Responses to openEHR platform product RFI

Context about CaboLabs

CaboLabs helps companies and organizations to build, improve and integrate systems and apps for health. Was founded in 2012 by Pablo Pazos Gutierrez, a Computer Engineer from Uruguay. He started to work with openEHR in 2006 and created the first online openEHR course in 2011.

Providing consultancy, training and coaching services, related to software architecture design, system / data integration, clinical database design, clinical database audit, product evaluation before acquisition, and implementation of standards like HL7, DICOM and openEHR. More info: https://cabolabs.com/en/

CaboLabs created EHRServer, the first open source clinical data repository compliant with the openEHR standard. CloudEHRServer is the SaaS cloud solution that can cut development time in eHealth projects, while standardizing all your clinical data and enabling interoperability. CloudEHRServer is provided on a subscription model.
Check our SaaS site: https://cloudehrserver.com/

EHRServer is an openEHR back-end system, marketed as a Clinical Data Management and Sharing Open Platform compliant with the openEHR standard. It is composed by a Web Console for management and audit, a REST API to integrate end-user apps, and a Vendor Neutral Clinical Data Repository. From the Web Console a manager can manage EHRs, audit Versioned Compositions and Contributions, create stored queries from our Query Builder (not AQL), manage Operational Templates (OPT 1.4), manage accounts, organizations, users and roles, and audit event logs.

This is not a system for end-users (clinicians, nurses, clinical technicians, etc.), is a tool for clinical information management, for managers. End user apps are created over the EHRServer and can store and retrieve data via the REST API.

The EHRServer can be translated to any language, currently supports Spanish and English.

EHRServer is open source (https://github.com/ppazos/cabolabs-ehrserver), and was the first open source implementation of an openEHR CDR (https://cabolabs.com/blog/article/ehrserver_from_proof_of_concept_to_an_open_source_saas_product-5a419d4a94600.html).

3.	Company Information	
3.1.	General	
3.1.1	Company name	CaboLabs
3.1.2	Company main office location	Montevideo, Uruguay
3.1.3	Company location in Sweden (cities)	No

3.1.4	Number of employees (total)	1 (we work with/through our network of partners in Argentina, Colombia, Panama, USA, Spain, Chile, Australia and Korea)
3.1.5	Number of employees in Sweden	0
3.1.6	Web address to company product site	https://cloudehrserver.com
3.2	Contact	
3.2.1	Name of sales contact	Pablo Pazos Gutiérrez
3.2.2	Email of sales contact	pablo.pazos@cabolabs.com
3.2.3	Phone number of sales contact	+598 99 043 145
3.2.4	Name of technical contact	Pablo Pazos Gutiérrez
3.2.5	Email of technical contact	pablo.pazos@cabolabs.com
3.2.6	Phone number of technical contact	+598 99 043 145
3.3	Partner	
3.3.1	Does the company have any sales partners in Sweden	No
4	Product information	
4.1	General	
4.1.1	Name of product	EHRServer / CLoudEHRServer
4.1.2	Current version of product	v1.2, releases can be checked here https://github.com/ppazos/cabolabs-ehrserver/releases
4.1.3	Number/size of installations?	1 cloud (multi-tenant, 5 companies using it), 2 test public servers (300+ users until 2017), don't have numbers on installations of open source solution, but we have 43 forks of the open source project https://github.com/ppazos/cabolabs-ehrserver
4.1.4	Describe the product update strategy (ex. number of major/minor update/year)	Revisions and minor updates (bug fixes, improvements, etc.) are released almost weekly to the community and updated monthly to the cloud server. Major versions (new features, integrations, refactors) are released each 2 or 3 months to the open source community and the cloud server is

		updated after each major release. Change requests and bug reports are filled as GitHub issues and major releases are managed as milestones: https://github.com/ppazos/cabolabs-ehrserver/milestones
4.2	Support	
4.2.1	Availability of support (24/7, 8/5 or other)	Any combination is possible, depends on the hired level of service. With current clients of the CloudEHRServer, chat support is available 24/7 and they got an answer right away (for clients on similar time zones, for clients with more than 8 hours of difference they get an answer in about 3 hours). Email and video conference are also available. I also tend to invite clients to workshops, demos and Q&A sessions so they can ask questions and have a little training for their teams to help implementing openEHR on the client side.
4.2.3	Availability of on-site installation support (free or billed)	All my current clients are remote, if I need to access a physical server on premise, we setup a VPN or do SSH using public/private key access. Also my current clients don't have servers on premise, they use the CloudEHRServer SaaS. If it is cost effective and the situation needs to be physically on-site, that can be coordinated.
4.2.4	Availability of Health (best practice) checks?	We do functional testing, load testing (https://cabolabs.com/blog/article/i_tried_to_kill_the_ehrserver-5a9a340db6ebf.html), and monitor the cloud server constantly. Also have daily contact with clients that report any issue as they happen, most are solved within the same day.
4.3	Licensing	
4.3.1	Describe the license model for the product	The product is free to use. We charge for service (installing, configuring, managing, coaching and training, support, development, design archetypes and templates, etc.). Commercial licensing is necessary if the customer is going to develop their solution using the EHRServer and selling their solution, including the EHRServer, to third parties, this is a development license. Other type of license is for whitelabelling (adapting the software to the look and feel of a company so they can provide the EHRServer to their clients as if that was developed by them).
4.3.2	Does the license model have options for setting up development and QA/environments (not for real patient care) that differs from production environment licenses?	All usage is free, production or QA/staging. As stated above, only if the client wants to extend the system or provide it as part of their solution, a paid development or whitelabelling license is offered.
4.3.3	Describe support agreement alternatives for the product	1. 24/7 with access to private chat room, email and video calls 2. 8/5 on the client's time zone, access to private chat room, email and video calls 3. time based, on demand: we offer X amount of hours monthly that can be used for support, training,

		coaching, development, etc. (this is a generic consultancy service)
4.4	Procurement & pricing	
4.4.1	Is the product offered through Swedish public sector framework agreements	No
4.4.2	If possible, please provide approximate price examples for some scenarios. Are there alternative price models regarding initial and recurring costs?	Our business model is not based on usage licenses. Number of patients or users doesn't affect the implementation costs in terms of license. There are two kinds of implementation models, SaaS cloud deployment and local on-premise deployment. For SaaS we charge for infrastructure, setup, maintenance and support. For local on-premise deployment we charge for setup, maintenance and support. On any case, recurring costs are based on the contracted service level for support and maintenance (based on time / month). We can also do knowledge transfer to have trained people on-site to make management/audit tasks. I can provide hardware server requirements (CPU, memory, disk)if you can provide an number of patients/EHRs (RFI mentioned 500K), estimated number of clinical documents that will be created per month, estimated number of queries per month. For on-premise installations, hardware should be provided by the customer.
4.4.3	How does your business model provide compensation if promised functions would be specified in a contract but would not be available in time as promised?	Separating what is already supported by the EHRServer and related services (currently available) from the features / systems that should be developed (not currently available). For the later, we can establish a schedule of payments based on deliverables, no payment until features are delivered and approve by the counterpart.
5	Functional requirements	
5.1	Basic framework	
5.1.1	What parts of the openEHR Reference Model Specification are fully implemented and according to what version of the specification?	EHRServer supports most of EHR, Common, Data Structures, Support and Data Types for openEHR 1.0.2. Also supports OPT 1.4 (openEHR 1.0.2). Since it is a purely clinical data repository, it doesn't include the Demographic RM (that should be implemented in an external system like a MPI). The EHRServer supports:

		1. Managing EHRs 2. Managing Versioned Compositions with Audit 3. References to demographic entities through Party Proxies 4. All concrete data types but the INTERVAL<T> ones (adding support for INTERVAL is on the roadmap) 5. We are currently working on improving the FOLDER management (FOLDERS are supported but we are working on making them more usable)
5.1.2	What parts of the openEHR REST API specification are fully implemented? What formats are supported? Are any other non standard REST APIs implemented?	The official openEHR REST API was released a couple of weeks ago. We don't have an implementation yet but it is on our roadmap. I personally participated of the SEC committee in charge of creating the official API. The EHRServer offers a REST API that is pretty close to the official REST API and uses only openEHR valid formats to commit data (other implementations use their own custom format). Both XML and JSON are supported. And for query results we will add other output formats like CSV in the future (useful for data extraction and analysis on a spreadsheet).
5.1.3	Is the openEHR Archetype Query Language specification fully implemented? Are there any additional capabilities?	AQL is not supported. We offer path based queries created from a GUI on our Query Builder inside the EHRServer, that can include complex conditions and support the addition of semantic conditions specified with SNOMED CT Expressions. https://cabolabs.com/blog/article/openehr__snomed_ct_a_perfect_combination_for_data_querying-5a440acd0f763.html
5.1.4	How is validation of EHR content done based on RM, archetypes and templates by the system? What types of versions of template-mechanisms are used for validation?	There are two levels of validation, syntactic validation is done against the Version XML Schema, since the Version is the commit unit. For the semantic validation, validating content against the definitions (OPT, archetypes), we are working with our partner VeraTech that has a validation framework that converts OPTs in content XSDs. This is under development.
5.1.5	Is GDL (Guideline Definition Language) (at least version 1.0, TRIAL DRAFT) supported? Are any other clinical decision support mechanisms available?	We provide SNOMED CT Expression integration inside openEHR data queries, and the concept of EHR Queries. This allows to create alerts and calculate some basic indicators. For instance, with the combination of those mechanisms, the EHRServer can return numbers of EHRs containing patients with certain gender, age range, that are obese, have a diagnosis of any type of diabetes (using SNOMED CT) and hypertension, also can return the specific EHR IDs matching that criteria. This can also be used for patient selection/recruitment for clinical trials. These are not formal CDS rules but are queries that can be used for CDS. Generic rule engine for CDS is not built on the EHRServer, another layer of rules should be implemented over the EHRServer to get data from the EHRServer queries and feed that to evaluate rules.

5.1.6	What parts of the new “Task Planning Model Specification” are implemented?	Task planning is a very new openEHR spec, I doubt any provider has an implementation of it besides the provider that requested/sponsored that specification to be created. It is not yet implemented on the EHRServer, but we plan to add support in the future, to maintain the execution state of instruction activities, and allow to query those structures.
5.1.7	What parts in the 5.1.x questions above that are not implemented right now will be available in September 2018?	Our plan is to have the official REST API implemented and also have semantic validation based on OPTs. Basic support for AQL and Task Planning will be added at some point, but is not a priority yet.
5.1.8	Describe available terminology service usage/integrations. Is the terminology service addressable from AQL queries? Is there a FHIR Terminology Service interface?	Since the EHRServer is a backend system, it doesn't require to use terminology services directly, that should be done at the end-user level. The EHRServer supports SNOMED CT Expressions in openEHR data queries based on archetype ids and paths, not using AQL yet.
5.2	Test & performance	
5.2.1	Please provide information and results from AQL query performance tests done for the product. (Have you for example run any of the “ORBDA” example tests?)	AQL is not supported yet.
5.2.2	Please provide information regarding other performance tests done or normal loads in significant real installations.	Performance depends on hardware infrastructure and usage. From our Cloud server, the performance of the Web Console and REST API is acceptable for a Web Application, most requests return requests in under a second, some complex queries need a couple of seconds. See appendix A for performance results using ApacheBench. We also did a heavy load test with burst EHR creation and clinical document commit to evaluate resource consumption more than performance https://cabolabs.com/blog/article/i_tried_to_kill_the_ehrserver-5a9a340db6ebf.html
5.2.3	The test cases/scripts in chapter 6 (“Conformance Schedule”) of the “openEHR EHR Platform Conformance” document are not finished, but when looking at the list of test descriptions, are there any of the listed capabilities your system has not yet implemented in some api-accessible form?	- ADL 1.4 (ADL is not supported by the EHRServer, it works with final OPTs 1.4, all ADL management is done before loading the EHRServer with OPTs) - OPT 1.4 (all requirements are met, upload, list and get is done via the Web Console, list and get can be done via API, not upload for security reasons) - ADL 2 archetypes and ADL 2 templates (EHRServer works with OPT 1.4, it doesn't support templates on ADL 2 yet) - Queries (queries can be created, deleted, listed and executed via the Web Console, can be created,

		listed and executed via the REST API)
5.3	Tooling & configuration	
5.3.1	Does the product contain an application development environment that enables applications, registries etc. to be built on the repository using openEHR data. Please describe.	The EHRServer is open source, can be downloaded and opened on any IDE that supports Java/Grails. The recommendation is to create apps externally and use the REST API to communicate with the back-end, rather than modifying the EHRServer core to add end-user functionalities. EHRServer is a back-end, management and audit system, users are managers not clinicians.
5.3.2	Is there a graphical drag and drop form generator (or similar functionality) available that makes it easy to create HTML5-based data entry forms (including client side validation and basic constraint checking) based on openEHR templates.	EHRServer is a back-end, management and audit system, users are managers not clinicians. Apps should be created separately from the EHRServer and connect via REST. UI generation and data validation on the apps is responsibility of the app developers.
5.3.3	Is there a function to render compositions as human-readable documents (resolving at/id-codes and hiding “technical” attributes)	Yes. Internally we use the official openEHR XML format, and XSLT to display contents for humans. Those can be accessed via the Administrative Web Console, and will be very simple to access the generated HTML from the REST API (not available yet, but can be added in a couple of hours if required).
5.3.4	Is an easy to use (e.g. drag-and-drop?) query editor available to create AQL queries based on Archetypes and Templates?	The EHRServer Query Builder is more "point and click" than "drag and drop". It is very easy to create queries selecting templates, archetypes and paths, and defining conditions over data points. This is not AQL.
5.3.5	Are functions like domains or namespaces available to achieve a logical separation of data between different care organisations using a physically shared server instance?	Yes. EHRServer is multi-tenant. It actually support creating different organizations on the same server instance,, and managing user permissions for each organization. Users can only access data from one organization at a time. Committed compositions, operational templates and queries are all associated with a specific organization.
6	Non-functional requirements	
6.1	Infrastructure	
6.1.1	List supported OS	It's all Java, any OS is supported.
6.1.2	Support for cluster configuration (describe)	Clusters can be configured at the RDBMS layer, this is set outside the app. We are working at the app layer on a sync API that will allow to create EHRServer instance clusters at the app layer.
6.1.3	List supported DBMS	It is based on Hibernate, any RDBMS is supported with minor adjustments. Default is MySQL.

6.1.4	Support of management packs for Microsoft System Center	The system has no dependencies with any virtualization technology. It can be virtualized using any platform. In production we have Apache Tomcat, MySQL, a SMTP server and the EHRServer deployed as a WAR on Tomcat. That can be a box that can be virtualized on any platform. Also it can be divided to have the DBMS on a different box.
6.1.5	Describe minimum hardware requirements for a test installation	Very minimal: 1 CPU/1 Core, 512 MB RAM, 1 GB HD, 10/100 Mbps network Minimal recommended: 1 CPU/4 cores, 2 GB RAM, 10 GB HD, 1000 Mbps network
6.1.6	Limitations on using virtualization (hardware/IaaS)?	None.
6.2	Security	
6.2.1	Support of role based authorization? Describe (default/typical) roles	Yes. We have four basic roles, that can be assigned to individual users over many organizations. Roles: ADMIN: super admin, can see/manage everything ACCOUNT MANAGER: manages an account and it's organizations ORGANIZATION MANAGER: manages specific organizations inside an account USER: can only access the REST API, not the Web Console
6.2.2	Support of authentication tickets issued by an Identity Provider (e,g, SAML)?	The Web Console has normal username/password authentication. Also the organization number is required to login. The API uses the same data to login and the app retrieves JWT (JSON Web Tokens) for successful logins, that are used on subsequent requests on the API.
6.2.3	Support of logging; access and change?	Yes. Every action form the Administrative Web Console and the REST API is logged. We plan to externalize that to a syslog server in the future to avoid storing audit data on the EHRServer database.
6.3	Training	
6.3.1	Availability of course or on-line training for administrators? Describe	CaboLabs brings many courses related to health information systems, standards and interoperability. We have plenty of experience in this area, giving courses and workshops internationally since 2011. We give openEHR-related courses for clinicians, project managers/administrators, and ITC professionals. From conceptual to very technical levels.

6.3.2	Availability of course or on-line training for technicians? Describe	Yes, see above.
6.3.3	Availability of course or on-line training for users? Describe	Yes. Note that users of the EHRServer are managers and auditors, not clinicians.
6.4	Usage	
6.4.1	Is the number of registered users limited, if so what is the limit?	No. Business model is not based on licenses that constraint number of users.
6.4.2	Is the number of simultaneous users limited, if so what is the limit?	No. Business model is not based on licenses that constraint number simultaneous of users.
6.4.3	Is the number of managed assets limited, if so what is the limit?	No. Business model is not based on licenses that constraint number of assets (EHRs, compositions, queries, OPTs, etc).
6.4.4	Does the license model allow usage for research as well as caregiving?	Our business model is not based on licenses. You can use the EHRServer for any purpose required. We especially support research, as all the data in stored in the EHRServer is anonymous, preserving privacy. https://cloudehrserver.com/learn/use_case_research_and_training https://cloudehrserver.com/learn/anonymous_clinical_information
6.4.5	Does the software product provide client libraries to support the development of software against the system, if so in what program languages?	Yes. There are many open source libraries available, and we plan to support more. Also we have a Swagger (Open API) specification of the EHRServer REST API, and Swagger has a code generation tool that can be used to generate clients for more than 20 different technologies. We also provide an Insomnia REST Client script, Insomnia also supports some code generation. Current libraries are: Groovy, PHP and Javascript (REST API clients). We also provide open source sample client applications (apps with GUI, not only REST API clients). All is accessible from my GitHub account https://github.com/ppazos
6.5	Management	
6.5.1	Is it possible to export system configuration between different instances of the installation? If so how?	Grails, the underlying dev framework, can use external configuration files, those can be copied between servers. Changing this is very simple.

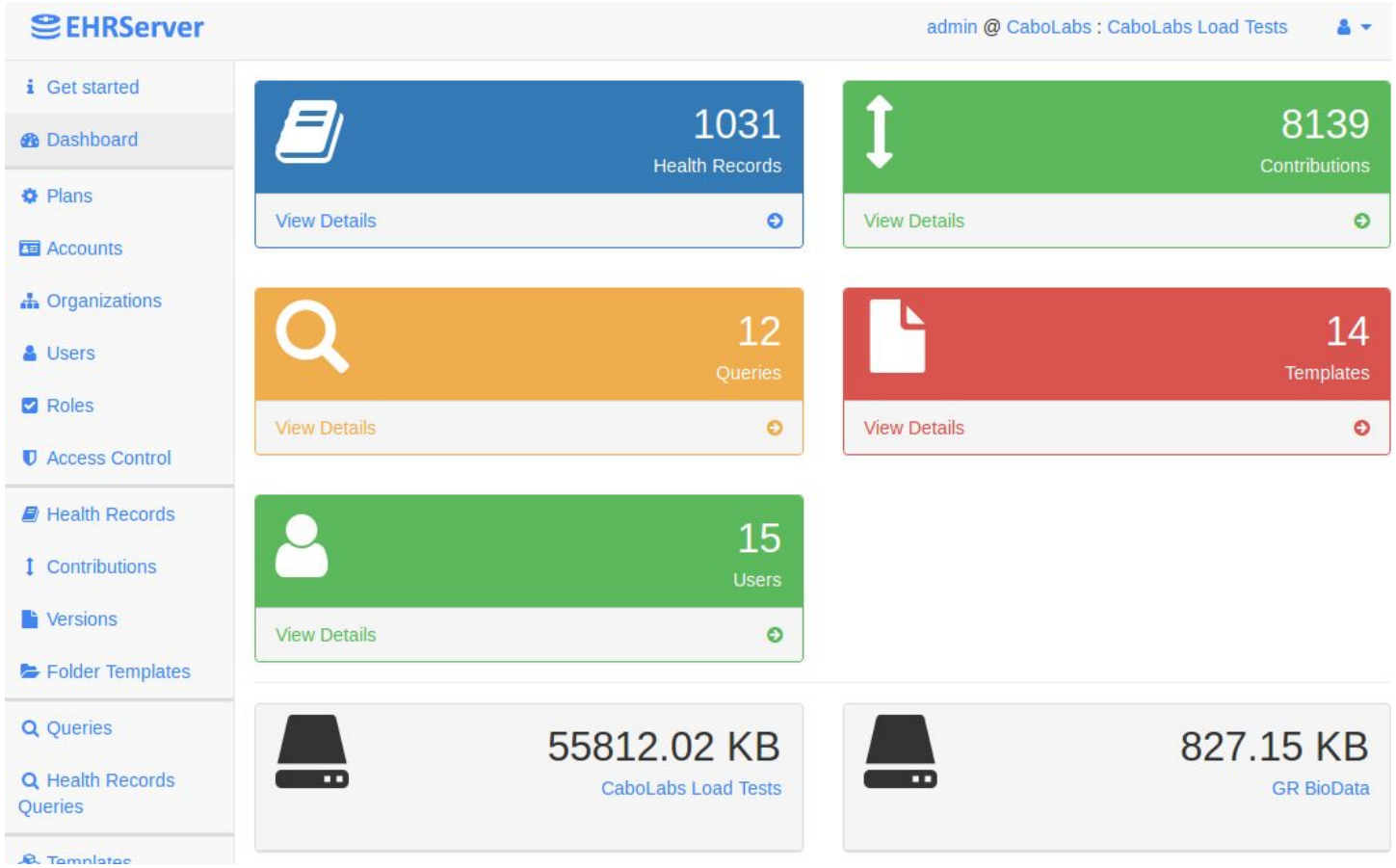
6.5.2	Is it possible to run multiple instances of the installation on the same network without conflicts? If so how?	Yes. Instances are independent from each other. Each server will have it's own IP.
6.5.3	Is it possible to run different versions of the same system simultaneously within the same instance?	My interpretation for "instance" physical server (hardware), and the question is about having different instances of the same software running on the same hardware. "system" can be interpreted as software+hardware. If that is the questions, yes, it can be done. Each software instance should be deployed on a different context on the same Tomcat, or under different ports.
6.5.4	Does the software allow soft launches of new versions?	Yes. We did just that in all 2017 for our Beta Partners. Gave early access and training for new releases before the public release. Also since this is open source, everyone can download the latest code and run it locally.
6.6	Integrations	
6.6.1	Does the software product have an interface to support import/export of HL7v2 messages?	We are working on receiving HL7 v2.x messages through a middleware (Mirth Connect), and transform that into the canonical openEHR format. Similar to this, CDA will also be accepted as an input format.
6.6.2	Does the software product have an interface to support import/export of HL7 FHIR messages?	We'll have support for receiving and retrieving FHIR compositions. This is not yet implemented, really waiting on HL7 to make FHIR normative.
6.6.3	Does the system support automated extraction of required IHE XDS.b data from openEHR compositions?	We are working on openEHR-XDS.b mappings. I proposed to have an official mapping to the openEHR Foundation, form the Specifications Editorial Committee (SEC), and we worked together with Ian McNicoll on an initial proposal. This is not yet implemented.
6.6.4	Does the system support extraction, mapping and storage of required DICOM metadata from KOS Objects to openEHR compositions	Not yet. We are working with Mirth Connect that allows to receive DICOM objects and do mappings to XML. We are planning to receive DICOM SRs this year, so KOS will be included on that. Mirth Connect uses DCM4CHE Toolkit inside, the Java implementation of DICOM services. We have used that technology since 2008, and Mirth since 2007.
6.6.5	Describe other integration support features of the platform	To support SNOMED CT expressions in queries, we actually use an external system that evaluates the expressions. That is a service from one of our partners. Via Mirth Connect (not directly implemented in the EHRServer) we can receive data in many formats (HL7 v2.x, XML, JSON, X12, CSV) and communication protocols (TCP, MLLP, HTTP, SOAP, DICOM, File Servers) and transform that into a valid openEHR XML instance that can be committed to the EHRServer. The same applies to output formats for queries, we can do conversions of the results to output any kind of format. All of these just requires to do some mapping work on the middleware.

Appendix A. Performance tests with ApacheBench

https://httpd.apache.org/docs/2.4/programs/ab.html

Context of the tests

We have loaded 1K EHRs and 8K clinical documents in total.



Server: Ubuntu 16.04, 1 CPU core, 2GB RAM, located in USA (EHRServer)
Client: Linux Mint 17.3, 4 CPU cores, 8 GB RAM, located in Uruguay (ApacheBench)

All tests are done by invoking the EHRServer REST API with a valid authorization token (authorization step was done previously and authorization time is not considered for the tests). All tests are single requests to the API and have different levels of complexity.

Considerations

Results depend on many factors, # of EHRs, # of Documents, # of documents containing each type of data used in the query criteria. Also on client and server machine configurations (since communications are between 2 machines, the client machine hardware also affects the results, that is #of cores, available memory, network speed), also the connection to the internet affects the results (is not the same a 56kbps connection or a 100 mbps connection).

Take into account that the pattern followed by AB is not a normal user pattern, AB is making burst requests, a lot of requests in a very short period of time. Depending on the types of applications, implementation techniques (like caching responses), and the final user needs, the usage patterns might vary, and also the response times and perceived performance by the end user.

A meaningful performance test should be binded to a specific usage pattern (so it can be simulated), database size, and specific data requirements. Also considering these elements, optimal server and client hardware configuration requirements can be estimated.

A final item to consider is that for specific usage patterns and requirements, optimizations can be implemented.

Considering all of these, the data presented here is merely anecdotic, and just serves as a reference ot the specific use cases and patterns used on each specific test.

Test 1: get first 30 EHRs in JSON

Individual response size: 6842 bytes
Total transferred: 7057000 bytes
Total requests: 1000

Individual response time reported by Insomnia REST Client: < 200 ms

Sample response:


```
{
    "ehrs": [
        {
                "uid": "11111111-1111-1111-1111-111111111111",
                "dateCreated": "2018-03-10 02:51:30",
                "subjectUid": "11111111-1111-1111-1111-111111111111",
                "systemId": "CABOLABS_EHR_SERVER",
                "organizationUid": "e9d13294-bce7-44e7-9635-8e906da0c914"
        },
...
```


Concurrency	Total time (s)	Requests per second	50% reponse under (ms)	98% response under (ms)	longest request (ms)	transfer rate (Kb/s)
5	77.634	12.88	390	432	442	88.77
10	44.768	22.34	452	510	552	153.94
20	27.806	35.96	566	649	698	247.84
50	16.825	59.44	838	1184	1536	409.61

Concurrency is the number of requests that are sent in parallel from AB.

Test 2: execute stored query, get clinical documents that contains diagnosis = any type of diabetes (using SNOMED CT expressions), from all EHRs

Individual response size: 9733 bytes
Total transferred: 9946694 bytes
Total requests: 1000

Individual response time reported by Insomnia REST Client: < 300 ms

Sample response:


```
{
    "results": {
        "6ddff437-b670-4db7-b633-7d3e54d4c419": [
                {
                        "uid": "13758272-eec8-43b4-b36b-1a48ff18a397",
                        "category": "event",
                        "startTime": "2011-03-25 18:20:14",
                        "subjectId": "4911a12d-657e-4ea8-893c-bc09422ca372",
                        "ehrUid": "6ddff437-b670-4db7-b633-7d3e54d4c419",
                        "templateId": "encuentro_diagnostico_codificado.es.v1",
                        "archetypeId": "openEHR-EHR-COMPOSITION.encounter.v1",
                        "lastVersion": true,
                        "organizationUid": "57919991-faa1-44c4-836d-da82cb8290dc",
                        "parent": "03fb281e-44e1-498f-8b2f-93c9178eacef::CABOLABS-LOADEHR::1"
                },
...
```


Queries with SNOMED expressions use an external service to evaluate the SNOMED expression, that result is cached to accelerate local execution of the queries without invoking the external service each time. Also improves performance for real time queries.

Concurrency	Total time (s)	Requests per second	50% reponse under (ms)	98% response under (ms)	longest request (ms)	transfer rate (Kb/s)
5	96.953	10.31	478	581	641	100.18
10	59.898	16.70	557	779	1835	162.17
20	39.215	25.50	789	1008	1309	247.71
50	26.98	37.07	1334	2247	2844	360.04

Test 3: execute query to get compositions containing records of low body mass index from all EHRs

Individual response size: 9535 bytes
Total transferred: 9749989 bytes
Total requests: 1000

Individual response time reported by Insomnia REST Client: < 200 ms

Sample response:


```
{
    "results": {
        "49cd207e-a85e-4fd8-9ba8-637f1dd5eabd": [
                {
                        "uid": "19dd7a01-d246-4011-83e9-77cf8aff6bfb",
                        "category": "event",
                        "startTime": "2016-12-10 00:57:25",
                        "subjectId": "a6e9920b-cef1-423f-b619-292f724c389c",
                        "ehrUid": "49cd207e-a85e-4fd8-9ba8-637f1dd5eabd",
                        "templateId": "control_del_peso.es.v1",
                        "archetypeId": "openEHR-EHR-COMPOSITION.encounter.v1",
                        "lastVersion": true,
                        "organizationUid": "57919991-faa1-44c4-836d-da82cb8290dc",
                        "parent": "ab448b7d-b250-4a14-9f94-93aed334dddb::CABOLABS-LOADEHR::1"
                }
        ],
...
```


Concurrency	Total time (s)	Requests per second	50% reponse under (ms)	98% response under (ms)	longest request (ms)	transfer rate (Kb/s)
5	127.079	7.87	629	757	898	74.93
10	92.759	10.78	923	1131	1237	102.65
20	68.874	14.52	1386	1747	1962	138.25
50	66.326	15.08	3245	5225	5578	143.57

Test 4: execute query to get body weight and BMI data from a patient (one specific EHR)

Individual response size: 909 bytes
Total transferred: 1123939 bytes
Total requests: 1000

Individual response time reported by Insomnia REST Client: < 200 ms

Sample response:


```
{
    "results": {
        "openEHR-EHR-OBSERVATION.body_weight.v1/data[at0002]/events[at0003]/data[at0001]/items[at0004]/value<DV_QUANTITY>": {
                "type": "DV_QUANTITY",
                "name": {
                        "ISO_639-1::es": "Peso",
                        "ISO_639-1::en": "Weight"
                },
                "serie": [
                        {
                                "magnitude": 111.0,
                                "units": "kg",
                                "date": "2010-03-07 01:02:29"
                        },
                        {
                                "magnitude": 138.0,
                                "units": "kg",
                                "date": "2015-01-30 18:36:41"
                        },
...
```


This kind of query is used to chart numeric data about vitals or lab results.

Concurrency	Total time (s)	Requests per second	50% reponse under (ms)	98% response under (ms)	longest request (ms)	transfer rate (Kb/s)
5	84.787	11.79	421	516	546	12.95
10	47.22	21.18	472	531	612	23.24
20	27.282	36.65	555	642	713	40.25

50	17.361	57.6	868	1296	1837	63.27

Test 5: execute EHR Query to get patients older than 35 years old, masculine, with obesity and any type of diabetes (using SNOMED CT expressions, internally executes the query from Test 2)

Individual response size: 5968 bytes
Total transferred: 6183000 bytes
Total requests: 1000

Individual response time reported by Insomnia REST Client: < 800 ms

Sample response: (list of EHR UIDs that match the criteria)

```
[
    "0269563b-31dc-48e7-b430-4d468de2078e",
    "02ff2fae-2b5a-44ad-89b6-76feae4dac83",
    "0608575a-3d8e-4182-81f5-2f849e6de62c",
...
```

Note: This kind of query is executed to get reports, for pop health, or to select patients to participate in certain care and wellness plans or clinical trials, thus executed in batch since it checks all the EHRs and takes some time to do so. The EHRServer provides the ability of creating such queries in minutes and then be able to run them from the Web Console or the REST API. The execution pattern for these kind of queries won't be in bursts like our test here.

Concurrency	Total time (s)	Requests per second	50% reponse under (ms)	98% response under (ms)	longest request (ms)	transfer rate (Kb/s)
5	606.413	1.65	3001	3668	3768	9.96
10	523.171	1.91	5198	6261	6627	11.54
20	524.29	1.91	10750	13027	14263	11.52
50	493.854	2.02	25382	28559	29590	13.29