Integrating image management and analysis into OpenClinica using web services

Thomas M. Deserno[a1], Daniel Haak[a], Christian Samsel[b], Johan Gehlen[a], Klaus Kabino[a]

[a]Department of Medical Informatics, RWTH Aachen University, Germany
[b]Chair of Communication and Distributed Systems, RWTH Aachen University, Germany

ABSTRACT

Although image-based measures have become an important surrogate for primary endpoints in controlled clinical trials, electronic data capture (EDC) insufficiently supports image and signal data files. In this paper, we suggest a simple extension of OpenClinica, the world’s largest open source EDC system, to handle image data files, process image and signal data, and fill out the electronic case report forms (eCRF) accordingly. We use the web service server interface that is integrated with OpenClinica. The missing client component is substituted by CRF embedded JavaScript and a PHP proxy on server side. JavaScript is also used to display images within the OpenClinica interface. The counterpart system was developed using the Google Web Toolkit (GWT) and the Java application programming interface (API) for eXtensible Markup Language (XML) web services (JAX-WS). Image processing is implemented in Java using ImageJ libraries. We demonstrate the workflow for CRFs of a conjunctival provocation test, where two photographs of a human eye are captured, transferred into the eCRF, segmented and measured. The secure file transfer protocol (SFTP) is used to transfer the data files between the systems, and web services are used to fill the eCRFs, which also integrate resulting images generated by the analysis process. Both, images as well as computed measures are automatically displayed within the OpenClinica eCRFs and can be evaluated by the study nurse after file upload. This allows re-capturing of images in case of evaluation failure, and avoids elaborative query management. In future, DICOM-based data transfer will be implemented.

Keywords: Controlled clinical trials, Electronic data capture (EDC), Clinical data management (CDM), Case report form (CRF), Image analysis, Signal analysis, Integration, Interface, Protocol, OpenClinica

1. INTRODUCTION

Image-based measures have become an important surrogate for primary endpoints in controlled clinical trials [1]. Replacing the paper-based case report form (CRF), electronic CRFs (eCRF) have been established in such trials. All entries can be evaluated since range checks or more complex rules are applied immediately at data entry. If necessary, values can be corrected directly before data entry is completed. This avoids elaborative query processing, improves data quality, and saves costs of studies. During the last years, OpenClinica has been established as the world's leading open source clinical trials software for electronic data capture (EDC) and clinical data management (CDM) [2]. It is used to collect, manage, and store data on clinical trials in an electronic database, and studies using OpenClinica have been approved by regulatory authorities such that the Federal Drug Administration (FDA).

However, image processing and image-based surrogates are not supported sufficiently in OpenClinica. In a worst case scenario, which however is still practiced, multi-center clinical trials are performed where, image or signal data is captured, manually uploaded to a central computing center, and centrally processed. The obtained values are sent back to the study site by mail or fax, and a local study nurse completes the CRF manually re-entering the image-based measurements.
In this paper, we aim at interfacing OpenClinica to support integrated image and signal data processing without changing its implementation structure and code. This will ensure compatibility with future releases of the OpenClinica system.

[1] Corresponding author: Thomas M. Deserno (nè Lehmann), Department of Medical Informatics, RWTH Aachen University, Pauwelsstr. 30, D - 52057 Aachen, Germany, email: deserno@ieee.org; phone: +49 241 80 88793, fax: +49 241 80 33 88793.

Medical Imaging 2013: Advanced PACS-based Imaging Informatics and Therapeutic Applications,
edited by Maria Y. Law, William W. Boonn, Proc. of SPIE Vol. 8674, 86740F
© 2013 SPIE · CCC code: 1605-7422/13/$18 · doi: 10.1117/12.2008004

2. MATERIAL AND METHODS

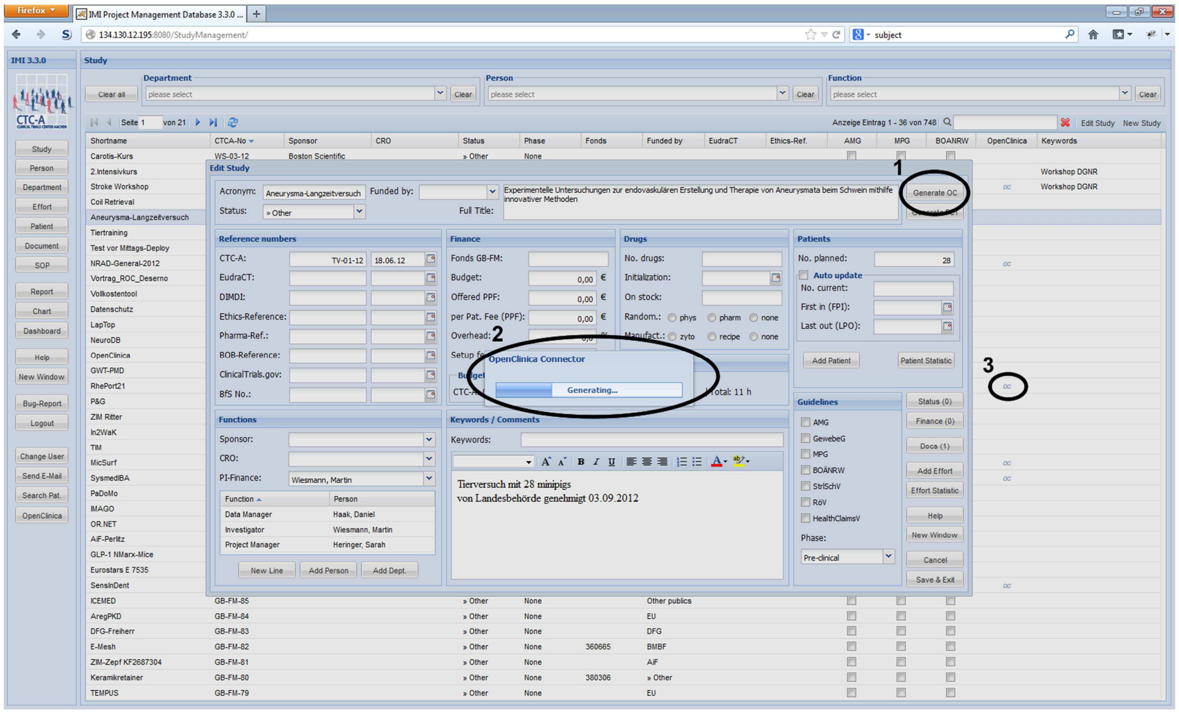
For conception, existing systems and previous publications have been analyzed, resulting in ideas of a communication architecture and workflow.

2.1 Existing systems

As a unit of the medical school of RWTH Aachen University, the Clinical Trial Center Aachen (CTC-A) is managing and supporting clinical trials at Aachen University Hospital. In particular, a study management tool (SMT) has been developed, where all important study-metadata information is stored, such as the name of the study, the study sites, and the study personal including their roles and access levels. This information is digitally mapped in the web application, which has been developed with the Google Web Toolkit (GWT) [3]. Through any web browser, the user can enter, modify and visualize data in charts or reports..

In previous work [4], a concept has been presented, where meta-data on clinical trials is automatically transferred from the Study Management Tool (SMT) [5] into OpenClinica, which is used to manage medical data captured in the trials. The user can easily transfer this information into OpenClinica and connect studies between both systems. This workflow is illustrated in Fig.1: (1) Activation of a button in the detail view of a certain study in the SMT triggers the study transfer; (2) web service transfer status is illustrated in a progress bar and (3) user can simply login in OpenClinica by clicking on the OpenClinica logo, after completion of transfer.

To provide this connection between the CTMS and EDC system, OpenClinica web services (OC-WS) are used, which is part of the OpenClinica system structure. OC-WS provides web service functionality in OpenClinica, including Simple Object Access Protocol (SOAP) support. Inside OC-WS various endpoints can be addressed, which allow modification of study, user and site information in OpenClinica. The data transfer is triggered by a web service client in the SMT, which has been integrated with the Java application programming interface (API) for eXtensible Markup Language (XML) web services (JAX-WS) [6]. So far, OC-WS only offers a web service server, which receives and processes SOAP messages. A web service client that triggers communication with other systems is not embedded.



Edit Study

Acronym: Aneurysma-Langzeitversuch
Funded by:
Status: » Other
Full Title: Experimentelle Untersuchungen zur endovaskulären Erstellung und Therapie von Aneurysmata beim Schwein mithilfe innovativer Methoden

Reference numbers

CTC-A:	TV-01-12	18.06.12
EudraCT:		
DIMDI:		
Ethics-Reference:		
Pharma-Ref.:		
BOB-Reference:		
ClinicalTrials.gov:		
BfS No.:		

Finance

Fonds GB-FM:	
Budget:	0,00 €
Offered PPF:	0,00 €
per Pat. Fee (PPF):	0,00 €
Overhead:	0,0 %
Setup fe	

Budget
CTC-A | Total: 11 h

Drugs

No. drugs:
Initialization:
On stock:
Random.: ○ phys ○ pharm ○ none
Manufact.: ○ zyto ○ recipe ○ none

Patients

No. planned: 28
☐ Auto update
No. current:
First in (FPI):
Last out (LPO):

Functions

Sponsor:
CRO:
PI-Finance: Wiesmann, Martin

Function	Person
Data Manager	Haak, Daniel
Investigator	Wiesmann, Martin
Project Manager	Heringer, Sarah

Keywords / Comments

Keywords:

Tierversuch mit 28 minipigs
von Landesbehörde genehmigt 03.09.2012

Guidelines

☐ AMG
☐ GewebeG
☐ MPG
☐ BOÄNRW
☐ StrlSchV
☐ RöV
☐ HealthClaimsV

Phase: Pre-clinical

Shortname	CTCA-No	Sponsor	CRO	Status	Phase	Fonds	Funded by	EudraCT	Ethics-Ref.	AMG	MPG	BOANRW	OpenClinica	Keywords
Carotis-Kurs	WS-03-12	Boston Scientific		» Other	None					☐	☐	☐		
2.Intensivkurs														Workshop DGNR
Stroke Workshop													oc	Workshop DGNR
Coil Retrieval														
Aneurysma-Langzeitversuch														
Tiertraining														
Test vor Mittags-Deploy														
NRAD-General-2012													oc	
Vortrag_ROC_Deserno														
Vollkostentool														
Datenschutz														
LapTop														
NeuroDB														
OpenClinica														
GWT-PMD														
RhePort21													oc	
P&G														
ZIM Ritter														
In2WaK														
TIM														
MicSurf													oc	
SysmedIBA													oc	
PaDoMo														
IMAGO														
OR.NET														
AiF-Perlitz														
GLP-1 NMarx-Mice														
Eurostars E 7535														
SensInDent													oc	
ICEMED	GB-FM-85			» Other	None		Other publics			☐	☐	☐		
AregPKD	GB-FM-84			» Other	None		EU			☐	☐	☐		
DFG-Freiherr	GB-FM-83			» Other	None		DFG			☐	☐	☐		
E-Mesh	GB-FM-82			» Other	None	360665	BMBF			☐	☐	☐		
ZIM-Zepf KF2687304	GB-FM-81			» Other	None		AiF			☐	☐	☐		
Keramikretainer	GB-FM-80			» Other	None	380306	» Other			☐	☐	☐		
TEMPUS	GB-FM-79			» Other	None		EU			☐	☐	☐		

Figure 1: OpenClinica integration in the Study Management Tool.

In addition, images and other file-based data can be uploaded into an eCRF of OpenClinica, but neither visualization nor processing is comprehensively supported. The data is collected simply in a special directory of the file system and OpenClinica stores the filename and location in the database table. Based on JavaScript code that is plugged into the eCRF definition, dynamic image display has been suggested for OpenClinica [7].

2.2 Concept of interfacing

The concept of interfacing includes extensions of the system architecture of OpenClinica and a GWT application, which is illustrated exemplary with the SMT, and a workflow process, that pictures the user-triggered communication steps between those web applications.

2.2.1 Systems architecture

The suggested structure is visualized in Fig 2. On the OpenClinica Site (Fig. 2, right side), the OpenClinica web service and core components are visualized. OpenClinica's system architecture is composed of: (i) the OpenClinica core system that is used for interaction, data entry and user interfacing; (ii) the OC WS that allows automatic data transfer into the system; (iii) a database management system (DBMS) that is provided by PostgreSQL and persistently stores the data in relational tables, and (iv) the connection to the server's file system. To avoid comprehensive alteration of OpenClinica, the missing WS client is not added in our concept, but the versatile CRF is enhanced by JavaScript simulating a web service client that invokes the communication. Since the data files themselves are stored outside the database, file exchange is proposed using the secure file transfer protocol (SFTP).
On the GWT application side (Fig. 2, left side) the architecture is composed of: (i) an ExtGWT user interface; (ii) JAX-WS server and client module to receive and process or trigger web service message, respectively; (iii) the Java image analysis tool for image processing, using ImageJ libraries; (iv) a MySQL database, which provides a DBMS; (v) connection to the server's file system, and (vi) a SFTP client.

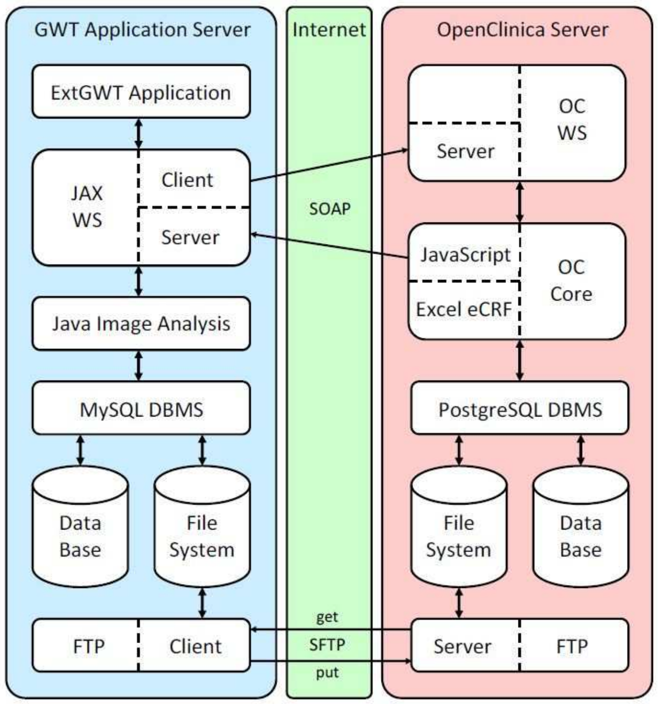


Figure 2: Architecture of GWT application and OpenClinica.

2.2.2 Workflow

Fig. 3 depicts the workflow and data transfer. The left hand side refers to metadata, while the figure's right hand side depicts the transfer of image or signal data files. The protocols interfacing OpenClinica with image processing are SOAP and SFTP. The user initiates the process via the browser interface. Supporting the workflow, the eCRFs are filled in Step 1 by the study personal, and the image data is uploaded using the file upload tool as provided by the OpenClinica software. Now, two different processes are invoked: On the metadata transfer side, JavaScript code embedded in the eCRF definition is used to send a SOAP message with an image ID to the computer system performing the image processing (Step 2a) and to display the user the uploaded images (Step 3a). The JAX-WS server receives this message (4a) and calls the image analysis software (Step 5a). Meanwhile on image transfer side, the image is stored by OC core in the file system (Step 2b) and gets visible for the user (Step 3b). In the next step, the SFTP image transfer is invoked (Step 4b) and sent to the FTP server (Step 5b). Now, the Java image analysis software queries for the transferred image (Step 6) and starts processing after the image file is available (Step 7). After finishing image analysis, measurements and additional images to visualize the results are provided and the tool invokes the transfer back to OpenClinica (Steps 8a, 8b). The measurements information is sent back on the metadata transfer side using the JAX-WS web service client (Step 9a), that sends a SOAP message to the, by OC-WS offered, data endpoint. The measurements are forwarded to OC Core (Step 10a) and stored (Step 11a). At the same moment, on image transfer side, the result image is send back analogy to the initial transfer by SFTP and saved in OpenClinica's file system (Steps 9b-11b). The modified data elements are now available in OC Core (Step 12), integrated into the eCRF and visible for the user (Step 13).

3. RESULTS

In the results, we present implementation details of our concept realization and illustrate our solution based on the application of conjunctival provocation test (CPT) [8].

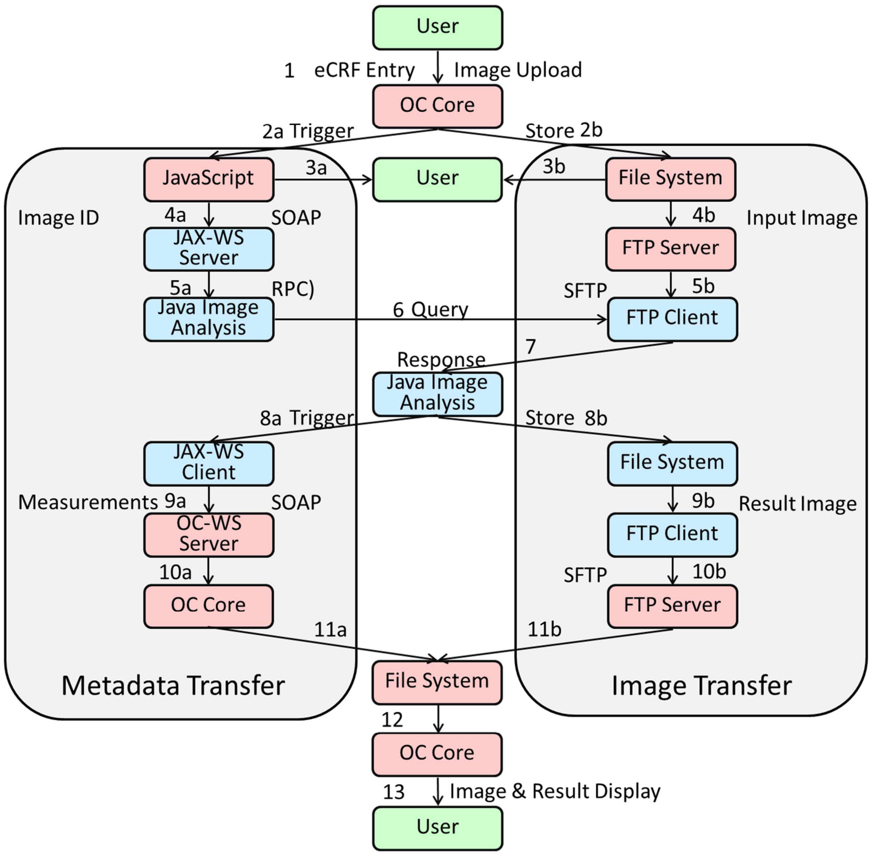


Figure 3: Workflow of GWT application and OpenClinica communication.

```
1  <script>
2
3  function callPHPProxy(){
4
5  // Prepare PHP proxy call
6  xmlhttp = new XMLHttpRequest();
7  xmlhttp.open('GET','http://oc-server/phpproxy.php?imageID=42', true);
8
9  // Send request
10  xmlhttp.send(null);
11 }
12 </script>
```

Listing 1: JavaScript module calls PHP proxy and provides parameters.

3.1 Implementation

In general, the concept has been realized as presented. Images can be uploaded to OpenClinica, triggering a Java image analysis module on a counterpart system, resulting data and images are transferred back and illustrated in OpenClinica's eCRF. However, in implementation the concept has been affected by small modifications.

Comparing with the initial concept workflow (cf. Fig. 3), the process is started by the user in step 1, initializing an image upload in OpenClinica. Steps 2a and 3a are performed accordingly and the image is directly visualized to the user's interface through CRF embedded JavaScript code. Following the concept, the JavaScript module sends a SOAP message with the ID of the uploaded image to the JAX-WS web service part of the GWT application (Step 4a). In realization of this concept step, it became obvious that this solution conflicts with the same origin policy [9,10] of web browser. For security reasons, it is not allowed to perform cross domain calls from client side scripts like JavaScript. For script programming languages, which are usually stored on a web server, but executed on the client side after transfer to the user's browser, it is prohibited to request or modify elements without the same origin. In this case, "same origin" means same hostname, protocol and port. In our concept, we outlined to transfer data between computer systems with different origins. Since the same-origin policy, this is not possible with JavaScript. Following this regulation and providing internet security, the web browser automatically ignores corresponding JavaScript code and does not perform the commands as expected. There are various workarounds to perform cross domain calls anyway [11,12]. For our realization, we forward the call action and parameter to a PHP proxy on the OpenClinica server. The code of the corresponding JavaScript method callPHPProxy() is illustrated in Listing 1. In Lines 6 and 7, a XMLHttpRequest object is created and prepared by inclusion of necessary parameters. This object is sent to call the PHP on server side (Line 10). Since the PHP proxy is invoked on the OpenClinica server, it has the same origin as the JavaScript module and browsers support this type of communication.

Listing 2 shows code details of the PHP proxy. Here, the parameters are extracted from the JavaScript call (Line 4) and a SOAP request is built (Lines 7-8). For this we use the SOAPClient object, which is provided by PHP and allows easily using SOAP functionality in PHP code. In Line 11, we send the SOAP message and call the method startImageAnalysis() on the GWT server, accordingly to Step 5a in the concept (cf. Fig. 2). With this solution, we successfully realized the invocation of the Java image analysis module on GWT server side by uploading an image in OpenClinica (Steps 1-5a).

At the same time, adopting the concept, the uploaded image has to be transferred from OpenClinica to the GWT server by SFTP (Steps 4b,5b). This is done inside the web service server method startImageAnalysis() in the GWT application. Here, the image transfer is performed before starting the image analysis process. For this, the Java Secure Channel (JSch) library by JCraft [13], that offers easy handling of SFTP connections and data transfer in Java, has been used. The corresponding Java code is illustrated in Listing 3. After creating a JSch object (Line 4), we set the required session information (Lines 7-10) and open a channel to the OpenClinica server (Lines 13-15). With a simple get-Statement in

```
<?php

// Get parameters
$imageID = ($_POST['imageID']) ? ($_POST['imageID']) : ($_GET['imageID']);

// Create SOAP client
$client = new SOAPClient('http://gwt-server/image?wsdl');
$client->__setLocation('http://gwt-server/image');

// Invoke SOAP method
 $client->startImageAnalysis(array('imageID' => $imageID));
?>
```

Listing 2: PHP proxy creates SOAP message and calls OC-WS.

Line 18 the image transfer to the local GWT server is initiated. After completion of image transfer, the image analysis Java program is invoked by the callJavaImageAnalysis()-statement.

Now the Java images analysis is performed, that produces measurements and/or result images. Following Step 8b of the concept, the result image is transferred back to OpenClinica server within the existing SFTP session and channel by invocation of the put-method (Line 24). In addition, measurement information is forwarded by calling sendSOAPResults() to the JAX-WS client (Step 8a).

At this moment, the result image is already transferred back to the OpenClinica server and the measurements information transfer has been initiated. Now, we just have to prompt OpenClinica to integrate the image and the measurement in the eCRF (steps 8a-13). In general, it is possible for user to import data in OpenClinica and fill eCRF fields automatically by using a data import web interface. Here, a well-formed Extensible Markup Language (XML) file in Clinical Data Interchange Standards Consortium Object Data Module (CDISC ODM) [14,15] format has to be uploaded, where field values are assigned to OpenClinica Object Identifiers, that identify eCRF input elements. For this data import routine, the OC-WS also provides a import()-method [16]. In preparation, a SOAP message is created that includes a XML file in CDISC ODM format in the body part. OC-WS receives the SOAP message, extracts the XML file and automatically triggers the data import interface to fill the eCRF.

After this action, the image and data results of the Java image analysis are included in the eCRF and, therefore, the workflow is successfully completed.

3.2 Example workflow

Our implementation is based on a concrete workflow example for conjunctival provocation test (CPT) [8], where images of eyes are analyzed and eye redness measurements extracted. For data acquisition inside a study, which is performing this test, the corresponding eCRF is filled automatically with those measurements after image upload by a study nurse. As outlined in the concept, the communication and image analysis part is almost completely hidden for the user. Figs. 4 and 5 show the OpenClinica eCRF data input interfaces, which are visible for the user and illustrate the workflow: A control and a provocation image is captured and uploaded manually (Fig. 4). Both images are processed automatically to quantify the response to allergic solution that is administered into the lower conjunctival sac of the eye between capturing baseline and follow-up images [8]. The segmented region of interest (ROI) is visualized in OpenClinica, and the measures are filled in the appropriate fields automatically, awaiting final approval by the study nurse (Fig. 5).

```
public String startImageAnalysis(String imageID){

    // Create JSch SFTP object
    JSch jsch = new JSch();

    // Configure session
    Session session = jsch.getSession("user", "oc-server");
    session.setConfig("StrictHostKeyChecking", "no");
    session.setPassword("password");
    session.connect();

    // Configure channel
    Channel channel = session.openChannel("sftp");
    channel.connect();
    ChannelSftp sftpchannel = (ChannelSftp) channel;

    // Get image
    sftpchannel.get(filename,local_filename);

    // Call Java module
    callJavaImageAnalysis();

    // Transfer result image back
    sftpchannel.put(local_result_filename, result_filename));

    // Close channel and session
    sftpchannel.exit();
    session.disconnect();

    // Trigger JAX-WS to send SOAP message with results
    sendSOAPResults();

    return "Success!";
}
```

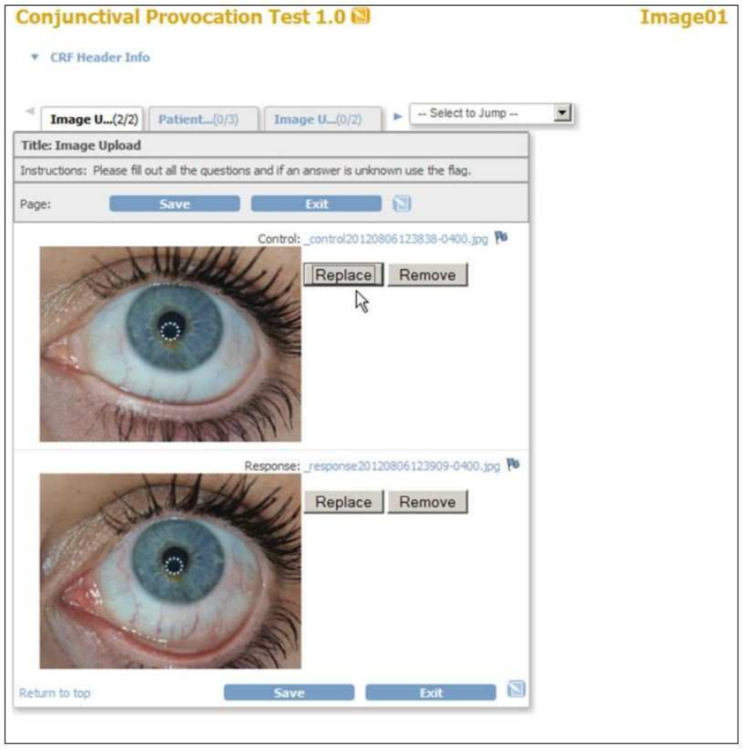
Listing 3: GWT web service server method transfers image files and triggers result transfer.

4. DISCUSSION

In this work, a concept for automatically processing image data files with OpenClinica has been presented. In implementation, this concept has been successfully realized. As result, an extension to OpenClinica is provided that allows OpenClinica to handle image data files, process images, and fill out eCRFs with image analysis results.

In general, the concept follows the idea to provide image processing functionality by extending OpenClinica with modules and without modification of the OpenClinica core implementation. Hence, the OpenClinica core module can be updated without any conflicts or loss of functionality. In addition, image and metadata transfer can be parallelized and de-coupled. This is advantageous for future integration of DICOM-based image data communication and supports the user's workflow allowing him to continuously filling the CRF while larger image data is still transferred. On the other side, the possibilities for realization of this functionality and restricted and new code modules need to be adapted to the OpenClinica systems. Therefore, in implementation, the concepts had to be adapted by small modifications.

The JavaScript SOAP client had to be extended by a PHP proxy to avoid not allowed cross domain calls. In usual, many approaches are possible to solve this issue (i.e. Flash, JSONP or CORS) [11]. We decided for a server proxy realization with PHP, because this solution is supported by all established browser application, not very costly in implementation,



Title: Image Upload

Instructions: Please fill out all the questions and if an answer is unknown use the flag.

Page:

Control: _control20120806123838-0400.jpg

Response: _response20120806123909-0400.jpg

Figure 4: Upload of CPT images in OpenClinica.

prevents inclusion of new security vulnerabilities and is still not invasive for the OpenClinica implementation. However, in other system architectures and server configurations, other solutions could offer more advantages.

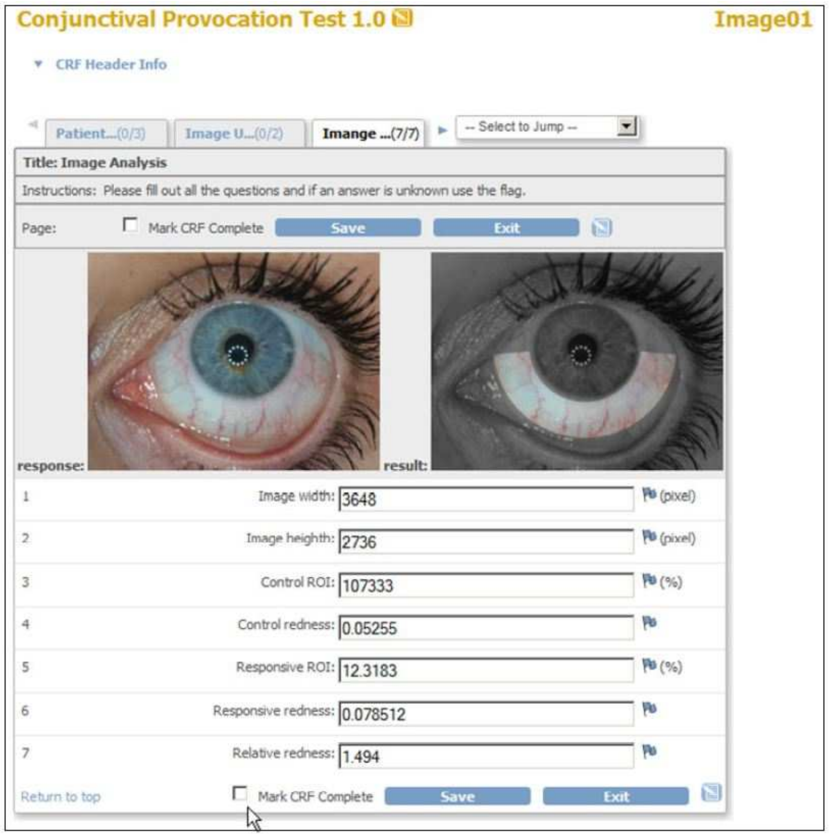
In general, our realization follows the goal in providing a first prototype implementation. Therefore, various aspects for improvement still exists. For the prototype implementation, focusing on estimation of suitability of the concept several security mechanisms have been circumvented. For example, no SOAP security headers have been used and server access has not been restricted. To provide a solution that is applicable in daily routine, the concept and realization should be extended by established security mechanisms.

In addition, in our case, the Java image analysis software is installed on the GWT server, but remote procedure calls (RPC) may also be employed to run the image analysis remotely.

5. CONCLUSION

In conclusion, a concept for image processing and analysis integration into OpenClinica for complete data management in controlled clinical trials supporting image-based surrogate endpoints has been presented. The interface uses JavaScript code in the eCRF definition and the OpenClinica web services server component that is already part of the system. Our concept does not require changes in OpenClinica source codes and does not affect OpenClinica updates. The concept has been successfully implemented with small modifications in an applicable prototype. As an example, the workflow of the extension for a CPT has been illustrated, where photographic non-DICOM images from a multi-center study are taken, centrally collected, persistently stored, and automatically processed in the eCRF of OpenClinica.

In future, the suitability of the prototype should be evaluated regarding image based data acquisition in other medical applications. Furthermore, the realization of the concept should be improved by integration of security mechanisms and evaluated in clinical daily routine. To provide more possibilities in interfacing, DICOM support will be integrated.



Title: Image Analysis

Instructions: Please fill out all the questions and if an answer is unknown use the flag.

Page: ☐ Mark CRF Complete

1	Image width:	3648	(pixel)
2	Image heighth:	2736	(pixel)
3	Control ROI:	107333	(%)
4	Control redness:	0.05255	
5	Responsive ROI:	12.3183	(%)
6	Responsive redness:	0.078512	
7	Relative redness:	1.494	

☐ Mark CRF Complete

Figure 5: CPT result visualization in OpenClinica.

REFERENCES

[1] Analoui M. Quantitative medical image analysis for clinical development of therapeutics. In: Deserno TM. Biomedical Image Analysis. Berlin: Springer; 2011. p. 359-375.
[2] Leroux H, McBride S, Gibson S. On selecting a clinical trial management system for large scale, multicenter, multi-modal clinical research study. Stud Health Technol Inform 2011;168:89-95.
[3] Google Web Toolkit – Google Developers [Internet]. 2012 Oct 25 [cited 2013 Feb 3]. Available from: https://developers.google.com/web-toolkit/
[4] Deserno TM, Samsel C, Haak D, Spitzer K. Data, function, and context integration of OpenClinica using web services. Proc GMDS 2012 (in German).
[5] Deserno TM, Deserno V, Legewie V, Schafhausen J, Eisert A, Schmidt-Kotsas A et al. IT-support for translational management in clinical trials based on the Google Web Toolkit. Proc GMDS 2011 (in German).
[6] JAX-WS Reference Implementation – Java.net [Internet]. 2013 [cited 2013 Feb 3]. Available from: http://jax-ws.java.net/
[7] OpenClinica User Manual/ShowingAnImage –Wikibooks, open books for an open world [Internet]. 2011 July 27 [updated 2012 Mar 21; cited 2013 Feb 3].
Available from: http://en.wikibooks.org/wiki/OpenClinica_User_Manual/ShowingAnImage
[8] Bista SR, Mösges R, Dogan S, Astvatsatourov A, Deserno TM. Automatic conjunctival provocation test combining Hough transform and self-calibrated color measurements in plain photography. Procs SPIE 2013; 8670(91) in press.
[9] Hallaraker O, Vigna G. Detecting malicious JavaScript code in Mozilla. ICECCS Proc. 2005; 85- 94.
[10] Flanagan D. JavaScript: The Definitive Guide. O'Reilly Media, Inc;2011.

[11] Ullmann C. Calling Cross Domain Web Services in AJAX [Internet]. 2006 Dec 29 [cited 2013 Feb 3]. Available from: http://www.simple-talk.com/dotnet/asp.net/calling-cross-domain-web-services-in-ajax/
[12] Brinzarea-Iamandi, B, Darie, C. AJAX and PHP: Building Modern Web Applications : Build User-Friendly Web 2.0 Applications with JavaScript and PHP. Packt Pub;2009.
[13] JSch – Java Secure Channel [Internet]. 2012 [cited 2013 Feb 3]. Available from: http://www.jcraft.com/jsch/
[14] ODM Certification & Archiving and Interchange of Metadata [Internet]. 2013 [cited 2013 Feb 3]. Available from: http://www.cdisc.org/odm
[15] Breil B, Kenneweg J, Fritz F, Bruland P, Doods D, Trinczek B et al. Multilingual medical data models in ODM Format - a novel form-based approach to semantic interoperability between routine healthcare and clinical research. Appl Clin Inf 2012;3: 276-89.
[16] Data Web Service – OpenClinica Reference Guide [Internet]. [cited 2013 Feb 3]. Available from: https://docs.openclinica.com/3.1/technical-documents/openclinica-web-services-guide/data-web-service#content-title-3002