

RFI Region Östergötaland

OpenEHR Technical Platform Product

Status

Alexander Theodorsen

15. mars 2018

Distribusjon: DIPS Konfidensiell

Contents

1	Introduction.....	1
2	Company Information.....	1
3	Product information	1
4	Functional requirements	4
5	Non-functional requirements	9
	Appendix 1	13

© 2018 DIPS AS.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of DIPS AS.

DIPS AS
Postboks 1435
8037 Bodø
Norway
<http://dips.no>
+47 75 59 20 00

1 Introduction

The answer for *Region Östergötaland OpenEHR Technical Platform* RFI is based solely on the DIPS EHR Craft Platform. To elaborate in more detail on the features and functionality, DIPS has taken the liberty in adding a more detailed description in a separate Appendix, Appendix 1.

In addition it is worth mentioning that DIPS have for the release of DIPS Arena 17.2, our enterprise EHR and patient administration system, developed an additional Surgery module on the basis of OpenEHR.

2 Company Information

3.1 General

3.1.1	Company name	DIPS AS
3.1.2	Company main office location	Bodø, Norge
3.1.3	Company location in Sweden (cities)	Stockholm
3.1.4	Number of employees (total)	292
3.1.5	Number of employees in Sweden	2
3.1.6	Web address to company product site	http://www.dips.com

3.2 Contact

3.2.1	Name of sales contact	Alexander Theodorsen
3.2.2	E-mail of sales contact	ath@dips.com
3.2.3	Phone number of sales contact	+4790574486
3.2.4	Name of technical contact	Bjørn Næss
3.2.5	E-mail of technical contact	bna@dips.com
3.2.6	Phone number of technical contact	+4793432910

3.3 Partner

3.3.1	Does the company have any sales partners in Sweden? (Y (names)/N)	N
-------	---	---

3 Product information

4.1 General

4.1.1	Name of product?	DIPS EHR Craft Platform
4.1.2	Current version of product?	Version 5.1.0

4.1.3	Number/size of installations?	Helse Nord ~ 500 000 inhabitants Helse Vest ~1 million inhabitants Helse Sør-Øst ~ 2.6 million inhabitants
4.1.4	Describe the product update strategy (ex. number of major/minor update/year)	Two major versions each year. Minor versions will be released depending on the bugs and errors the customer's reports.

4.2 Support

4.2.1	Availability of support? (24/7, 8/5 or other)	Customer service available on workdays between 8am to 4 pm. Stand-by support 24/7 for category A and B errors in production environments according to support agreement.
4.2.2		
4.2.3	Availability of on-site installation support? (Free or billed)	On-site installation support will be offered as a billed service

4.2.4	Availability of Health (best practice) checks?	<p>Health Checks is available. The EHR server supports health checks using the following endpoint: {ehr_server_url}/status/health</p> <p>This endpoint returns a JSON response according to the format given below:</p> <pre>{ "Version": "1", "Timestamp": "2018-03-09T08:15:47.3039Z", "IsHealthy": true, "Unhealthy": {}, "Healthy": { "Index": "Solr@https://vt-lab2-solr04.testlab.local:443/solr/man-testlab2-yr2-ehr_shard1_replica2: 43 ms", "Index Sync With Storage": "Storage and index are in sync.", "Storage": "oracle@man-testlab2-yr: 0 ms", "Unrecoverable Transactions": "0 unrecoverable transactions" } }</pre> <p>The most important field is IsHealthy, which is true when the following conditions are met:</p> <ol style="list-style-type: none"> 1. The database is running and reachable. 2. The index is running and reachable. 3. The index and database are synchronized. 4. There are no unrecoverable pending transactions to the index. <p>If any of the conditions are not met, the unhealthy field will contain the cause of the failing health check.</p>
-------	--	---

4.3 Licensing

4.3.1	Describe the license model for the product (CPU, user, other)	For later discussions.
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?	For later discussions.

4.3.3	Describe support agreement alternatives for the product	SLA agreements will be part of tender negotiations.
-------	---	---

4.4 Procurement & pricing

4.4.1	Is the product offered through Swedish public sector framework agreements (“Kammarollegiet” procurement contract) (E.g. via an existing Swedish partner)	Not at present. DIPS has had talks with companies with framework agreements through Kammerkollaget.
4.4.2	If possible, please provide approximate price examples for some scenarios. Are there alternative price models regarding initial and recurring costs?	For later discussions.
4.4.3	How does your business model provide compensation if promised functions (e.g. like described in 5.1.7) would be specified in a contract but available in time?	For later discussions.

4 Functional requirements

5.1 Basic framework

5.1.1 What parts of the the openEHR Reference Model Specification are fully implemented, and according what version of the specification?

We have implemented all parts of the OpenEHR Reference Model according to the latest versions.

Specification	Supported version
EHR	1.0.3
Demographic	1.0.4 (*)
Common	1.0.4
Data Structures	1.0.3
Data Types	1.0.3
Support	1.0.3
Integration	Not supported
EHR Extract	Not supported

(*) The reference model for Demographic is implemented. We have so far not used this specification for real data.

5.1.2 What parts of the the openEHR REST API Specification are fully implemented? What formats (e.g. JSON and XML) are supported? Are any other (non standard) REST APIs implemented?

We have implemented all functional parts of the OpenEHR REST API Specification. Currently we have not finalized the technical transformation into the formal formats defined in the OpenEHR REST API. This is a minor work which will be finalized Q2 2018.

We support the EHR, Query and LINK functions. For the definitions, we support Templates, Archetypes, GDLs and our proprietary VAQM definitions.

We have implemented a CDS endpoint. This is an endpoint to execute GDL definitions within a set of EHR's.

5.1.3 Is the openEHR Archetype Query Language specification (at least version 1.0, Trial Draft) fully implemented? Are there any additional capabilities, e.g. full text search, FOLDER-based filtering etc?

Yes – we have implemented the Archetype Query Language specification. We have put a lot of efforts in this part of the system. Based on our long term work with AQL we have faced some challenges which we have shared with the community. One example is the permutation issue: <https://github.com/DIPSASA/openehr-conformance/tree/master/aql> . DIPS AS want the AQL specification to be actively maintained and further developed to support the real needs from healthcare.

What is interesting with our server is the implementation of query scope and tags. These are functions to split the EHR into domain specific groups. Example of such use-cases will be episode of care, period of care, surgery planning, etc. This extension to the query model provides a powerful mechanism for clients who needs to scope the data within clinical contexts.

OpenEHR FOLDER is supported by the system. We have also implemented AQL functions that cover the scope/filter of data within folders.

Currently we have no support free text search within elements in the openEHR structure.

5.1.4 How is validation of EHR content done based on RM, archetypes and templates by the system? What types and versions of template-mechanisms are used for validation?

Validation is based on the imported Templates. Templates are added to the system using the REST API. We support Operational Templates version 1.4.

In the production system the validation is done on the EMR system integrated with the EHR Craft Server. The EMR system uses the software library from EHR Craft to do the validation. We will enable backend validation using the same software packages during Q2.

5.1.5 Is GDL (Guideline Definition Language) (at least version 1.0, TRIAL DRAFT) supported? Are any other clinical decision support mechanisms available?

EHR Craft Server has a rules engine that implements the OpenEHR GDL specification (version 1.0). This service is accessible through a REST service provided by REST API. Using the GDL editor, available from openEHR.org, users can create and edit Guides (GDL files), import the guides to EHR Craft Server and execute the guides using the REST interface either directly or by using EHR Craft.

Other CDS is based on calculation rules and JavaScript support in the Form Renderer. These features combined with reuse based on AQL provides a powerful platform to create CDS screens in the application.

5.1.6 What parts of the new “Task Planning Model Specification” are implemented?

We have currently not implemented the “Task Planning Model Specification”.

We have been deeply involved in the specification and have provided lots of input to the specification. We did this since we are highly focused on enabling vendor neutral clinical processes.

The implementation of this specification will be done during 2018 with a planned release in the end of the year.

5.1.7 What parts in the 5.1.x questions above that are not implemented right now will be available in September 2018?

1. OpenEHR REST API
2. Backend validation based on imported Templates

5.1.8 Describe available terminology service usage/integrations. Is the terminology service addressable from AQL queries? Is there a FHIR Terminology Service interface?

We have no integration with external terminology services.

5.2 Tests & 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?)

The AQL query performance of our product is measured in customer's production environment. We have extensive logging of the performance from different parts of the system, i.e. the web request, the query pipeline and the roundtrips and usage of the backend index (SolR).

We will provide new data from our production installations to show the AQL performance on request.

The performance tests of the product in test and development is done using Apache JMeter tests.

5.2.2 Please provide information regarding other performance tests done or normal loads in significant real installations.

We have no other performance tests.

5.2.3 The test cases/scripts in chapter 6 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?

- We have not implemented the ADL2 and OPT2 specifications
- The demographics support are not tested
- Some of the Admin functions might need to be implemented (i.e. EHR dump/load)
- Security and privacy will be handled by the clients using the CDR

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 product is developed in C# and provide lots of useful client libraries to work with OpenEHR templates, forms and data.

EHR Craft Studio is the desktop application embedding tools and components, and can be seen as an OpenEHR IDE, built to simplify the building and testing of clinical models like forms, queries, VAQM, charts, etc. EHR Craft is the tool made for developers and clinical modellers and can be easily distributed and is automatically updated through a web-service. This allows customers to distribute the development IDE to users outside the network domain. EHR Craft Studio can be used on top of both EHR Craft Server and Marand's Think!EHR.

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.

EHR Craft Forms provide a quick and scalable solution to create input forms with advanced functionality without writing code. This is especially useful when creating several similar, but not identical forms for different clinical situations. The forms provide a possibility for extremely simple and cost effective prototyping as well as development and testing in close collaboration with end-users.

The Forms module consist of a form designer and a form renderer.

Arena Form Designer lets users without programming skills create advanced forms with dependencies and calculations based on OpenEHR templates.

Arena Form Renderer is a .Net-component that renders a user-interface with layout, dependencies and validation based on the form definition. A Form Renderer for Web applications (HTML/Web App) is currently being implemented.

Note:

The definition of the forms is currently not a formal part of the OpenEHR specification but as vendors, both Marand and DIPS currently use the same basic format for design and rendering of forms and the specification is in process to be made open source.

Forms Specification

The following is a short description of the form specification. A form is defined by the following parts:

- Form description – the structural definition of the content. This is where the validation, user-input components, language, etc. is defined
- Form layout - defines how the components from the form description should be placed in the form
- Form dependencies - defines the dependencies between elements in the Compositions.
- Form manifest – metadata with information about the current form. Includes name, version, timestamp, etc.
- Form script – JavaScript that uses the jAPI defined by the renderer. This is a powerful way to build application within the context of an OpenEHR form.

The different parts are defined in files that are packaged in a ZIP archive.

Form Extensions

Tags and Annotations are attributes where the application adds extension to the basic form definitions. Tags are simple string based keywords. Annotations are key/value based attribute definitions. The semantic of Tags and Annotations is “only” known by the specific application using the form.

The development of EHR Craft forms extension is based on the experience of using the platform to create no-code applications as part of our own EMR suite, and provides a large set of extra features on top of the OpenEHR specification and joint forms definition. This makes it possible for customers to expand the form definitions with functions needed for a specific and complex application, without breaking the interoperability on template and archetype level.

To illustrate the properties for EHR Craft Platform's form functionality, some examples of Tags and Annotations that governs form functionality in EHR Craft are listed below. Currently a total of 28 annotations, 10 tags and 103 functions are supported. The total list can be shared on request.

Note:

Both Think!EHR and EHR Craft Platform support the use of annotations, tags and functions in its' form definitions, and have functionality to add such in their form designers. However, the support for different annotations, tags and functions in form will vary between different vendors form renders. DIPS makes all definitions of supported annotations, tags and functions publicly available in order for other vendors to be able to support the same. Annotations, tags and function ONLY influence the behaviour of forms (GUI), and have no impact on the stored data.

Tag	Description
IncludeInMasterDetail	If ShowAsMasterDetail set on a container then this tag will define which elements to include in the table. If no IncludeInMasterDetail is set then all elements in the container will be columns in the table.
ShowAsMasterDetail	Shows a container (CLUSTER/ENTRY) as a master-detail view. The active container is displayed as defined in the form, and the other entries are displayed in a table below. This is useful to make a compact form with many entries of the same type.
ShowAsWizard	When applied the form will be presented for the use one container at a time, with buttons for next and previous, the keys for this annotation is edit and "null". Edit will present the form as a wizard only when editing

Annotations

Annotation	Description
alertType	Defines how a container should be displayed. Normally this is used to change background color. Allowed values are: TIP, NOTE, INFO, INFORMATION, IMPORTANT, WARNING, SUCCESS, ERROR, EXCEPTION, TITLE
aql	Defines the AQL for reuse operations. Se SELECT annotation below.
calc	Use in fields for the purpose of showing results of functions (See function chapter)
constraint.func	Defines the constraint of a field. This could for instance be a past or future date. When using this annotation, the value could be a calculation that sets the constraint.
select	For reuse of data based on the AQL annotation above. Syntax {#variable:<path to instance>}
showAsWizard	Form works like a wizard with one container for each step. Renderer displays buttons for next and previous

stepInterval

Controls the precision level of any step interval in a forms element containing a DV_QUANTITY or DV_COUNT. Could be an integer or double.

5.3.3 Is there a function to render compositions as a human-readable documents (resolving at/id-codes and hiding technical attributes)

To render any OpenEHR composition we have developed some generic XSLT stylesheets. They transform the content into either PDF or HTML formats.

To leverage the functionality in EHR Craft Form Designer we have developed a format that combines the form description with data. The format is called Unify and we use it to define the format and data of OpenEHR compositions.

We are currently implementing an HTML5 web component which renders Unify definitions.

5.3.4 Is an easy to use (drag-and-drop?) query editor available to create AQL queries from Archetypes and Templates?

EHR Craft Studio provides functionality to present the content of Templates. From the Templates you can easily select any item and auto-generate AQL to inspect data from the element/container.

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?

This feature is not currently used to separate data from different organisations or systems. The feature is implemented into the server and the indexing of data. The server will be able to add the filters based on system-id in the identifiers.

5 Non-functional requirements

6.1. Infrastructure

6.1.1 List supported OS

The following software components is required to run the EHR Craft Server:

- Windows Server 2008R2 or newer
 - Docker is still experimental and recommended only for development and testing.

6.1.2 Support for cluster configuration (describe)

6.1.3 List supported DBMS

The server does not use any database specific functions. This makes it possible to support any DBMS which has the needed transactional capabilities. Currently we have implemented support for the following DBMS:

- Oracle
- Microsoft SQL Server

The following DBMS are planned:

- MySQL
- PostgreSQL

6.1.4 Support of management packs for Microsoft System Center

Currently not supported.

6.1.5 Describe minimum hardware requirements for a test installation

There are several possibilities for a test installation:

- Minimal installation of a total server
 - Database
 - Apache SOLR
 - EHR Craft Server
- Docker container
- In memory OpenEHR CDR for testing purposes (will be implemented during 2018)

The minimal installation of a total server will typically run fine on a normal laptop. Further definition on the hardware requirements depends on the size of the test installation.

6.1.6 Limitations on using virtualization (hardware/laaS)?

No known limitations.

6.2 Security

6.2.1 Support of role based authorization? Describe (default/typical) roles

Currently the server is implemented as an open backed server. The authorization is performed by the owning client.

6.2.2 Support of authentication tickets issued by an Identity Provider (e.g, SAML)

Authentication is only supported by using client certificates. Authenticated users have full access to all functions. Authorization and access control must be performed by the client.

6.2.3 Support of logging; access and change?

There is an extensive logging of the use of the OpenEHR server. All artefacts are versioned which makes it possible to track changes over time.

6.3 Training

6.3.1	Availability of course or on-line training for administrators? Describe	On-site or skype course will be offered. Documentation will be provided.
-------	---	---

6.3.2	Availability of course or on-line training for technicians? Describe	On-site or skype course will be offered. Documentation will be provided.
6.3.3	Availability of course or on-line training for users? Describe	On-site or skype course will be offered. Documentation will be provided.

6.4 Usage

6.4.1	Is the number of registered users limited, if so what is the limit?	No limits
6.4.2	Is the number of simultaneous users limited, if so what is the limit?	No limits
6.4.3	Is the number of managed assets limited, if so what is the limit?	No limits
6.4.4	Does the license model allow usage for research as well as caregiving?	Yes
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?	C#/.NET

6.5 Management

6.5.1	Is it possible to export system configuration between different instances of the installation? If so how?	This is done by using the REST API. We have implemented a package manager system for distribution of configurations. This will be enabled in the OpenEHR CDR during 2018.
6.5.2	Is it possible to run multiple instances of the installation on the same network without conflicts? If so how?	Yes -each server is set up with the backed configuration.

6.5.3	Is it possible to run different versions of the same system simultaneously within the same instance?	Yes - to make migration into newer versions we have implemented a solution to run different versions on the same system.
6.5.4	Does the software allow soft launches of new versions?	No.

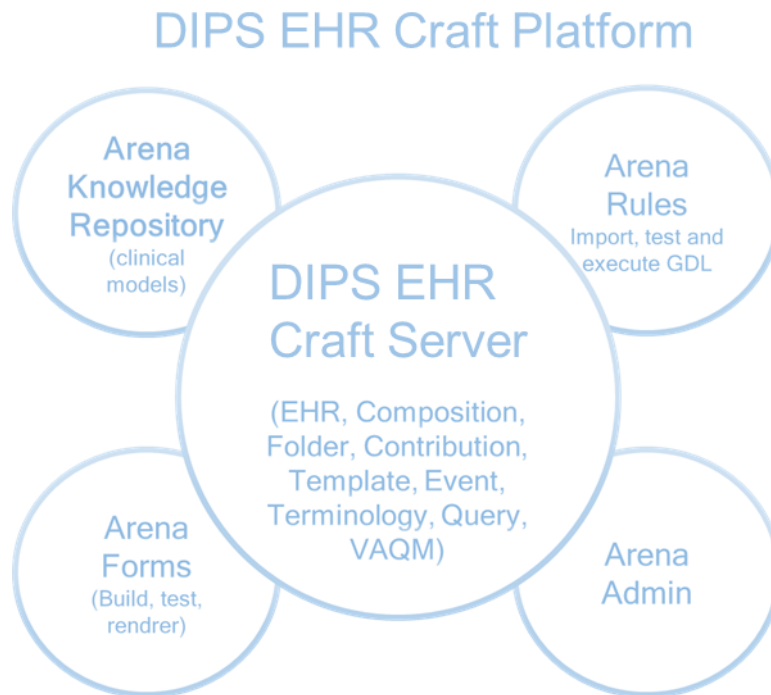
6.6 Integrations

6.6.1	Does the software product have an interface to support the automated import of HL7v2 messages?	<p>No. Our EHR-server does not have this in place. The integration layer (Connectivity Suite) is part of our EHR/PAS-product. However, our strategy implies moving towards a standalone EHR-server, separated from our PAS. This product will comply with relevant integration interfaces and profiles such as IHE XDS.b, FHIR, OpenEHR and HL7 v.2.</p> <p>DIPS have extensive knowledge in the integrations and interoperability domain.</p>
6.6.2	Does the software product have an interface to support the import of HL7 FHIR messages?	No. See 6.6.1
6.6.3	Does the system support automated extraction of required IHE XDS.b data from openEHR compositions?	No. See 6.6.1
6.6.4	Does the system support automated extraction, mapping and storage of required DICOM metadata from KOS Objects to openEHR compositions	No. See 6.6.1

Appendix 1

DIPS EHR Craft Platform

The DIPS EHR Craft Platform is a concept and product for a computable health information platform. The purpose of the platform is to give developers tools, components and services to develop functionality fast, consistent and standardized. The platform contains several modules as shown in the figure below.



DIPS EHR Craft Server

DIPS EHR Craft Server is an OpenEHR server implementation. It is the key component of DIPS EHR Craft Platform.

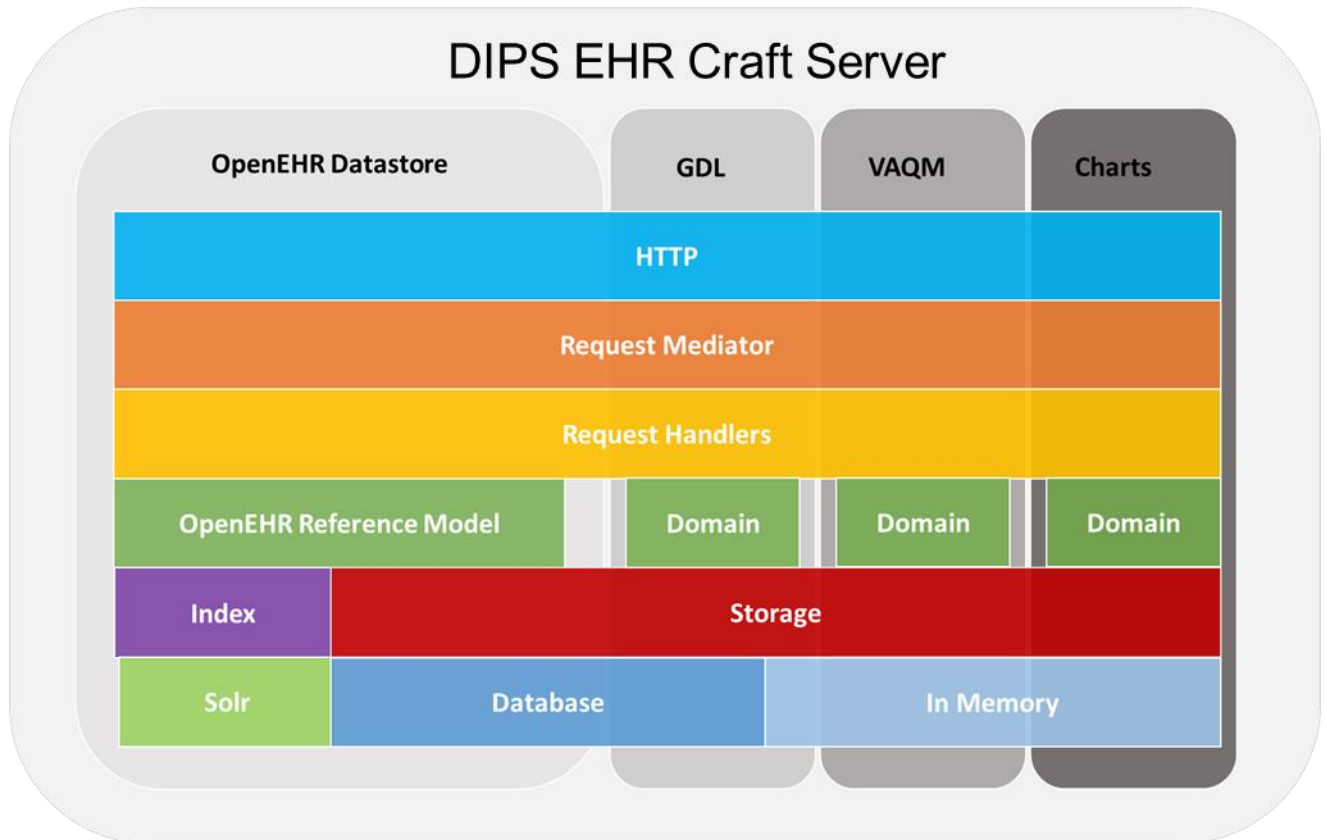
The central function of the OpenEHR server is to manage Compositions. The composition endpoint has functions for basic CRUD operations on entries. All compositions in an OpenEHR system is bound to a subject of care, which is the EHR object with the primary identifier of the subject of care. The EHR service implements operations to create and retrieve EHR information.

One of the central elements of the OpenEHR server is the ability to query for structured data across compositions and EHRs. The archetype query language (AQL) is a powerful language to express queries to retrieve data from the saved compositions. The query service accepts AQL queries as argument to retrieve and filter data.

On top of the OpenEHR specifications, DIPS EHR Craft implements a virtual archetype query model (VAQM). VAQM can be compared to *stored queries*, but includes more features and enables simplified application development.

DIPS EHR Craft Server Architecture

In the following, we provide an overview of the architecture of DIPS EHR Craft Server. This is extract of the full documentation, which will be provided on delivery.



5.1.1.1 HTTP/Rest Interface

Interaction with DIPS EHR Craft Server is through the REST interfaces. The primary REST services are EHR, Composition and Query. These service endpoints are compatible with Think!EHR/EHRScope and openEHR REST API v1.0

5.1.1.2 OpenEHR Reference Model

DIPS EHR Craft Server and its components are compliant with OpenEHR Reference model in current version. DIPS is active in the OpenEHR community and we update the implementation as the specification evolves.

A common use-case is to store elements bound to Terminologies. Below is an example showing how a coded text is bound to SNOMED-CT terminology. This is the OpenEHR way of representing both the Terminology ID and actual code. All these attributes are available when executing AQL on the data.

```

<items xsi:type="ELEMENT" archetype_node_id="at0121">
  <name>
    <value>Procedure request</value>
  </name>
  <value xsi:type="DV_CODED_TEXT">
    <value>Cytologic examination of breast</value>
    <defining_code>
      <terminology_id>
        <value>SNOMED-CT</value>
      </terminology_id>
      <code_string>372277005</code_string>
    </defining_code>
  </value>
</items>

```

5.1.1.3 Database

DIPS EHR Craft Server is database agnostic. We have most experience using Oracle DB as this is the preferred RDBMS with our current customers. DIPS EHR Craft supports other databases, i.e. PostgreSQL.

For a developer or clinical modelling setup, DIPS EHR Craft Server may also use filesystem as database. This is of course not recommended for production data.

5.1.1.4 Apache Solr

DIPS EHR Craft Server uses Solr for indexing of data. For production data it is necessary to have running Solr instance so that the server functions correctly. The Solr indexing is one of the techniques used to give the DIPS EHR Craft Platform its high performance.

There are two possibilities for configuring connections to Solr:

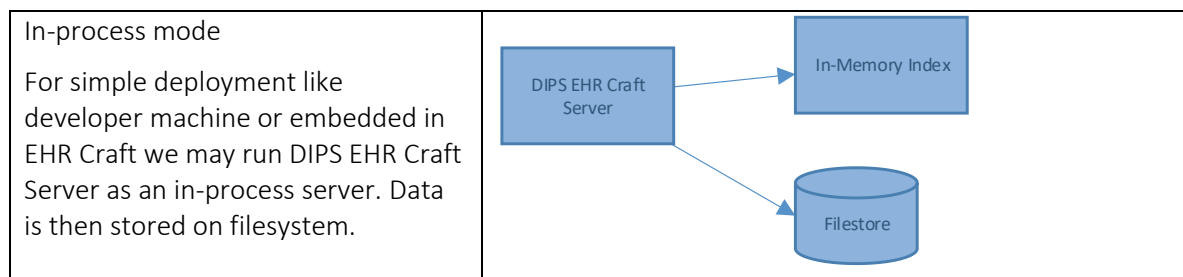
- Solr Standalone – just configure the server with the URL to the Solr server.
- Solr Cloud – uses Apache Zookeeper to enable highly reliable distributed coordination

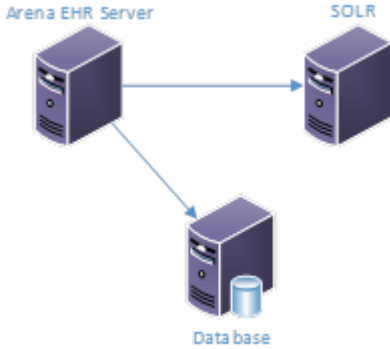
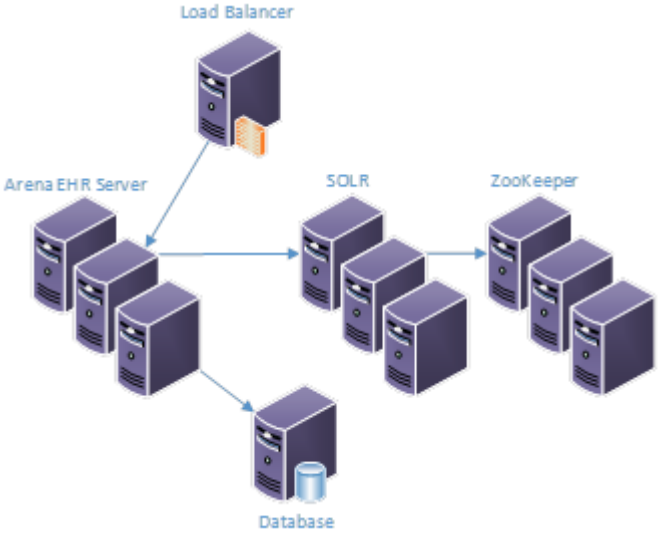
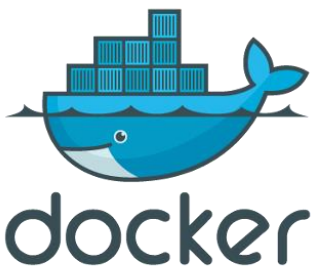
For large scale deployment of mission critical systems, index synchronization needs to be handled. The index synchronization feature in DIPS EHR Craft Server is designed to handle a multi-environment setup.

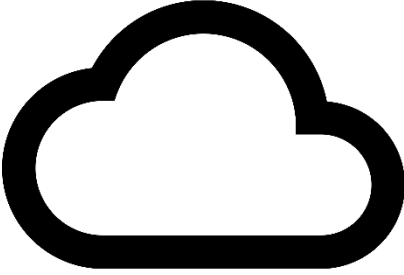
The detailed description on Solr indexing and index synchronization is provided on delivery.

Deployment

DIPS EHR Craft Server is a modular server implementation. It is designed to function from a small setup based on in-process usage to the most complex deployments in the largest hospitals. Below we provide some examples. The specific deployment model for this delivery must be worked out in cooperation.



<p>Simple deployment</p> <p>One host may have all processes installed.</p> <p>As load and data volumes increase, customers may deploy to several hosts.</p>	 <p>A diagram illustrating a simple deployment architecture. It shows three server icons: 'Arena EHR Server' on the left, 'SOLR' on the right, and 'Data base' at the bottom center. Arrows indicate connections: one from 'Arena EHR Server' to 'SOLR', one from 'Arena EHR Server' to 'Data base', and one from 'SOLR' to 'Data base'.</p>
<p>Complex deployment</p> <p>For large-scale installation, DIPS EHR Craft Server support more complex deployment with several datacentres and load balancing the traffic between several nodes.</p>	 <p>A diagram illustrating a complex deployment architecture. At the top center is a 'Load Balancer' server. Below it, on the left, is a group of three 'Arena EHR Server' icons. In the middle is a group of three 'SOLR' icons. On the right is a group of three 'ZooKeeper' icons. At the bottom center is a 'Database' server. Arrows show the following connections: from the 'Load Balancer' to each of the three 'Arena EHR Server' icons; from each of the three 'Arena EHR Server' icons to the 'Database' server; from each of the three 'Arena EHR Server' icons to each of the three 'SOLR' icons; and from each of the three 'SOLR' icons to each of the three 'ZooKeeper' icons.</p>
	<p>DIPS EHR Craft Server supports Docker.</p>

<p>CLOUD</p> 	<p>DIPS EHR Craft Server have been tested on Microsoft Azure. The deployment is based on Docker containers for each component: DIPS EHR Craft Server, Apache Solr and Database.</p>
--	---

DIPS EHR Craft Knowledge Repository

DIPS EHR Craft Knowledge Repository is a module in DIPS EHR Craft server to manage different clinical models. Archetypes, Templates and Guidelines (GDL) are models that are edited by using tools available from openEHR.org. In addition, DIPS EHR Craft Platform supports a set of other knowledge resources, described below:

- Forms (see separate chapter on Arena Forms)
- VAQM
- Charts
- Folder definitions

VAQM

VAQM (Virtual Archetype Query Model) is a model of a structured query and presentation model on top of OpenEHR data. The model is designed with two main purposes in mind:

- Be able to do define structured and reusable queries for simplified application development
- Compact definition of the presentation of OpenEHR data

VAQM defines a set of paths that may be used to express the presentation of the data. Below is an example of VAQM expression for blood pressure. This expression is compact and human readable, which makes it easy to build applications on the OpenEHR reference models.

```
$bp.systolic/magnitude / $bp.diastolic/magnitude $bp.systolic/units
```

VAQM editor is a tool to create and edit definitions. This editor is embedded in EHR Craft.

A full description of the VAQM format will be provided on delivery.

Charts

Based on VAQM we have developed a model to define chart datasets. Chart editor is a lightweight editor to create Chart definitions. The editor is embedded in EHR Craft. Chart is available as REST services in DIPS EHR Craft Server and simplifies development of especially graphical visualisation of data.

Note:

VAQM and Charts are features not specified in the OpenEHR-standard currently. Their existence is based on the experience on developing on top of the OpenEHR stack, and simplifies the development of applications. In the same way as Marand has spearheaded REST APIs through EHRscape, leading to an OpenEHR specification proposed for REST APIs, the specifications for VAQM and Charts are also made public so that other vendors can implement and also proposed as expansion of the OpenEHR standard. We look at it the same way we do with web-browsers: Different vendors will develop new features on top of the standards, leading to expansion of the standards, and higher expectations to functionality among the users. It is however important for developers to be aware of the difference between features based on the specification and other useful, not-yet-standard features that will at least for a period of time limit platform interchangeability and application portability. It is however important to note that these supra-standard features in no way affect the data, and data interoperability.

Folder definitions

FOLDER is a class defined in OpenEHR. DIPS have developed a specification to be able to create and commit folders into the OpenEHR server. DIPS EHR Craft has an embedded renderer for FOLDER instances.