PatientRole/ID]
                 [.//procedure/code[@code="150415003"][@codeSystem="2.16.840.1.113883.6.96"]]
                  //observation/code[@code="270504008"][@codeSystem="2.16.840.1.113883.6.96"]

/ClinicalDocument[RecordTarget/PatientRole/ID]
                 [.//procedure/code[@code="150415003"][@codeSystem="2.16.840.1.113883.6.96"]]
                 //section[code/@code="29545-1"][code/@codeSystem="2.16.840.1.113883.6.1"]
                 /text[contains(.,"alopecia")])
```

(b) Two maximal XPath expressions for Q2

```
(: ********** Query template A ********** :)
for $x in doc("...")/ClinicalDocument
where $x//procedure/code[@code="150415003"][@codeSystem="2.16.840.1.113883.6.96"] and
      $x//observation/code[@code="270504008"][@codeSystem="2.16.840.1.113883.6.96"]
return <res> {$x/RecordTarget/PatientRole/ID} </res>



(: ********** Query template B ********** :)
for $x in doc("...")/ClinicalDocument
where $x//procedure/code[@code="150415003"][@codeSystem="2.16.840.1.113883.6.96"] and
  $x//section[code/@code="29545-1"][code/@codeSystem="2.16.840.1.113883.6.1"]/text[contains(.,
  "alopecia")]
return <res> {$x/RecordTarget/PatientRole/ID} </res>
```

(c) Templates of queries shipped to data providers containing relevant documents

```
count ( distinct-values (for $x in doc("results.xml")//ID return $x) )
```

(d) Counting and duplicate elimination performed locally

**Fig. 5 – A location oblivious query that processes both coded content as well as textual content.**

generates a RSA public/private key pair. Each data provider also maintains a white list of data providers that can access its data (e.g., the white list can contain public keys of those data providers).

When *psi*X is invoked to first identify relevant data providers and their documents, the public keys of the data providers are also returned. (Recall that along with the document's signature, the public key of the data provider is stored by *psi*X (Algorithm 1).) When the query initiator contacts a data provider that owns the matching documents, the data provider first verifies the identity of the query initiator. Essentially, the query initiator computes the hash of a message

```
Q3: How many cases of small cell lung cancer are noted among smoking females in the target
population?
count (
 for $x in collection("CDN")/ClinicalDocument[RecordTarget/PatientRole/Patient = "F"]
 where ($x//procedure/code[@code="254632001"][@codeSystem="2.16.840.1.113883.6.96"] or
   $x//observation/code[@code="254632001"][@codeSystem="2.16.840.1.113883.6.96"]) and
   $x//section[code/@code="10164-2"][code/@codeSystem="2.16.840.1.113883.6.1"]/text[contains(.,
   "smoker")]
 return $x/RecordTarget/PatientRole/ID
)
```

(a)

```
/ClinicalDocument[RecordTarget/PatientRole/ID][RecordTarget/PatientRole/Patient = "F"]
                 [.//procedure/code[@code="254632001"][@codeSystem="2.16.840.1.113883.6.96"]]
                 //section[code/@code="10164-2"][code/@codeSystem="2.16.840.1.113883.6.1"]
                 /text[contains(.,"smoker")]

/ClinicalDocument[RecordTarget/PatientRole/ID][RecordTarget/PatientRole/Patient = "F"]
                 [.//observation/code[@code="254632001"][@codeSystem="2.16.840.1.113883.6.96"]]
                 //section[code/@code="10164-2"][code/@codeSystem="2.16.840.1.113883.6.1"]
                 /text[contains(.,"smoker")]
```

(b) Two maximal XPath expressions for Q3

```
(: ********** Query template A ********** :)
for $x in doc("...")/ClinicalDocument[RecordTarget/PatientRole/Patient = "F"]
where $x//procedure/code[@code="254632001"][@codeSystem="2.16.840.1.113883.6.96"] and
  $x//section[code/@code="10164-2"][code/@codeSystem="2.16.840.1.113883.6.1"]/text[contains(.,
  "smoker")]
return <res> {$x/RecordTarget/PatientRole/ID} </res>

(: ********** Query template B ********** :)
for $x in doc("...")/ClinicalDocument[RecordTarget/PatientRole/Patient = "F"]
where $x//observation/code[@code="254632001"][@codeSystem="2.16.840.1.113883.6.96"] and
  $x//section[code/@code="10164-2"][code/@codeSystem="2.16.840.1.113883.6.1"]/text[contains(.,
  "smoker")]
return <res> {$x/RecordTarget/PatientRole/ID} </res>
```

(c) Templates of queries shipped to data providers containing relevant documents

```
count ( distinct-values (for $x in doc("results.xml")//ID return $x) )
```

(d) Counting and duplicate elimination performed locally

**Fig. 6 – Another location oblivious query that processes both coded content as well as textual content.**

and encrypts it using its private key. Both the message, which can be encrypted using the public key of the data provider, and the encrypted hash are sent to the data provider. The data provider uses the public key of the query initiator (or checks its white list) and decrypts the encrypted hash and also generates a hash of the message (after decrypting it). When both match, the data provider considers the verification to be successful. If the verification fails, then the query initiator is refused further processing. We illustrate this process in Fig. 9(a).

When the verification succeeds, the data provider executes the shipped queries on its local database. The query

```
Q4: How many patients have had past medical history of "anem ia"?


(: *********** Keyword query *********** :)
count (
 for $x in collection("CDN")/ClinicalDocument
 where $x//section[code/@code="11348-0"][code/@codeSystem="2.16.840.1.113883.6.1"]/text[
   contains(.,"anemia")]
 return $x/RecordTarget/PatientRole/ID
)
```

(a) XQuery query

```
/ClinicalDocument[RecordTarget/PatientRole/ID]//section[code/@code="11348-0"]
                 [code/@codeSystem="2.16.840.1.113883.6.1"]/text[contains(.,"anemia")]
```

(b) One maximal XPath expression for Q4

```
(: ************ Query template ************ :)
for $x in doc("...")/ClinicalDocument
where $x//section[code/@code="11348-0"][code/@codeSystem="2.16.840.1.113883.6.1"]/text[
   contains(.,"anemia")]
return <res> {$x/RecordTarget/PatientRole/ID} </res>
```

(c) Template of the query shipped to data providers with matching documents

```
count ( distinct-values (for $x in doc("results.xml")//ID return $x) )
```

(d) Counting and duplicate elimination performed locally

**Fig. 7 – Keyword query in CDN.**

is encrypted by the query initiator (using the public key of the data provider) and the results from the data provider are encrypted (using the public key of the query initiator). This prevents malicious attacks and eavesdropping. When the size of the results returned by a data provider is large, the data providers generates an AES key and uses it to encrypt the results. The AES key itself is encrypted using the public key of the query initiator. We illustrate the entire process in Fig. 9(b).

Note that never is an actual HL7 document exchanged or transferred through the network and it always resides with the owner. Moreover, through our hybrid shipping approach and the above security schemes, only an authorized query initiator and the data providers owning the relevant data for a query deal with unencrypted data.

### 3.5. *User interface*

The user interface of CDN is designed to allow a clinician or researcher to easily publish a HL7 v3 document and pose structured queries and keyword queries related to cancer diagnosis and treatment. For structured queries, a browse hierarchy is provided to simplify the input process for structured queries. A form for posing incidence related queries on colon cancer is shown in Fig. 10. (The user interface will be enhanced in the future to allow the entry of join queries.)

**Algorithm 4.** Generation of the FOR clause for a query

```
// Input: P is a list of predicates on patient attributes to be tested in the
// header section of HL7 CDA
// Output: The FOR clause for a query
proc Generate_FOR(list P)
1: if P is ∅then
2:     fp ← "/ClinicalDocument"
   else
3:     Let P = ¶p1, p2, ..., pi♢
4:     fp ←
       "/ClinicalDocument[./RecordTarget/PatientRole/p1 and
       ./RecordTarget/PatientRole/p2 and ···
       ./RecordTarget/PatientRole/pi]"
5: return concatenate("for $x in collection("CDN " )", fp)
endproc
```

### 3.6. *Generating XQuery queries*

A user provides inputs using form interfaces and specific XQuery query templates are associated with these interfaces. (We do not perform any natural language processing (NLP).) The generated XQuery queries have `for`, `where`, and `return` clauses. The `count()` function is finally applied to each query.

```
Q5: Find the number of patients who were given medications during hospital course that have
caused an allergy in one or more patients.
count ( distinct-values (
 for $e in collection("CDN")/ClinicalDocument,
     $f in collection("CDN")//section[code/@code="45675-6"][code/@codeSystem=
     "2.16.840.1.113883.6.1"]
 where $e/structuredBody/section[code/@code="8648-8"][code/@codeSystem="2.16.840.1.113883.6.1"]/
  manufacturedMaterial/@ApplicationNumber = $f/manufacturedMaterial/@ApplicationNumber
  return $e/RecordTarget/PatientRole/ID
))
```

(a) Join query in CDN

```
/ClinicalDocument[RecordTarget/PatientRole/ID]/structuredBody/section[code/@code="8648-8"]
         [code/@codeSystem="2.16.840.1.113883.6.1"]/manufacturedMaterial/@ApplicationNumber
//section[code/@code="45675-6"][code/@codeSystem="2.16.840.1.113883.6.1"]/
         manufacturedMaterial/@ApplicationNumber
```

(b) Maximal XPath expressions extracted from the query

```
(: ********** Query template A ********* :)
for $e in doc("...")/ClinicalDocument
where
  $e/structuredBody/section[code/@code="8648-8"][code/@codeSystem="2.16.840.1.113883.6.1"]/
  manufacturedMaterial[@ApplicationNumber]
return
<res> <arg1>{$e/structuredBody/section[code/@code="8648-8"]
       [code/@codeSystem="2.16.840.1.113883.6.1"]/manufacturedMaterial}</arg1>
 <arg2>{$e/RecordTarget/PatientRole/ID}</arg2>
</res>

(: ********** Query template B ********* :)
for $f in
   doc("...")//section[code/@code="45675-6"][code/@codeSystem="2.16.840.1.113883.6.1"]/
                      manufacturedMaterial[@ApplicationNumber]
  return  <res> <arg1>{$f}</arg1> </res>
```

(c) Templates of queries shipped to publishers with matching documents to enable local joins

```
count( distinct-values(
   for $e in doc("A.xml")//res, $f in doc("B.xml")//res
   where $e/arg1/manufacturedMaterial/@ApplicationNumber =
         $f/arg1/manufacturedMaterial/@ApplicationNumber
   return $e/arg2/ID
))
```



(d) Join operation, duplicate elimination, and aggregation performed locally

**Fig. 8 – Step-by-step evaluation of a location oblivious XQuery query with a join.**

Algorithm 4 shows the steps for generating the `for` clause of a query. An appropriate XPath expression is generated depending on the patient's attributes (e.g., gender, birthdate) to be matched in the HL7 CDA header section.

Algorithm 5 shows the steps for generating the `where` clause of a query where we are interested in patients matching a set of medical conditions. Textual content in certain HL7 CDA sections are also matched based on patient's conditions (e.g.,

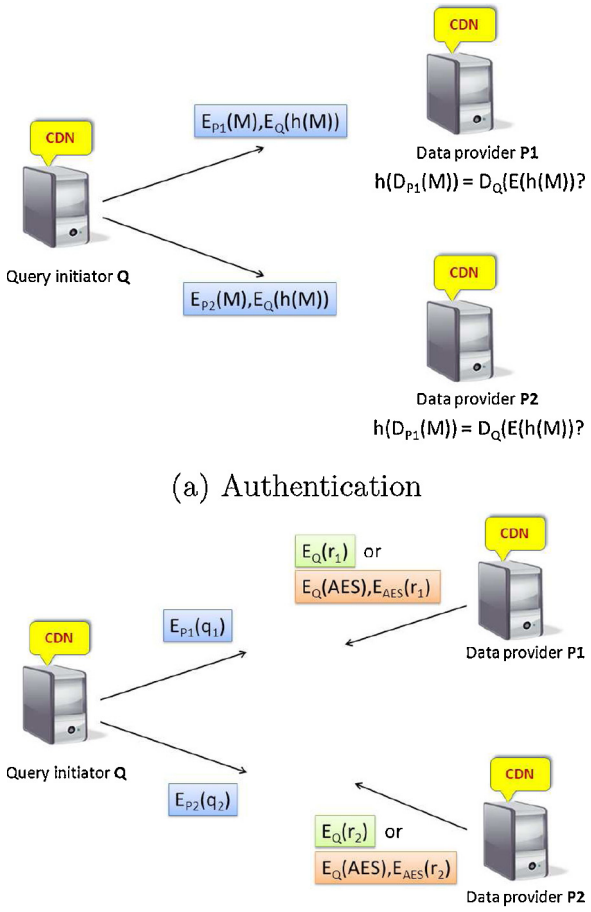



**Fig. 9 – Security schemes in CDN.**

check for "smoker" in history of present illness). Algorithm 6 shows the steps for generating the `where` clause of a query where we are interested in patients who had a procedure that caused a particular outcome, which may or may not be coded. Finally, Algorithm 7 shows the steps for generating the `where` clause of a query where keyword matching is performed. Different HL7 CDA sections can be queried. Currently, the `return` clause "`return $x/RecordTarget/PatientRole/ID`" is used

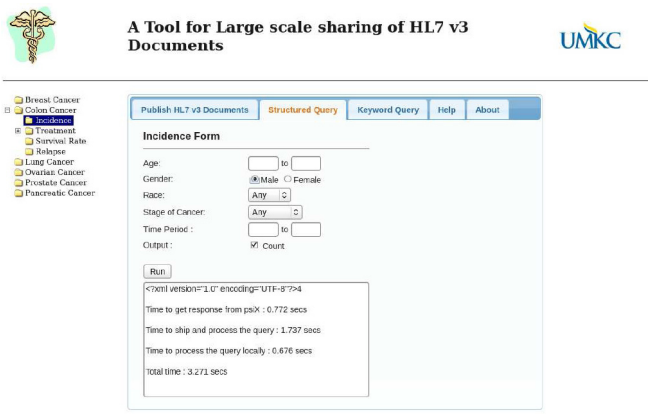



**Fig. 10 – A screenshot of the user interface in CDN.**

in each query. Note that the algorithms use concatenate(...) to concatenate a group of input strings.

**Algorithm 5.** Generation of the WHERE clause for a query based on medical conditions

// **Input**: C is a list of conditions; K is a list of keywords and CDA sections
// **Output**: The WHERE clause for a query

**proc** `Generate_WHERE_Condition`(list C, list K)

1: Let C = {($c_1$, $cs_1$), ($c_2$, $cs_2$),..., ($c_n$, $cs_n$)} be the list of conditions to match, where $c_i$ is the condition and $cs_i$ is the codesystem to use for $c_i$ (e.g., SNOMED CT, ICD-9, LOINC)

2: $wp \leftarrow$ "`((`$x//procedure/code[@code=`$code_{c_1}$`][@codeSystem=`$code_{cs_1}$`] or `$x//observation/code[@code=`$code_{c_1}$`][@codeSystem=`$code_{cs_1}$`]) and ... (`$x//procedure/code[@code=`$code_{c_n}$`][@codeSystem=`$code_{cs_n}$`] or `$x//observation/code[@code=`$code_{c_n}$`][@codeSystem=`$code_{cs_n}$`]))`"

3: Let K = {($k_1$, $S_1$, $cS_1$), ($k_2$, $S_2$, $cS_2$),..., ($k_m$, $S_m$, $cS_m$)} denote the list of keywords to match in different CDA sections such as patient's history of present illness, past medical history, etc, where $k_i$ is the keyword, $S_i$ is the section, and $cS_i$ is the codesystem to use for $S_i$. (e.g., Match "smoker" in history of present illness.)

4: $wp \leftarrow$ `concatenate` ($wp$, "`and`", "`($x//section[code/@code=`$code_{S_1}$`][code/@codeSystem=`$code_{cS_1}$`]/text[contains(.,`$k_1$`)] and ... $x//section[code/@code=`$code_{S_m}$`][code/@codeSystem=`$code_{cS_m}$`]/text[contains(.,`$k_m$`)])`")

5: **return** `concatenate` ("`where`", $wp$)

**endproc**

**Algorithm 6.** Generation of the WHERE clause for a query with a procedure causing an outcome

// **Input**: P is the procedure; E is the outcome
// **Output**: The WHERE clause for a query

**proc** `GenerateWHERE_Procedure_Outcome`(C, E)

1: Let P = (($p$, $cs_p$), ($S_1$, $cS_1$)), where $p$ is the procedure, $cs_p$ is the codesystem for $p$ (e.g., SNOMED CT, ICD-9, LOINC), $S_1$ is the CDA section to search in, and $cS_1$ is the codesystem to use for $S_1$. (e.g., Match for "chemotherapy" in history of present illness.)

2: Let E = (($e$, $cs_e$), ($S_2$, $cS_2$)), where $e$ is the outcome and $cs_e$ is the codesystem for $e$, $S_2$ is the CDA section to search in, and $cS_2$ is the codesystem to use for $S_2$. (e.g., Match for side-effect "alopecia" in physical examination.)

3: $wp \leftarrow$ "$\$x$//section[code/@code=$code_{S_1}$] [code/@codeSystem=$code_{cS_1}$]// procedure/code[@code=$code_p$] [@codeSystem=$code_{cs_p}$] and ($\$x$//section[code/@code=$code_{S_2}$] [code/@codeSystem=$code_{cS_2}$] // observation/code[@code=$code_e$] [@codeSystem=$code_{cs_e}$] or $\$x$//section[code/@code=$code_{S_2}$] [code/@codeSystem=$code_{cS_2}$]/ text[contains(.,$e$)])"

4: **return** concatenate("where", $wp$)

**endproc**

**Algorithm 7.** Generation of WHERE clause for keyword-based searching

// **Input**: K is the Boolean expression with keywords; S is the CDA section to search

// **Output**: The WHERE clause for a query

**proc** GenerateWHERE_Keyword(K, S)

1: Let K = ($k_1$ $op_1$ $k_2$ $op_2$... $k_i$) be the Boolean expression to query, where $k_i$ is a keyword and $op_i$ is a Boolean operator.

2: Let S = ($s$, $cs$), where $s$ is the CDA section and $cs$ is the corresponding codesystem. (e.g., Match for "anemia" and "migraine" in past medical history.)

3: $wp \leftarrow$ "//section[code/@code=$code_s$] [code/@codeSystem=$code_{cs}$]/text [contains(.,$k_1$) $op_1$ contains(.,$k_2$) $op_2$ ⋯ contains(., $k_i$)]"

4: **return** concatenate("where", $wp$)

**endproc**

### 3.7. *Joining and leaving* CDN

CDN can be setup to have any data provider join or leave the network of data providers at any time with little administration. Furthermore, if only authorized data providers should be permitted to join the network, then one data provider that usually acts as the bootstrap node, can maintain the public key of each authorized data provider and only allow a data provider to connect to the network after authentication using public key cryptography. In CDN, we expect low degree of churn unlike in Internet-scale P2P applications. We leave it up to the data provider to implement the desired authentication scheme for users in the institution that can interact with CDN using the GUI. For example, the data provider can rely on a single sign-on (SSO) approach used in many institutions.

## 4. Implementation and evaluation

We implemented CDN in Java using Eclipse and the open-source JSP and Servlet Container called Apache Tomcat (version 6). We used open-source XQuery processors, namely, SAXON [51] and BaseX [9], for storing and query processing of HL7 CDA documents in CDN. Available security libraries in Java were used for implementing the security schemes in CDN. The *psi*X codebase was written in C++and was implemented using the Chord DHT package [81]. To speed up query processing, CDN employed multithreading to ship queries in parallel to qualifying data providers.

CDN can be deployed in a conventional distributed environment consisting of a network of machines that communicate using TCP/IP protocols (e.g., a wide-area network). Because we did not have access to a large number of such machines, we used Amazon Elastic Compute Cloud (EC2) [79] as a testbed for our experiments. This way we could set up a large number of data providers with non-trivial network latencies between them, and each having a separate IP address. We could also study the impact of different processing capabilities of the data providers on the query processing time. Note that the performance evaluation reported in the conference version of this article [71] was conducted in a local cluster with five machines using a smaller number of HL7 CDA documents. Furthermore, multithreading was not implemented during query processing. In this article, we report a comprehensive evaluation of CDN using a large number of data providers and published documents.

Next, we report the performance evaluation of CDN in two settings, namely, a small-scale setting and a large-scale setting.

### 4.1. *Performance of* CDN *in a small-scale setting*

We report the performance evaluation of CDN using a real dataset on 20 data providers and a larger dataset of synthetic documents using 25, 50, and 75 data providers.

#### 4.1.1. *Using a real dataset*

(a) *Dataset of HL7 CDA documents*. We obtained deidentified discharge summaries from the NLP research datasets available from the i2b2 project [84]. From these discharge summaries, we created 335 HL7 CDA documents. These documents contained both coded content as well as textual content and had the following sections: History of Present Illness, Physical Examination, Past Medical History, Past Surgical History, Allergies, Hospital Course, Discharge Date, Discharge Diagnosis, and Discharge Disposition. The codes were drawn from LOINC, SNOMED CT, and FDA NDC (National Drug Code Directory). Clinical findings, observations, procedures, and manufactured materials in the discharge summaries were assigned appropriate codes. Human intervention was necessary due to the unstructured nature of textual content in the discharge summaries. For example, abbreviations were used in the discharge summaries such as B.C. for breast cancer and A. Fib. for atrial fibrillation. Stop words were removed in the textual content of the documents.

(b) *Setup of* CDN *and distribution of documents*. We set up CDN on 20 Amazon EC2 micro instances (or virtual machines), in the US East (Northern Virginia) region in the same availability zone. A micro instance provides one virtual core, up to 2 EC2 compute units (for short periodic bursts), 613 MB main memory, 8 GB of storage, and low I/O performance. Note that this instance type is the cheapest ($0.02/h) but provides the least amount of CPU resources among all the instance types available on Amazon EC2. It allows occasional spikes in the CPU usage based on availability. We ran one data provider on each

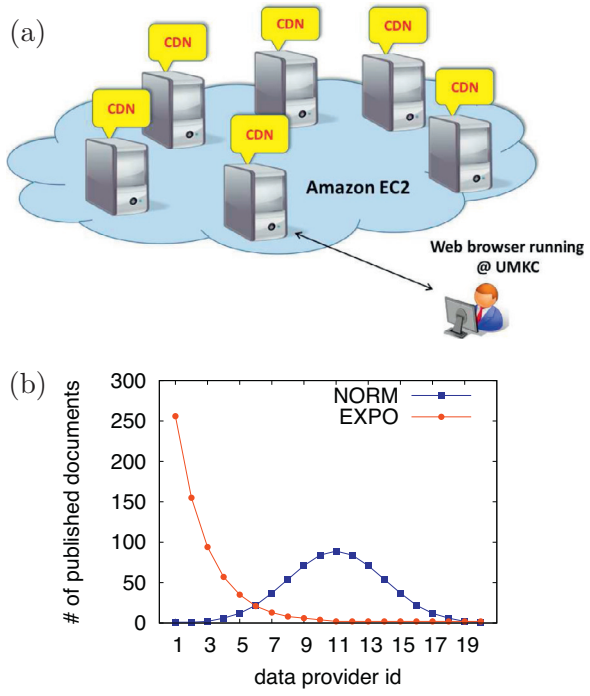



**Fig. 11 – (a) Setup of CDN on Amazon EC2 (b) Distribution of published documents.**

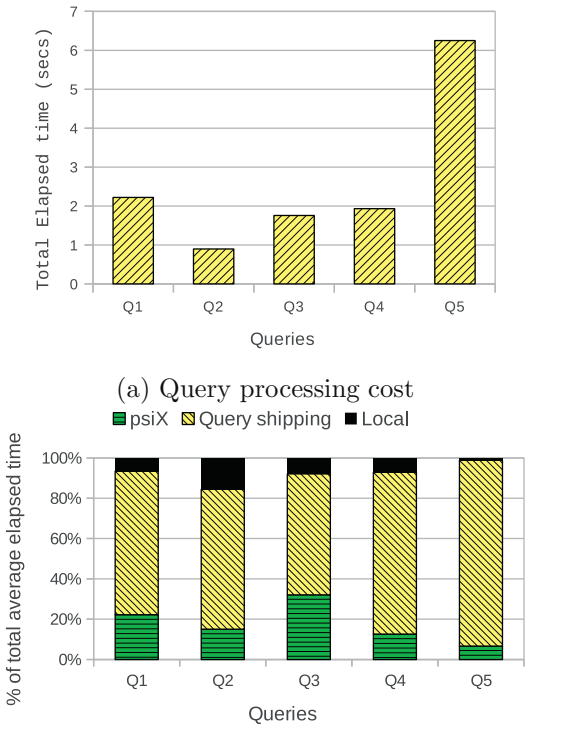



**Fig. 12 – Performance evaluation (NORM).**

micro instance and therefore, the data providers behaved as low-end compute servers. The instances communicated using TCP/IP protocols and ran 32 bit Linux. CDN was contacted via a web browser running at University of Missouri-Kansas City to either do publishing or querying. (See Fig. 11(a).)

We experimented with two distributions for the number of documents published by data providers as shown in Fig. 11(b). NORM was generated using a Gaussian distribution ($\mu = 0$ and $\sigma = 4$). EXPO was generated using a exponential distribution ($\lambda = 0.5$). Although our original dataset had only 335 documents, we made one more copy of each document but with a different patient id. So in all, we published 670 documents. Table 1 shows the characteristics of the documents. Because of the small number of published documents by each data provider, we used SAXON as the XQuery processor at each data provider. We will refer to the experiments conducted on these distributions as NORM and EXPO.

One of the data providers issued the location oblivious queries Q1–Q5 shown in Figs. 4–8.[2] We measured the total elapsed time for each query once it was chosen to run through the web browser running on-site. We report the average time over 5 runs.

(c) *Effect of Using* psiX. During query processing, CDN used *psi*X to locate relevant documents and data providers for a given query. Then it shipped subqueries to those qualifying data providers rather than contacting every data provider in the network. In Table 2, we report the number of maximal XPath expressions processed for a query and the total number of data providers identified by *psi*X, which were then contacted during query processing. For queries Q1–Q4, CDN contacted a small number of data providers during query processing. For Q5, the join query, 60% of the published documents were relevant to either one of the two maximal XPath expressions in Q5, and hence most of the data providers were contacted during query processing. It is interesting to note that for query Q4, the number of data providers contacted differed significantly for NORM and EXPO due to the difference in the distribution of published documents.

(d) *Performance for the* NORM *distribution.* Fig. 12(a) shows the average total elapsed time per query. Query Q5 is a join query and is more complex than the other queries. For Q5, CDN shipped subqueries to all the data providers as they contained relevant documents for the query. Q5 generated more results than the other queries. Hence CDN took more time to process Q5. Fig. 12(b) shows how much time was spent in the three phases of query processing, namely, (a) to locate relevant XML documents using *psi*X, (b) to ship queries to relevant data providers and receive the results using the security schemes in CDN, and (c) to perform local processing such as joins, duplicate elimination, and aggregation. We observed that the first phase where *psi*X was used, consumed under 35% of the total elapsed time and the final phase of local processing was less than 20% of the total time. The phase involving query shipping consumed most of the time as it required shipping queries to qualifying data providers and obtaining results in a secure manner (using the scheme described in Section 3.4).

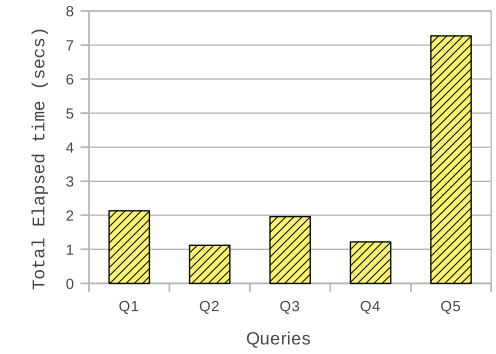
[2] This data provider published the least number of documents.

**Table 1 – Characteristics of the real documents.**

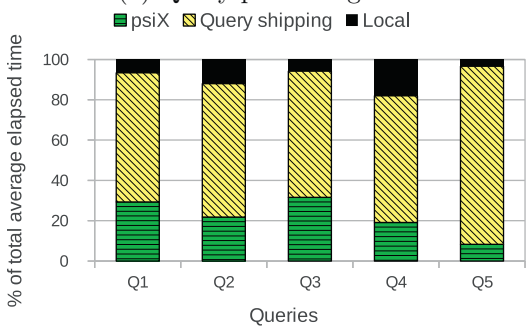
| Total # of documents | Total # of elements | Total # of attributes | Avg. size of documents |
|---|---|---|---|
| 670 | 38256 | 47112 | 6.88 KB |

**Table 2 – Effect of using *psi*X.**

| Query | # of maximal XPath expressions | Total # of data providers contacted (NORM) | Total # of data providers contacted (EXPO) |
|---|---|---|---|
| Q1 | 2 | 4 | 3 |
| Q2 | 2 | 2 | 3 |
| Q3 | 2 | 3 | 4 |
| Q4 | 1 | 8 | 3 |
| Q5 | 2 | 20 | 18 |




(a) Query processing cost




(b) Breakup of time spent during query processing

**Fig. 13 – Performance evaluation (EXPO).**

(e) *Performance for the* EXPO *Distribution*. Fig. 13(a) shows the average total elapsed time per query. As before, CDN took more time to process the join query Q5. Fig. 13(b) shows how much time was spent in the three phases of query processing. We observed that the first phase where *psi*X was used, consumed under 35% of the total elapsed time and the final phase of local processing was less than 20% of the total time. Note that we used a different set of 20 EC2 instances for EXPO. For EXPO, the time consumed by *psi*X was higher than that for NORM.

It is interesting to note that Q4 was processed faster for EXPO than NORM. This is because of a significant reduction in the number of data providers contacted for EXPO as compared with NORM. (See Table 2.)

#### 4.1.2. *Using synthetic datasets*

We evaluated the performance of CDN using synthetic datasets with up to 75 data providers, where each data provider had limited computing power and published a small number of documents. [3] Because we had limited number of documents in our real dataset, we generated synthetic HL7 CDA documents with structure identical to the documents in the real dataset but with randomly generated code values. Table 3 shows the characteristics of the documents. We tested CDN by varying the number of data providers; we set up 25, 50, and 75 data providers, and one data provider ran on each EC2 micro instance. Thus, in our setup, the data providers behaved as low-end compute servers.

We tested each location oblivious query listed in Fig. 14 on the synthetic dataset by issuing it from one data provider. For each query, we measured the average total elapsed time and the time spent during the three phases of query processing.

(a) *Evaluation results for 25 and 50 data providers*. We set up CDN with 25 data providers on 25 EC2 micro instances. All the instances ran in the US East region in the same availability zone. (The average round-trip time between instances was 0.6 ms.) A total of 2500 documents were published; each data provider published 100 synthetic HL7 v3 documents. Fig. 15(a) shows the average total elapsed time for queries Q6–Q8. Fig. 15(b) shows the time spent during the different phases of query processing. The time was dominated by the query shipping phase as many of the data providers were contacted during this phase and the data providers were run on EC2 micro instances.

Next, we set up CDN with 50 data providers on 50 EC2 micro instances. All the instances ran in the US East region in the same availability zone. (The average round-trip time between instances was 0.6 ms.) A total of 5000 documents were published; each data provider published 100 synthetic HL7 v3 documents. Fig. 16(a) shows the average total elapsed time for queries Q6, Q7, and Q8. Fig. 16(b) shows the time spent during different phases of query processing. As before, the time was dominated by the query shipping phase. Each query returned more results and contacted more data providers than in the previous setup with 25 data providers, because we published

[3] SAXON was used as the XQuery processor at each data provider.

**Table 3 – Characteristics of the synthetic documents.**

| Total # of documents | Total # of elements | Total # of attributes | Avg. size of documents |
|---|---|---|---|
| 2500 | 143,020 | 178,690 | 6.66 KB |
| 5000 | 285,670 | 352,180 | 6.82 KB |
| 7500 | 428,280 | 529,890 | 6.78 KB |

```
count (
 for $x in collection("CDN")/ClinicalDocument
 where $x/RecordTarget/PatientRole/Patient = "M" and
  ($x//observation/code[@codeSystem="2.16.840.1.113883.6.96"][@code = "58"] or
   $x//procedure/code[@codeSystem="2.16.840.1.113883.6.96"][@code = "13"])
 return $x/RecordTarget/PatientRole/ID
)
```

(a) Query Q6

```
count (
 for $x in collection("CDN")/ClinicalDocument
 where $x//procedure/code[@code="52"][@codeSystem="2.16.840.1.113883.6.96"] and
  ($x//observation/code[@code="24"][@codeSystem="2.16.840.1.113883.6.96"] or
   $x//section[code/@code="11493_4"][code/@codeSystem="2.16.840.1.113883.6.1"]/
    text[contains(., "myocardial")])
 return $x/RecordTarget/PatientRole/ID
)
```

(b) Query Q7

```
count (
 for $x in
   collection("CDN")/ClinicalDocument[RecordTarget/PatientRole/Patient = "F"]
 where ($x//procedure/code[@code="23"][@codeSystem="2.16.840.1.113883.6.96"] or
   $x//observation/code[@code="41"][@codeSystem="2.16.840.1.113883.6.96"]) and
   $x//section[code/@code="10184-2"][code/@codeSystem="2.16.840.1.113883.6.1"]/
    text[contains(., "smoker")]
 return $x/RecordTarget/PatientRole/ID
)
```

(c) Query Q8

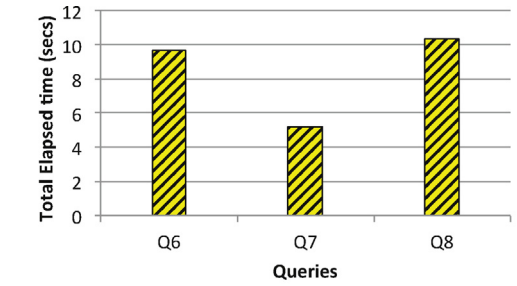
**Fig. 14 – Queries for the synthetic dataset.**

double the number of documents. Therefore, the total time increased.

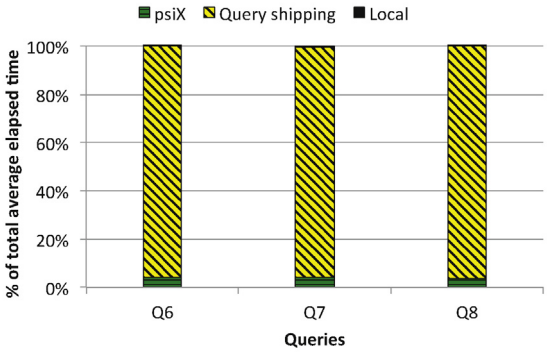
The average time spent by *psi*X to process the maximal XPath expressions in a query ranged from 0.17 to 0.44 s. Overall, *psi*X was efficient in locating relevant documents in a network of 25 and 50 data providers.

(b) *Evaluation results for 75 data providers.* Next, we conducted a study by running CDN with 75 data providers on 75 EC2 micro instances. We ran 25 instances in one availability zone and 50 instances in another availability zone; both the availability zones belonged to the US East region. (The average round-trip time across these two availability zones was 2.0 ms.) A total of 7500 documents were published; each data provider published 100 synthetic HL7 v3 documents.

In this experiment, we wanted to understand how the query processing cost of CDN was affected when the number of data providers involved during query processing (i.e., to process shipped queries) changed. Fig. 17 shows the average total elapsed time for queries Q6–Q8. The number of data providers changed from 19 to 4. For each of these queries, the query processing time decreased as the number of data providers involved decreased. Figs. 18(a)–(c) shows the time spent

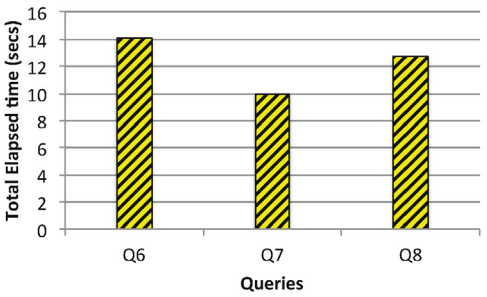



(a) Query processing cost

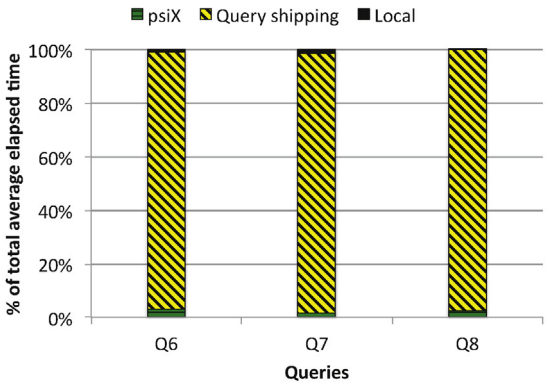



(b) Breakup of time spent during query processing

**Fig. 15 – Performance of CDN with 25 data providers and 2500 published documents.**




(a) Query processing cost




(b) Breakup of time spent during query processing

**Fig. 16 – Performance of CDN with 50 data providers and 5000 published documents.**

during the different phases of query processing for queries Q6, Q7, and Q8, respectively. As before, the time was dominated by the query shipping phase.

The average time spent by *psi*X to process the maximal XPath expressions in a query ranged from 0.25 to 0.39 s. Overall, *psi*X was efficient in locating relevant documents in a network of 75 data providers.

### 4.2. *Performance of CDN in a large-scale setting*

We evaluated the performance of CDN in a large-scale setting with hundreds of data providers and millions of documents. We selected the US East region and ran 200 EC2 instances in one data center (or availability zone) and the remaining 200 EC2 instances in another data center. On each instance, we ran one data provider. In all, CDN had a network of 400 data providers.

As in the earlier experiments, we generated synthetic HL7 documents with structure identical to those in the real dataset but with randomly generated code values. Table 4 shows the characteristics of the documents. Together, the data providers hosted a total of 4 million synthetic HL7 CDA documents. (Each data provider had 10,000 documents). Because of the large number of documents, we used BaseX [9] as the underlying XQuery processor on each data provider. The data providers built structural indexes on their XML documents.

We tested the location oblivious queries listed in Fig. 14 by issuing them from one data provider. The query initiator was in one data center and the data providers involved during

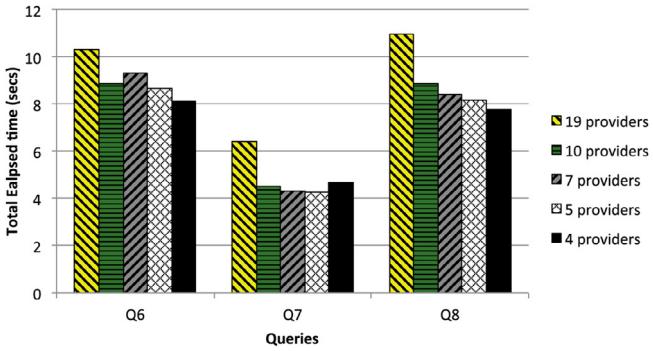



**Fig. 17 – Performance of CDN with 75 data providers and 7500 published documents.**

query shipping were in the other data center. This introduced non-trivial network latencies during query processing with an average round-trip time of 2 ms between the two data centers. We also varied the selectivities of the queries as well as the processing capabilities of the data providers.

We varied the selectivity of each query by varying the percentage of data providers in the network that were involved during the query shipping phase. (These data providers processed shipped queries.) We denote this percentage by $\alpha$. For each query, we measured the total elapsed time (averaged over 3 runs) for $\alpha = 1\%$, $\alpha = 10\%$, and $\alpha = 20\%$ of the total number

**Table 4 – Characteristics of the synthetic documents.**

| Total # of documents | Total # of elements | Total # of attributes | Avg. size of documents |
|---|---|---|---|
| 4,000,000 | 229,536,000 | 282,672,000 | 6.83 KB |

**Table 5 – The final counts obtained after the execution of queries with 400 data providers in the network.**

| Query | Final count (i.e., # of matches) | | |
|---|---|---|---|
| | $\alpha = 1\%$ (4 data providers) | $\alpha = 10\%$ (40 data providers) | $\alpha = 20\%$ (80 data providers) |
| Q6 | 1252 | 12,520 | 25,040 |
| Q7 | 28 | 280 | 560 |
| Q8 | 24 | 240 | 480 |

**Table 6 – Total size of the intermediate results for queries with 400 data providers in the network.**

| Query | Total size of intermediate results | | |
|---|---|---|---|
| | $\alpha = 1\%$ (4 data providers) | $\alpha = 10\%$ (40 data providers) | $\alpha = 20\%$ (80 data providers) |
| Q6 | 39,984 bytes | 399,732 bytes | 799,452 bytes |
| Q7 | 912 bytes | 9012 bytes | 18,012 bytes |
| Q8 | 792 bytes | 7812 bytes | 15,612 bytes |

of data providers in the network. (The total number of data providers was 400.) As the number of data providers involved during query shipping increased, more matches were found, and the total size of the intermediate results also increased. The final count value (i.e., the number of matching patients) obtained after the execution of each query for different values of $\alpha$ is shown in Table 5. The count value increased linearly with increase in $\alpha$. For each query, the size of the intermediate results for different values of $\alpha$ is shown in Table 6.

We varied the processing capabilities of the data providers by running them on 3 different EC2 instance types, namely, *small*, *medium*, and *high-CPU medium* instance types. (These instance types are more powerful than the *micro* instance type used in Section 4.1.) A small instance provides one virtual core with 1 EC2 compute unit, 1.7 GiB main memory, 160 GB of storage, and moderate I/O performance. A medium instance provides one virtual core with 2 EC2 compute units, 3.75 GiB main memory, 410 GB of storage, and moderate I/O performance. Finally, a high-CPU medium instance provides 5 EC2 compute units (2 virtual cores), 1.7 GiB main memory, 350 GB of storage, and moderate I/O performance.

Next, we report the total elapsed time (averaged over 3 runs) to process the queries. Fig. 19(a) shows the total elapsed time for $\alpha = 1\%$ on small, medium, and high-CPU medium instance types. As expected, when data providers ran on more powerful instances, the query processing time reduced and the best performance was obtained with high-CPU medium instances. For example, Q6 ran 3.6 times faster on high-CPU medium instances than on small instances. Fig. 19(b) shows the total elapsed time for $\alpha = 10\%$ on small, medium, and high-CPU medium instance types. As before, the query processing time reduced when more powerful instances were used to run the data providers. Finally, Fig. 19(c) shows the total elapsed time for $\alpha = 20\%$ on small, medium, and high-CPU medium instance types. Similar trends were observed as before.

From the results in Fig. 19, we observed that with increasing values of $\alpha$, the total elapsed time increased for all the queries. For example, with medium EC2 instances, Q6 was processed in 4.47 s, 15.72 s, and 87.74 s, for $\alpha = 1\%$, $\alpha = 10\%$, and $\alpha = 20\%$, respectively. This was expected because when $\alpha$ increased, more data providers were involved during query shipping, resulting in more matches. Because Q6 had lower selectivity than Q7 and Q8, in most cases, CDN required more time to process it for the same value of $\alpha$ and instance type. For example, with high-CPU medium instances, Q6–Q8 were processed in 70.28 s, 6.29 s, and 11.32 s, respectively, for $\alpha = 20\%$.

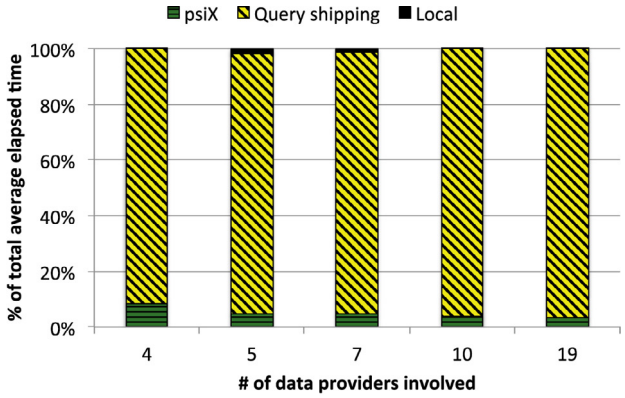
## 5. Discussion

In this section, we present the salient features of CDN and the significance of our experimental setup and performance evaluation results. We show how CDN can be extended to handle different standard terminologies in HL7 v3 documents using available mapping methods; we also point out that handling the variation in coding within a same terminology (e.g., SNOMED CT) is a serious challenge and requires further research. We also discuss our design choices for security and privacy in the context of recent work on privacy-preserving data publishing. Finally, we present our plans for future work.

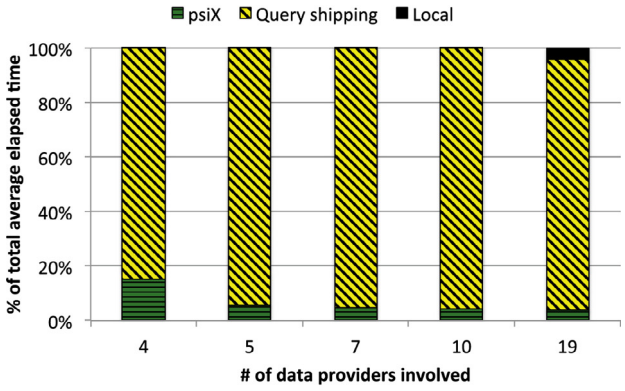
### 5.1. *Salient features*

The salient features of CDN are summarized below.

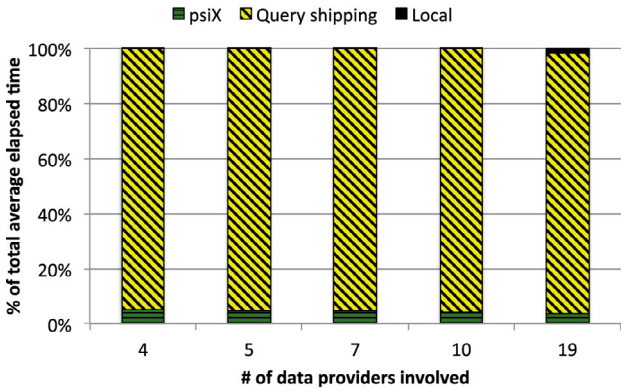
- The design of CDN is based the synergistic combination of the peer-to-peer (P2P) technology and the widely adopted XML standard and the XQuery language. The striking feature of CDN is the notion of a *location oblivious query*, wherein the locations of relevant data in the network (for a query) are discovered during query processing. A user simply issues a single location oblivious query (that is mapped to an XQuery query) across multiple data sources in the network to perform aggregations and joins.
- Though CDN employs a P2P model, it still provides the benefits of a federated database model such as ownership of data and ability to implement local access control policies. The HL7 v3 clinical documents are always stored with the owner




(a) Query Q6

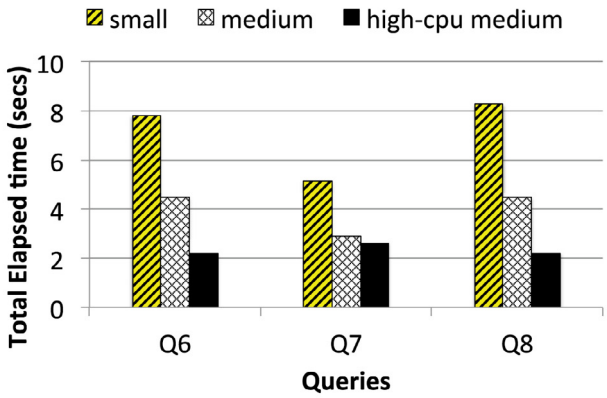



(b) Query Q7

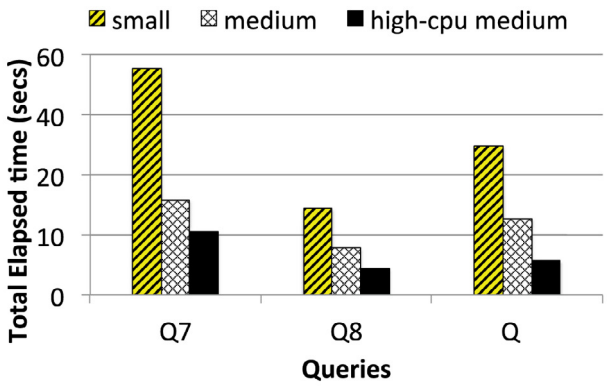



(c) Query Q8

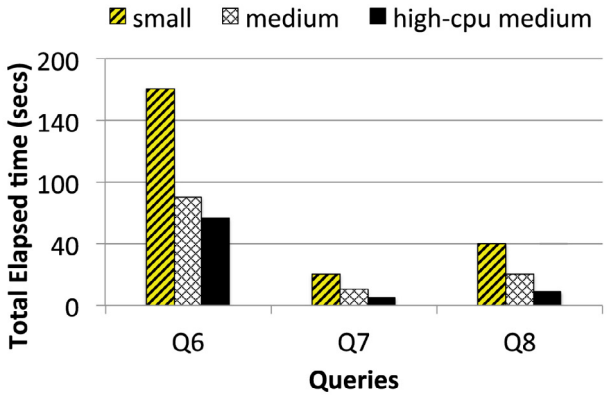
**Fig. 18 – Breakup of time spent during query processing.**




(a) $\alpha = 1\%$ (4 data providers were shipped queries)




(b) $\alpha = 10\%$ (40 data providers were shipped queries)




(c) $\alpha = 20\%$ (80 data providers were shipped queries)

**Fig. 19 – Total elapsed time on different EC2 instance types with 400 data providers and 4 million HL7 CDA synthetic documents.**

and are never exchanged or transferred across the network. While caBIG is designed to enable sharing of a wide range of biomedical and clinical data, CDN focuses only on sharing and querying HL7 v3 clinical documents.

- To process a location oblivious query, CDN has a novel location service called *psi*X for quickly identifying the locations of data relevant to the query [70]. This location service uses a novel distributed XML indexing technique that allows processing of XPath queries in a P2P environment.
- CDN employs a hybrid shipping approach [55] to process an XQuery query. Parts of the query are shipped across the network and parts of the query are processed by the data provider where the query was issued. Not only is this better than pure data shipping or pure query shipping w.r.t

resource utilization and performance, but also for enabling high levels of security and privacy during query processing.
- CDN employs standard cryptographic techniques (i.e., RSA and AES) to provide security to data providers, prevent unauthorized data providers, authenticate members in the network during query processing, and provide confidentiality and integrity to message exchanged during query processing.
- CDN is built using open-source software tools and has been tested in a distributed environment using Amazon EC2.

### 5.2. *Performance evaluation on Amazon EC2*

Next, we discuss the significance of our performance evaluation and experimental setup. We evaluated CDN in a distributed setting using Amazon EC2 as a testbed. Because we did not have access to a large number of machines in a wide-area network, we selected Amazon EC2. We set up a large number of data providers (up to 400), each with a separate IP address. We also studied the impact of different processing capabilities of the data providers on the query processing cost. We tested CDN with a real dataset of discharge summaries available from the i2b2 project [65] and a synthetic dataset containing 4 million documents. Not only did we test the performance of CDN within a data center but also across two different data centers with an average round-trip time of 2 ms between them. (This latency was non-trivial and much higher than that in a local area network. The average round-trip time between two hosts in our local Gigabit cluster was 0.11 ms. The average round-trip time between a host in this cluster and an EC2 instance in the US East region was 43 ms.) So our experimental setup is more realistic than a local area network.

CDN showed good performance on real and synthetic datasets. The query processing time was dominated by the activities in the query shipping phase. The following parameters affect the time taken by this phase: (a) the number of data providers that are involved during query shipping, (b) the network latencies to contact the data providers, (c) the size of partial results returned by the data providers, and (d) their processing power and system load during query processing; the latter two will affect the speed of encryption and decryption during query processing.

### 5.3. *Different coding standards*

It is possible for CDN to search HL7 v3 documents coded with different coding standards. This is because XQuery allows Boolean operators to combine different XPath expression. For example, one can search for clinical finding "Diverticula of colon," which may be coded using SNOMED CT or ICD-9-CM or ICD-10-CM using the XQuery query shown in Fig. 20. Fortunately, there are methods today to map codes across these standard clinical terminologies [3,2].

Within a standard there are variations how the codes are used by coding experts [7,6]. For example, there is more than one way to represent concepts in SNOMED CT through pre-coordination or post-coordination of codes. CDN does an exact match on the codes and therefore, cannot determine the equivalency between pre-coordinated and post-coordinated codes in HL7 v3 documents. One possible solution to overcome this issue is to use existing methods [27,50]: The pre- and post-coordinated concepts in HL7 v3 documents could be represented in a normal/canonical form before being published by a data provider. Pre-coordinated codes can still be retained in the documents as they facilitate faster searching when a query contains them. At query time, CDN could generate a query in the canonical form as well as have pre-coordinated codes. The location service *psi*X could search for documents that either match the maximal XPath expressions with pre-coordinated codes or in the canonical form. Similarly, the shipped queries will contain XQuery queries with pre-coordinated codes as well as in the canonical form. While processing the canonical form, the rules for testing equivalence can be applied [50]. Further research is necessary to fully solve this problem.

### 5.4. *Privacy-preserving data publishing*

Privacy-preserving data publishing has received much attention in recent years due to the need for sharing sensitive data in many applications. The goal is to preserve the privacy of individuals in a published dataset while maintaining its utility for tasks such as mining and analysis. Many privacy-preserving data publishing approaches have been proposed in recent years for relational data [82,10,61,36,57,30,11,35,42,87], set-valued data [40,88,43,17,83,18], and for both [64]. A few approaches have been developed for preserving the privacy of distributed XML content [12,21,39].

One may wonder if the aforementioned approaches can be used in CDN. We provide our thoughts by considering real-world, data sharing tools deployed in US hospitals such as SHRINE [86,85] and FURTHeR [60,59]. These systems employ a federated architecture, where the data providers control their data and only aggregate counts on matches for a query are returned by a data provider. This is done to abide by the state and federal laws [85]. CDN draws inspiration from these systems, wherein the data providers own their HL7 CDA documents and never transmit them through the network. Only partial results are transmitted, and the integrity and confidentiality of messages are preserved during transmission. Unlike healthcare datasets that are released for secondary use, the data owned by data providers in CDN can be queried only by authorized participants. Also note that CDN is not a publish-subscribe system.

The following scenarios can arise in a system designed for sharing clinical and biomedical data: (a) sharing of deidentified data, (b) sharing of limited datasets, or (c) sharing of identifiable data [1]. In the United States, according to the HIPAA Privacy Rule [44], protected health information (PHI) can be deidentified either by an expert statistician or by removing 18 identifiers. Data providers can share deidentified data without violating HIPAA. HIPAA also allows data providers to share limited datasets that contain certain identifiers in PHI (e.g., age, codes) provided they sign a data use agreement. Data providers can share identifiable data (e.g., in HIEs) based on a trust agreement [78].

If CDN is used to share and query deidentified data, the aforementioned privacy-preserving data publishing approaches can be utilized to deidentify CDA documents. These deidentified documents can then be processed by CDN. For sharing limited datasets and identifiable data, however,

```
count(
 for $x in collection("CDN")/ClinicalDocument
 where
 $x//observation/code[@codeSystem="2.16.840.1.113883.6.96"][@code = "63532004"] or
 $x//observation/code[@codeSystem="2.16.840.1.113883.6.103"][@code =  "562.10"] or
 $x//observation/code[@codeSystem="2.16.840.1.113883.6.90"][@code =  "K57.30"]
   return $x/RecordTarget/PatientRole/ID
)
```

**Fig. 20 – An XQuery query in CDN that searches using different coding standards.**

a privacy-preserving data publishing approach may not be needed. Note that data providers in CDN are free to implement local access control policies [23,31]. Yet, there are some additional issues to handle: Unauthorized entities should not be allowed to participate in the data sharing network. The integrity and confidentiality of messages exchanged between data providers must be preserved. For these reasons, we have adopted well-known cryptographic techniques in CDN. In fact, digital certificates are being used by caBIG services for authentication and secure transmission [16].

### 5.5. *Future plan*

In the future, we plan to use the popular i2b2 web front-end for CDN, similar to other efforts [59]. We plan to extend the current implementation of CDN to employ digital certificates. We also plan to study the quality of results returned by CDN. To encourage the adoption of CDN by the community, we believe a cloud-based deployment is compelling, because no software changes are needed on-premise at a hospital or a clinic, and users can simply run CDN using any web browser with appropriate security mechanisms. We have already established the feasibility of deployment of CDN and its performance implications in a cloud computing environment. We plan to study how CDN can leverage the elastic properties of a cloud infrastructure. In the future, a virtual machine image with CDN pre-installed can be provided in a cloud (e.g, Amazon EC2). A hospital or clinic could launch one or more instances, which would behave as CDN data providers, and connect to other data providers launched by other hospitals and clinics. Note that the instances will be under full control of the owner who launched them. A few companies (e.g., MedCommons, TC3 Health) have already developed HIPAA compliant applications using Amazon cloud services. We plan to release CDN as a public image on Amazon EC2 under the Open Health Tools Initiative (http://openhealthtools.org).

## 6. Conclusions

We presented a new software tool called CDN for sharing and querying of HL7 v3 clinical documents. CDN is based on the synergistic combination of the P2P model of computing and XML and XQuery technologies. The key feature of CDN is the notion of location oblivious queries. CDN provides complete ownership of data to a data provider similar to a federated database model. For HIPAA compliance, CDN uses standard cryptographic techniques for providing security to data providers and protecting the privacy of patient records. CDN has a simple user interface for posing structured as well as keyword queries on both coded as well as textual content in HL7 documents. We have evaluated CDN in a distributed environment using Amazon EC2 as a testbed. We tested CDN using real and synthetic datasets. Overall, CDN showed good performance in a setup with large number of data providers and documents.

## Acknowledgments

We thank the anonymous reviewers for their valuable suggestions. Deidentified clinical records used in this research were provided by the i2b2 National Center for Biomedical Computing funded by U54LM008748 and were originally prepared for the “Shared Tasks for Challenges in NLP for Clinical Data” organized by Dr. Ozlem Uzuner, i2b2 and SUNY.

This work was supported in part by the National Science Foundation Grant No. 1115871, University of Missouri Research Board, and Amazon Web Services (AWS) in Education Research Grant.

## REFERENCES

[1] caBIG Data Sharing Information, 2011. https://cabig-stage.nci.nih.gov/community/working_groups/DSIC_SLWG/data_sharing_policy

[2] ICD-9-CM to SNOMED CT Map, 2012. http://www.nlm.nih.gov/research/umls/mapping_projects/icd9cm_to_snomedct.html

[3] SNOMED CT to ICD-10-CM, 2012. http://www.nlm.nih.gov/research/umls/mapping_projects/snomedct_to_icd10cm.html

[4] S. Abiteboul, I. Manolescu, N. Polyzotis, N. Preda, C. Sun, XML Processing in DHT Networks., in: Proc. of the 24th IEEE ICDE, Cancun, 2008 Apr.

[5] N. Anderson, A. Abend, A. Mandel, E. Geraghty, D. Gabriel, R. Wynden, M. Kamerick, K. Anderson, J. Rainwater, P. Tarczy-Hornoch, Implementation of a deidentified federated data network for population-based cohort discovery, Journal of the American Medical Informatics Association (2011).

[6] J.E. Andrews, T.B. Patrick, R.L. Richesson, H. Brown, J.P. Krischer, Comparing heterogeneous SNOMED CT coding of clinical research concepts by examining normalized

expressions, Journal of Biomedical Informatics 41 (6) (2008) 1062–1069.
[7] J.E. Andrews, R.L. Richesson, J. Krischer, Variation of SNOMED CT coding of clinical research concepts among coding experts, Journal of the American Medical Informatics Association 14 (4) (2007) 497–506.
[8] E. Bach, J. Shallit, Algorithmic Number Theory (vol. 1: Efficient Algorithms), MIT Press, Cambridge, MA, 1996.
[9] BaseX. A Light-weight, High-performance and Scalable XML Database Engine and XPath/XQuery Processor, 2011. Available from: http://basex.org/home/
[10] R.J. Bayardo, R. Agrawal, Data privacy through optimal k-anonymization, in: Proceedings of the 21st International Conference on Data Engineering, 2005, pp. 217–228.
[11] R. Bhaskar, S. Laxman, A. Smith, A. Thakurta, Discovering frequent patterns in sensitive data, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 2010, pp. 503–512.
[12] A. Bonifati, A. Cuzzocrea, XlPPX: a lightweight framework for privacy preserving P2P XML databases in very large publish-subscribe systems, in: Proceedings of the 8th International Conference on E-commerce and Web Technologies, EC-Web'07, Regensburg, Germany, 2007, pp. 21–34.
[13] caBIG The caBIG Pilot Phase Report Executive Summary, 2007. https://cabig.nci.nih.gov/overview/pilotreport_ExSum.pdf
[14] caBIG caBIG Annual Report 2009, 2009. http://cabig.cancer.gov/resources/reports/2009ar/
[15] caBIG. caAdapter, 2010. https://cabig.nci.nih.gov/tools/caAdapter/
[16] caBIG. GAARDS, 2010. http://cagrid.org/display/gaards/Home
[17] J. Cao, P. Karras, C. Raïssi, K.-L. Tan, p-Uncertainty: inference-proof transaction anonymization, in: Proceedings of the VLDB Endowment, vol. 3(1–2), 2010 Sept., pp. 1033–1044.
[18] R. Chen, N. Mohammed, B.C.M. Fung, B.C. Desai, L. Xiong, Publishing set-valued data via differential privacy, PVLDB 4 (11) (2011) 1087–1098.
[19] I.O.M. Committee on Quality of Health Care in America, Crossing the Quality Chasm: A New Health System for the 21st Century, The National Academies Press, Washington, DC, 2001.
[20] E. Curtmola, A. Deutsch, D. Logothetis, K.K. Ramakrishnan, D. Srivastava, K. Yocum, XTreeNet: democratic community search, in: Proc. of the 34st VLDB Conference, Auckland, 2008, pp. 1448–1451.
[21] A. Cuzzocrea, E. Bertino, Privacy preserving OLAP over distributed XML data: a theoretically-sound secure-multiparty-computation approach, Journal of Computer and System Sciences 77 (6) (2011) 965–987.
[22] J. Daemen, V. Rijmen, The Design of Rijndael: AES – The Advanced Encryption Standard, Springer-Verlag, Berlin Heidelberg, Germany, 2002.
[23] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, P. Samarati, A fine-grained access control system for XML documents, ACM Transactions on Information and System Security 5 (2) (2002 May) 169–202.
[24] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, W. Vogels, Dynamo: Amazon's highly available key-value store, in: Proc. of 21st Symposium on Operating Systems Principles, Stevenson, WA, 2007, pp. 205–220.
[25] L.T. Detwiler, D. Suciu, J.D. Franklin, E.B. Moore, A.V. Poliakov, E.S. Lee, D.P. Corina, G.A. Ojemann, J.F. Brinkley, Distributed XQuery-based integration and visualization of multimodality brain mapping data, Frontiers in Neuroinformatics 3 (0) (2009).
[26] W. Diffie, M.E. Hellman, Multiuser cryptographic techniques, in: Proceedings of the June 7–10, 1976, National Computer Conference and Exposition, New York, NY, USA, ACM, 1976, pp. 109–112.
[27] R. Dolin, K. Spackman, D. Markwell, Selective retrieval of pre- and post-coordinated SNOMED concepts, in: Proceedings of the AMIA Symposium, American Medical Informatics Association, 2002, pp. 210–214.
[28] R.H. Dolin, L. Alschuler, S. Boyer, C. Beebe, F.M. Behlen, P.V. Biron, A. Shabo Shvo, HL7 clinical document architecture, release 2, Journal of the American Medical Informatics Association 13 (1) (2006) 30–39.
[29] DXQP. DXQP – Distributed XQuery Processor, 2010. http://sig.biostr.washington.edu/projects/dxqp/
[30] K.E. Emam, F.K. Dankar, R. Issa, E. Jonker, D. Amyot, E. Cogo, J.-P. Corriveau, M. Walker, S. Chowdhury, R. Vaillancourt, T. Roffey, J. Bottomley, A globally optimal k-anonymity method for the de-identification of health data, Journal of American Medical Informatics Association 16 (5) (2009) 670–682.
[31] W. Fan, C.-Y. Chan, M. Garofalakis, Secure XML querying with security views, in: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, SIGMOD'04, Paris, France, 2004, pp. 587–598.
[32] D. Fenstermacher, C. Street, T. McSherry, V. Nayak, C. Overby, M. Feldman, The cancer biomedical informatics grid (caBIG), in: Proceedings of IEEE Engineering in Medicine and Biology Society, Shanghai, China, 2005, pp. 743–746.
[33] M. Fernandez, T. Jim, K. Morton, N. Onose, J. Simeon, DXQ: a distributed XQuery scripting language, in: 4th International Workshop on XQuery Implementation Experience and Perspectives, 2007.
[34] M.F. Fernàndez, T. Jim, K. Morton, N. Onose, J. Siméon, Highly distributed XQuery with DXQ, in: Proc. of SIGMOD 2007, 2007, pp. 1159–1161.
[35] A. Friedman, A. Schuster, Data mining with differential privacy, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 2010, pp. 493–502.
[36] B. Fung, K. Wang, P. Yu, Anonymizing classification data for privacy preservation, IEEE Transactions on Knowledge and Data Engineering 19 (5) (2007) 711–725.
[37] L. Galanis, Y. Wang, S.R. Jeffery, D.J. DeWitt, Locating data sources in large distributed systems, in: Proc. of the 29th VLDB Conference, Berlin, 2003.
[38] Galax. Galax: An Implementation of XQuery, 2010. http://galax.sourceforge.net/
[39] J. Gao, T. Wang, D. Yang, XFlat: query-friendly encrypted XML view publishing, Information Sciences 178 (3) (2008) 774–787.
[40] G. Ghinita, Y. Tao, P. Kalnis, On the anonymization of sparse high-dimensional data, in: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, ICDE '08, 2008, pp. 715–724.
[41] D. Goldstein, P. Groen, U.S. National Health Information Network (NHIN) and Open Source Health Information Exchange (HIE) Solutions, 2006. http://www.hoise.com/vmw/07/articles/vmw/LV-VM-01-07-29.html
[42] M. Hay, V. Rastogi, G. Miklau, D. Suciu, Boosting the accuracy of differentially private histograms through consistency., in: Proceedings of the VLDB Endowment, vol. 3(1–2), 2010 Sept., pp. 1021–1032.
[43] Y. He, J.F. Naughton, Anonymization of set-valued data via top-down, Local Generalization. Proceedings of the VLDB Endowment 2 (August (1)) (2009) 934–945.

[44] HHS Summary of the HIPAA Privacy Rule. http://www.hhs.gov/ocr/privacy/hipaa/understanding/summary/privacysummary.pdf, 2003.

[45] HIMSS HIMSS Health Information Exchange, 2006. http://www.himss.org/asp/topics_hie.asp

[46] HIMSS Defining Health Information Exchange, 2009. http://www.himss.org/content/files/2009DefiningHIE.pdf

[47] HIMSS Overview of Health Information Exchange (HIE), 2009. http://www.himss.org/content/files/RHIO/RHIO_HIE_GeneralPresentation.pdf

[48] J. Hui, S.E. Knoop, P. Schwarz, HIWAS: enabling technology for analysis of clinical data in XML documents, PVLDB 4 (12) (2011) 1260–1271.

[49] J. Hui, P. Schwarz, S. Knoop, Analyzing clinical data in XML: bridging the gaps, in: Proc. of 2nd ACM International Health Informatics Symposium, Miami, FL, 2011.

[50] IHTSDO SNOMED Clinical Terms Transforming Expressions to Normal Forms – Draft for External Comment. http://www.ihtsdo.org, 2007.

[51] M. Kay, SAXON: The XSLT and XQuery Processor, Available from: http://saxon.sourceforge.net/, 2011.

[52] D.B. Keator, D. Wei, S. Gadde, H.J. Bockholt, J.S. Grethe, D. Marcus, N. Aucoin, I.B. Ozyurt, Derived data storage and exchange workflow for large-scale neuroimaging analyses on the BIRN grid, Frontiers in Neuroinformatics 3 (0) (2009).

[53] K. Kim, Clinical Data Standards in Health Care: Five Case Studies, 2005. http://www.chcf.org/publications/2005/07/clinical-data-standards-in-health-care-five-case-studies

[54] G. Koloniari, E. Pitoura, Peer-to-peer management of XML data: issues and research challenges, SIGMOD Record 34 (June (2)) (2005) 6–17.

[55] D. Kossmann, The state of the art in distributed query processing, ACM Computing Surveys 32 (4) (2000) 422–469.

[56] A. Lakshman, P. Malik, Cassandra: a structured storage system on a P2P network, in: Proc. of the 2008 ACM-SIGMOD Conference, Vancouver, Canada, 2008.

[57] K. LeFevre, D.J. DeWitt, R. Ramakrishnan, Workload-aware anonymization techniques for large-scale datasets, ACM Transactions on Database Systems 33 (September (3)) (2008), 17:1–17:47.

[58] LinkedIn. Project Voldemort – A Distributed Database, 2009. http://project-voldemort.com/

[59] O. Livne, N. Schultz, S. Narus, Federated querying architecture with clinical and translational health IT application, Journal of Medical Systems 35 (2011) 1211–1224.

[60] O.E. Livne, N.D. Schultz, S.P. Narus, Federated querying architecture for clinical and translational health IT, in: Proc. of the 1st ACM International Health Informatics Symposium, Arlington, Virginia, USA, 2010, pp. 250–256.

[61] A. Machanavajjhala, D. Kifer, J. Gehrke, M. Venkitasubramaniam, L-diversity: privacy beyond K-anonymity, ACM Transactions on Knowledge Discovery from Data 1 (1) (2007 Mar).

[62] P. Maymounkov, D. Mazières, Kademlia: a peer-to-peer information system based on the XOR metric, in: Proceedings of First International Workshop on Peer-to-Peer Systems, London, UK, 2002, pp. 53–65.

[63] C. Mead, Data interchange standards in healthcare IT – computable semantic interoperability: now possible but still difficult do we really need a better mousetrap? Journal of Healthcare Information Management 20 (1) (2006) 71–78.

[64] N. Mohammed, X. Jiang, R. Chen, B.C.M. Fung, L. Ohno-Machado, Privacy-preserving heterogeneous health data sharing, Journal of the American Medical Informatics Association (JAMIA) (2013).

[65] NCBC Informatics for Integrating Biology and the Bedside, 2004. https://www.i2b2.org/

[66] OHT The HL7 Tooling Project, 2008. https://www.projects.openhealthtools.org/sf/projects/hl7tooling/

[67] OHT. The Model-Driven Health Tools Project, 2009. https://www.projects.openhealthtools.org/sf/projects/mdht/

[68] P. Rao, S. Edlavitch, J. Hackman, T. Hickman, D. McNair, D. Rao, Towards large-scale sharing of electronic health records of cancer patients, in: Proc. of 1st ACM International Health Informatics Symposium, Arlington, VA, 2010, pp. 545–549.

[69] P. Rao, B. Moon, An internet-scale service for publishing and locating XML documents, in: Proc. of the 25th IEEE Intl. Conference on Data Engineering, Shanghai, China, 2009, pp. 1459–1462.

[70] P. Rao, B. Moon, Locating XML documents in a peer-to-peer network using distributed hash tables, IEEE Transactions on Knowledge and Data Engineering 21 (12) (2009 December) 1737–1752.

[71] P. Rao, T.K. Swami, D. Rao, M. Barnes, S. Thorve, P. Natoo, A software tool for large-scale sharing and querying of clinical documents modeled using HL7 version 3 standard, in: Proc. of 2nd ACM International Health Informatics Symposium, Miami, FL, 2011.

[72] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Schenker, A scalable content-addressable network, Proc. of the 2001 ACM-SIGCOMM Conference 16 (2001) 1–172.

[73] C. Re, J. Brinkley, K. Hinshaw, D. Suciu, Distributed XQuery, Proc. of the Workshop on Information Integration on the Web 11 (2004) 6–121.

[74] RedHat. SSL/TLS, ECC, and RSA. http://docs.redhat.com/docs/en-US/Red_Hat_Certificate_System/8.0/html/Deployment_Guide/SSL-TLS_ecc-and-rsa.html, 2007.

[75] R.L. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Communications of the ACM 21 (February (2)) (1978) 120–126.

[76] A. Rowstron, P. Druschel, Pastry: scalable, decentralized object location and routing for large-scale peer-to-peer systems, in: Proc. of the IFIP/ACM Intl. Conference on Distributed Systems Platforms (Middleware 2001), Heidelberg, Germany, November 2001.

[77] J. Saltz, S. Oster, S. Hastings, S. Langella, T. Kurc, W. Sanchez, M. Kher, A. Manisundaram, K. Shanbhag, P. Covitz, caGrid: design and implementation of the core architecture of the cancer biomedical informatics grid, Bioinformatics 22 (15) (2006) 1910–1916.

[78] M. Scholl, K. Stine, K. Lin, D. Steinberg, Security Architecture Design Process for Health Information Exchanges (HIEs). http://csrc.nist.gov/publications/nistir/ir7497/nistir-7497.pdf, 2010.

[79] A.W. Services. Amazon Elastic Compute Cloud (EC2), 2011. http://aws.amazon.com/ec2/

[80] W.W. Stead, H.S. Lin, Computational Technology for Effective Health Care: Immediate Steps and Strategic Directions, The National Academies Press, Washington, DC, 2009.

[81] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, H. Balakrishnan, Chord: a scalable peer-to-peer lookup service for internet applications, in: Proc. of the 2001 ACM-SIGCOMM Conference, San Diego, 2001, pp. 149–160.

[82] L. Sweeney, K-anonymity: a model for protecting privacy, International Journal of Uncertainty Fuzziness and Knowledge-based Systems 10 (5) (2002 Oct) 557–570.

[83] M. Terrovitis, N. Mamoulis, P. Kalnis, Local and global recoding methods for anonymizing set-valued data, The VLDB Journal 20 (1) (2011 Feb) 83–106.

[84] Özlem Uzuner, I. Goldstein, Y. Luo, I. Kohane, Identifying patient smoking status from medical discharge records, Journal of the American Medical Informatics Association 15 (1) (2008) 14–24.

[85] G.M. Weber, Federated queries of clinical data repositories: the sum of the parts does not equal the whole, Journal of American Medical Informatics Association (2013).
[86] G.M. Weber, S.N. Murphy, A.J. McMurry, D. MacFadden, D.J. Nigrin, S. Churchill, I.S. Kohane, The shared health research information network (SHRINE): a prototype federated query tool for clinical data repositories, Journal of the American Medical Informatics Association 16 (5) (2009 Sept.) 624–630.
[87] X. Xiao, G. Wang, J. Gehrke, Differential privacy via wavelet transforms, IEEE Transactions on Knowledge and Data Engineering 23 (8) (2011) 1200–1214.
[88] Y. Xu, B. Fung, K. Wang, A.-C. Fu, J. Pei, Publishing sensitive transactions for itemset utility, in: Proceedings of the 8th IEEE International Conference on Data Mining, 2008, pp. 1109–1114.
[89] Y. Zhang, P.A. Boncz, XRPC: interoperable and efficient distributed XQuery., in: Proc of Very Large Data Bases, Vienna, Austria, September 2007.
[90] B. Zhao, L. Huang, J. Stribling, S. Rhea, A. Joseph, J. Kubiatowicz, Tapestry: a resilient global-scale overlay for service deployment, IEEE Journal on Selected Areas in Communications 22 (1) (January 2004) 41–53.
[91] caBIG architecture workspace: common query language SIG, summary and initial recommendations. https://cabig.nci.nih.gov/archive/SIGs/Common%20Query%20Language/ArchWSQuery%20SIG_Recomd_F2F_%20March05.ppt