Karlsruhe Institute of Technology

Interaktive Nutzung von Aufklärungsergebnissen an einem Multi-Display-Arbeitsplatz zur Bildauswertung

Bachlorarbeit
von

Kai Westerkamp

An der Fakultät für Informatik
Fraunhofer IOSB (IAD)

Erstgutachter:	Prof. Dr.-Ing. habil. Jürgen Beyererr
Zweitgutachter:	Prof. Dr. rer. nat. Hartwig Steusloff
Betreuender Mitarbeiter:	MSc. Sebastian Maier

Bearbeitungszeit: 15.11.2014 – 15.03.2015

Abstract

A multi-display image interpretation workstation is developed by Fraunhofer IOSB. The workstation consists of four displays which display different tools that support the image analyst in his work. The main tools are image representation of monoscopic and stereoscopic images, a display of meta information, map display and different assistance systems. The workstation is equipped with an innovative operating concept which integrates new devices in addition to keyboard and mouse. This enables new input methods such as touch, interpretation of pointing gestures and of head rotation of the user.
In this work, various software components will be integrated to improve the efficiency of the image analyst. It is most effective to support the user in routine tasks. The objective of this work is a direct connection to the Coalition Shared Data Server (CSD). The CSD stores reconnaissance data of all connected systems. The multi-display workstation shall recieve reconnaissance results from this database and write back the results of a successful evaluation.

Zusammenfassung

Am Fraunhofer IOSB entsteht der Bildauswerteplatz der Zukunft. Der Multi-Display-Arbeitsplatz besteht aus vier Displays, auf denen unterschiedliche Werkzeuge dargestellt werden, die den Bildauswerter bei seiner Arbeit unterstützen. Dieses umfasst die Darstellung von monoskopischen und stereoskopischen Bildern, die Anzeige von Metainformationen, Kartendarstellung bis hin zu Assistenzsystemen. Hinzu kommt ein innovatives Bedienkonzept, welches nicht nur auf Tastatur und Maus beruht, sondern auch neuartige Eingabemethoden wie Touch, Zeigegesten und die Kopfdrehung des Nutzers integriert.
Im Rahmen dieser Arbeit findet die Integration der verschiedenen Werkzeuge statt, welche sich an einem kompletten Arbeitsablauf eines Bildauswerters orientieren und ihn bei Routineaufgaben unterstützt. Ziel dieser Arbeit war die direkte Anbindung an die Coalition Shared Data Server (CSD), welche das Abfragen und das Einstellen von Aufklärungsdaten aller beteiligten Systeme ermöglicht. Aus dieser Datenbank sollen die Aufklärungsaufträge abgerufen und die Ergebnisse nach erfolgreicher Auswertung eingestellt werden.

Inhaltsverzeichnis

1. Einleitung

In diesem Kapitel werden zunächst der Multi-Display-Arbeitsplatz zur Bildauswertung und dessen Komponenten vorgestellt. Im Kapitel 2 wird das Design der neuen interaktiven Features dargestellt. Insbesondere wird das Laden und Speichern von Auswertungsergebnissen aus einem Coalition Shared Data Server behandelt. Kapitel 3 befasst sich mit der Implementation der einzelnen Komponenten. Schließlich wird in Kapitel 4 die Code-Qualität betrachtet und in Kapitel 5 ein Ausblick gegeben, welche Features in Zukunft die interaktive Nutzung von Aufklärungsergebnissen am Bildauswerteplatz der Zukunft verbessern können.

1.1. Bildauswerteplatz der Zukunft

Der Bildauswerteplatz der Zukunft ist ein Arbeitsplatz, der einen Bildauswerter bei seiner Arbeit optimal unterstützen soll. Es sollen alle erforderlichen Werkzeuge für die Bildauswertung zur Verfügung stehen. Der Bildauswerteplatz ist mit vier 1080p Bildschirmen ausgestattet (siehe Abbildung 1.1). Der mittlere obere Bildschirm ist ein 3D Bildschirm und kann z.B. stereoskopische Luftbildaufnahmen anzeigen. Der untere flach liegende Bildschirm ist ein Touch-Bildschirm. Er kann z.B. verwendet werden, um Kartenmaterial der Umgebung anzuzeigen.

1.1.1. Stereo-Bildauswerter

Der Stereo-Bildauswerter (SBA 1.2) ist ein Programm zur Annotation von 2D- und 3D-Bilder, basierend auf dem JHotDraw GUI Framework zur Erstellung von grafischen Editoren[1]. Die Annotation umfasst klassische Einzeichnungen von Bildverarbeitungsprogrammen wie Text, Linien, Quadrate und Freihand-Einzeichnungen. Der Auswerter kann außerdem Geodaten hinzufügen, wenn diese nicht im Bild enthalten sind. Aus diesen Daten kann zum Beispiel ein Kompass oder ein Maßstab errechnet und eingezeichnet werden. Anschließend lässt sich das annotierte Bild mit den Einzeichnungen exportieren. Insbesondere das Anzeigen von stereoskopische Bildern gibt dem Bildauswerter einige Vorteile. So lässt sich zum Beispiel die Höhe eines Gebäudes wesentlich einfacher erkennen. Hierfür eignet sich der mittlere 3D Bildschirm besonders gut.

1.1.2. Geoviewer

Der Geoviewer ist ein Werkzeug zur Kartendarstellung. Hier können interaktiv Geodaten visualisiert werden, um eine bessere Analyse der Bilder zu erzielen (siehe [2]). Diese

Abbildung 1.1.: Der Bildauswerteplatz der Zukunft

Daten stammen aus diversen zivilen und militärischen Quellen und können auf der Karte angezeigt werden. Ein Anwendungsfall kann das Anzeigen von Tracks eines Schiffes oder Fahrzeuges sein. Außerdem ermöglicht die Software das Einzeichnen von Formen und Symbolen auf der Karte, die mit dem Backend synchronisiert werden und damit mit allen anderen verbundenen Anzeigesystemen, wie z.B. dem digitalen Lagetisch. Die Software unterstützt die Benutzung durch Touch Eingaben. Das wird am Bildauswerteplatz der Zukunft durch ein TouchDisplay ausgenutzt.

1.1.3. RecceMan®

RecceMan® ist eine Objekt-Identifikations-Software, die den Bildauswerter unterstützt, Objekte zu erkennen (siehe [3]). Die Aufgabe eines Bildauswerters ist die exakte Erkennung von Objekten anhand von charakteristischen Bildmerkmalen. Die speziell ausgebildeten Bildauswerter nutzen hierfür das dokumentierte Wissen aus Handbüchern. Die Erkennungsassistenz ermöglicht dem Bildauswerter, das Objekt durch Merkmale zu beschreiben. Anhand dieser Merkmale werden alle mögliche Kandidaten angezeigt. Für jeden Kandidat steht ein detaillierter Steckbrief zur Verfügung.

1.1.4. Weitere Funktonen

Ergänzend zu dem bereits genannten Programmen wird noch ein Webbrowser zum Anzeigen von Metadaten verwendet. Zu jedem ausgewerteten Bild muss der Bildauswerter einen Bericht anfertigen. Dafür wird häufig Standardsoftware verwendet, wie zum Beispiel Microsoft Word.

Der Bildauswerteplatz der Zukunft verfügt außerdem über zwei Kinect-Kameras, die oberhalb der mittleren Bildschirme angebracht sind. Mit ihnen wird die Bedienung eines Arbeitsplatzes mit vier Bildschirmen erleichtert. Bei der herkömmlichen Bedienung mit Maus

Abbildung 1.2.: Der Stereo-Bildauswerter. In das Bild wurden Einzeichnungen eingefügt.

und Tastatur muss die Maus eine große Distanz zurücklegen, sobald man zwischen den Bildschirmen wechselt. Die eine Kinect-Kamera erfasst die Kopfdrehung des Betrachters und die zweite erkennt, wenn der Nutzer mit dem Finger auf einen Bildschirm zeigt. Mit dieser Technologie ergeben sich neue Möglichkeiten der Interaktion mit dem System (siehe [4]).

1.2. Coalition Shared Data Server

Coalition Shared Data Server [5] ist der Speicherort von Aufklärungsergebnissen. Der Server implementiert den STANAG 4559 Standard und dient zur Speicherung von standardisierten Daten, wie zum Beispiel Videos, Bildern, Berichten und Pfaden. Auf diese Daten kann durch verschiedene schreibende und lesende Clients zugegriffen werden. Der Zugriff erfolgt über Corba und die Antwort ist ein Metadaten-XML-File, in dem alle Referenzinformationen enthalten sind. In dem XML-Dokument ist der Download-Link oder -Stream auf die eigentlichen Daten angegeben. Es ist möglich, Assoziationen zwischen den Daten zu erstellen, um Abhängigkeiten darzustellen.

2. Design

2.1. Motivation

Ein Ziel dieser Arbeit ist es, die bestehende Anbindung an die CSD des CSDAdapter wiederverwendbar für andere Projekte zu machen und zu aktualisieren. Außerdem soll auch der SBA an die CSD angebunden werden. Aktuell muss der Bildauswerter jedes Bild einzeln aus dem Dateisystem laden, was ineffizient und langwierig ist. Der Bildauswerter soll die Daten direkt aus der CSD laden können und seine ausgewerteten Ergebnisse dort abspeichern. Eine weitere Anforderung an den SBA ist, dass dieser auch ohne die Verbindung zur Middleware bzw. dem Backend funktionieren soll. Alle CSD Anfragen des SBA im CSD Adapter zu bearbeiten ist dementsprechend nicht erwünscht.

2.2. Übersicht über die bestehende Struktur

Der Bildauswerteplatz der Zukunft besteht aus mehreren Komponenten (siehe Abbildung 2.1). Die zentrale Komponente ist das Backend. Dieses speichert alle Informationen wie die Data Objekte, die auf dem Geoviewer visualisiert werden. Die Kommunikation zwischen dem Backend und anderen verbundenen Komponenten läuft über die Middleware. Der Geoviewer bezieht das Kartenmaterial von einem Geoserver. Hierfür werden unter anderem Web Map Service (WMS) und Web Feature Service (WFS) verwendet. Durch die Auswahl eines Data Objektes im Geoviewer können Metadaten zu diesem Objekt abgefragt werden. Diese Daten werden über die Metadaten Anzeige im Webbrowser dargestellt. Der Stereo Bildauswerter ist ebenfalls an das Backend angeschlossen. Bisher wird diese Verbindung beim Laden eines Bildes genutzt. Falls das Bild georeferenziert ist, sendet der SBA die Koordinaten an den Geoviewer, der dann zur Umgebung des Bildes zoomt. Der CSDAdapter implementiert eine Verbindung zur CSD. Mit Hilfe der Isaac.lib (Siehe [5, Clients für Datenzugriff]) wird auf den Server zugegriffen und es werden Datenobjekte für das Backend erstellt. Der Geoviewer beschränkt sich beim Laden aus der CSD auf einige wichtige Datentypen, insbesondere Bilder, Videos und Berichte. Diese werden an der richtigen Stelle auf die Karte angezeigt, sofern sie georeferenziert sind. Die Bilder aus der CSD werden heruntergeladen und können bei Bedarf als zusätzliches Kartenmaterial in den Geoserver integriert werden. Luftbilder von Aufklärungsflügen können so als aktuelles Kartenmaterial eingebunden werden. Damit eignen sich diese Datenobjekte nicht zum Laden und Verarbeiten im Stereo Bildauswerter.

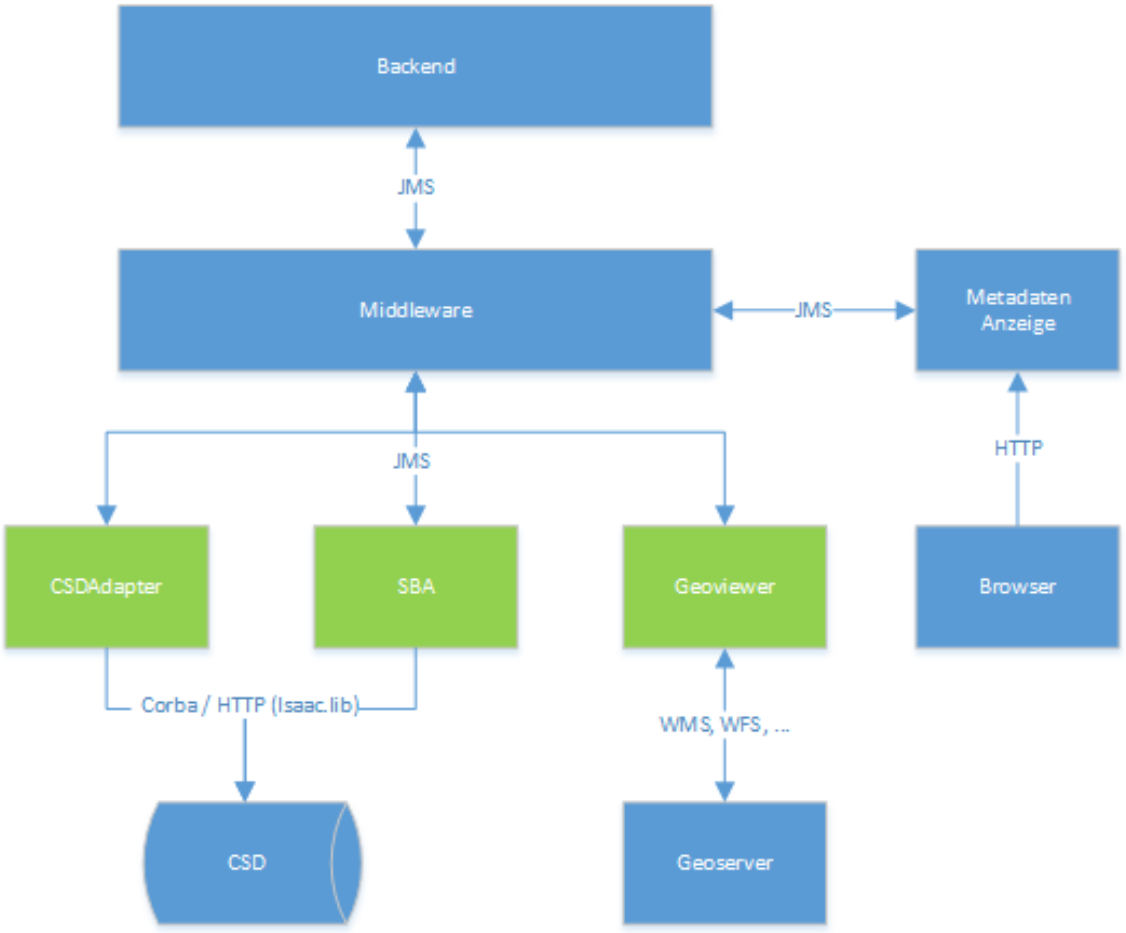

Abbildung 2.1.: Die Struktur der Komponenten des Bildauswerteplatzes. Die Kommunikation zwischen den Komponenten erfolgt hauptsächlich über den Java Message Service (JMS). In den grün eingefärbten Komponenten wurden während dieser Arbeit Veränderungen vorgenommen

2.3. Laden von Daten

Der erste Arbeitsschritt eines Bildauswerters ist das Laden der benötigten Daten, insbesondere das zur Auswertung vorgesehene Bild und die dazugehörigen Informationen aus der Umgebung. Bisher war das Importieren von Bildern nur aus dem Dateisystem möglich. Das Bild wird häufig durch den CSD-Server zur Verfügung gestellt und soll deshalb leicht in den SBA geladen werden können. Eine Schwierigkeit dabei ist, dass in der CSD nur Bilder im NSIF Format abgespeichert und bereitgestellt werden. Das NSIF Bildformat entspricht dem STANAG 4559 Standard und enthält neben dem Bild weitere Informationen. Der Ablauf einer CSD Abfrage soll auf dem SBA und dem Geoviewer verläuft gleich. Der Nutzer erstellt eine Anfrage für einen CSDTask. Dieser Task wird dann ausgeführt und die Ergebnisse für die jeweiligen Anwendungsfälle verarbeitet.

2.3.1. CSD Task erstellen

Um den Bildauswertern die Arbeit zu vereinfachen, soll zum Auslesen der CSD in allen Programmen dieselbe Nutzeroberfläche zur Verfügung stehen (siehe Abbildung 2.2). Die Benutzeroberfläche ermöglicht das Suchen nach vielen Kriterien, wie z.B. den Namen der Mission, des Erstellers sowie der Erdstellungszeit. Die Eingabeoberfläche war im

CSD-Plugin des Geoviewers vorhanden, musste aber angepasst werden, damit sie wiederverwendbar wird. Um das zu gewährleisten, wurde die Nutzeroberfläche außerdem in die CSDCommon Bibliothek verschoben.

Eine wichtige Funktion zur Einschränkung der Suchergebnisse ist das Auswählen einer Region. Der Geoviewer bietet hierfür eine einfache Möglichkeit. Der Nutzer kann durch eine Einzeichnung auf der Karte den Suchbereich selektieren und damit die Suche einschränken, um nur die für ihn interessanten Ergebnisse zu erhalten. Da die Verarbeitung vieler Ergebnisse merklich Zeit verbraucht, führt die Einschränkung der Region auch zu einer flüssigeren Bedienbarkeit. Auf dem SBA gibt es keine einfache Möglichkeit einen Bereich zu selektieren. Um eine alternative einfache Eingabe zur Verfügung zu stellen, wurde die Verbindung vom SBA zum Geoviewer um eine Regionsauswahl erweitert. Ist der SBA mit dem Geoviewer über das Backend verbunden und der Geoviewer gestartet, so aktiviert sich die Regionsauswahl für den Bildauswerter. Die Auswahl der Region für den SBA erfolgt auf dem Geoviewer und das Ergebnis wird an den SBA gesendet. In Abbildung 2.5 ist diese Verbindung über das Backend an das CSD-Plugin eingezeichnet.

Die Nutzeroberfläche verfügt über einige Comboboxen, die eine einfache Auswahl von Feldern ermöglichen. Diese Boxen sind durch ein Preset konfigurierbar, das einfach aus einer Konfigurationsdatei erstellt werden kann. Für den Nutzer hat das den Vorteil, dass er nicht alle Informationen von Hand eingeben muss. Das ist z.B. für die URLs der CSD-Server sinnvoll.

In den Abbildungen 2.4 und 2.5 ist diese Oberfläche als RequestUI in grün eingezeichnet. Das Laden des Presets erfolgt auf dem SBA direkt und beim Geoviewer wird dieses vom CSDADapter geladen.

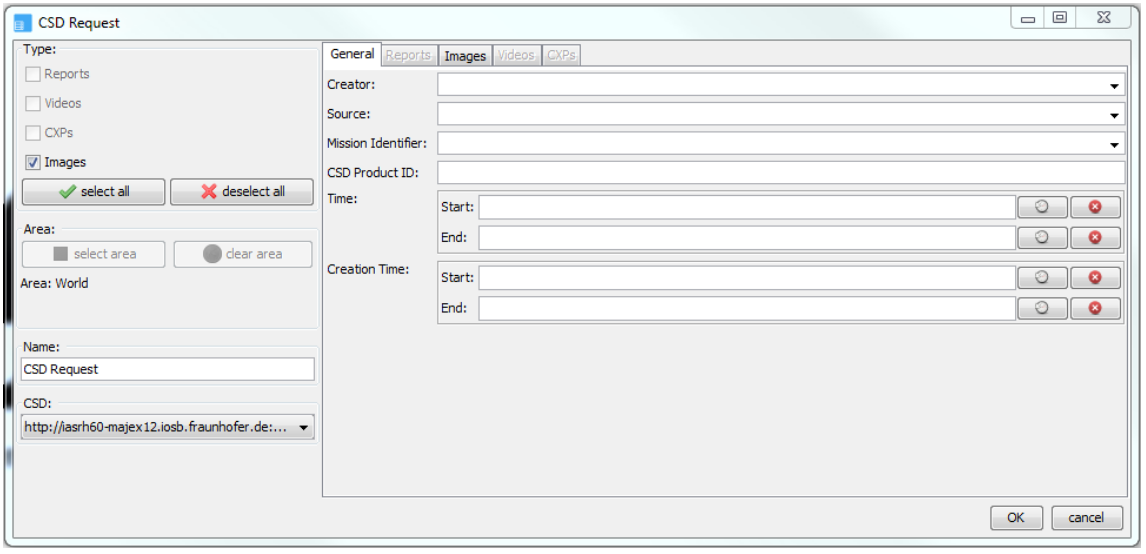

Abbildung 2.2.: Die Eingabe eines CSD-Requests im SBA. Der Request ist auf Bilder eingeschränkt, da der SBA nur Bilder unterstützt.

2.3.2. Task ausführen

Nachdem der Nutzer einen Task erstellt hat, muss dieser ausgeführt werden. Die Eingaben des Nutzers müssen in eine CSD-Query umgewandelt und die Verbindung zum Server hergestellt werden.

Der Geoviewer sendet die Task zur Verarbeitung an den CSD-Adapter. Der SBA verarbeitet den Task direkt. Bei beiden Programmen wird zunächst der Task gestartet und

anschließend mit dem CSDReadingClient der Isaac.lib auf den Server zugegriffen. In den Abbildungen 2.4 und 2.5 ist dieser Teil grün markiert. Die Verarbeitung eines Tasks und der Zugriff auf den CSD-Server ist in beiden Komponenten ähnlich und wurde deshalb in die CSDCommons-Bibliothek ausgelagert.

2.3.3. Ergebnis auswerten

Der Isaac.lib kann bei einer Abfrage ein Result Handler übergeben werden. Gibt es zu einer Abfrage ein Ergebnis, wird dieses wiederum dem Result Handler übergeben. Ein Ergebnis besteht aus einem MetaDaten XML Dokument. Für den SBA und den CSD Adapter müssen zunächst alle generellen Informationen aus diesem XML Dokument herausgelesen werden. Anschließend folgt die typspezifische Verarbeitung.

Beim SBA wird anschließend ein CSD-Data-Objekt mit allen nötigen Informationen erstellt und dieses Objekt dem CSD-DataStore hinzugefügt. Das Ergebnis der Anfrage wird anschließend in einer Tabelle dargestellt und der Nutzer kann daraus ein Bild zum Laden auswählen. In Abbildung 2.5 ist der CSD Data Sore und der SBAResultHandler eingezeichnet, der die Ergebnisverarbeitung übernimmt. Die ResultUI ist die Oberfläche zur Auswahl eines Ergebnisses. Ein Screenshot der UI ist in Abbildung 2.6 zu sehen.

Der CSD-Adapter trennt die Ergebnisse erst nach ihrem Typ, sodass zum Beispiel Bilder und Videos anders verarbeitet werden. In Abbildung 2.4 ist die Aufspaltung der Verarbeitung durch den MainResultHandler und mehrere spezielle ResultHandler dargestellt. Im den meisten Fällen wird zuerst das eigentliche Objekt heruntergeladen und für den Geoviwer aufbereitet. Bei Bildern wird nach dem Herunterladen geprüft, ob das Bild direkt in das Kartenmaterial eingebunden werden kann (Abbildung 2.3). Aus dem Metadaten-XML-File wird eine Metadaten-Anzeige erstellt. Außerdem wird ein Data Objekt erstellt und an das Backend gesendet, sodass das Ergebnis auf der Karte visualisiert werden kann. Im Geoviewer werden diese Objekt dargestellt und können zur Anzeige der Metadaten im Metadaten-Display verwendet werden.

Abbildung 2.3.: Der Geoviewer. Auf der Karte werden Luftbilder aus der CSD angezeigt.

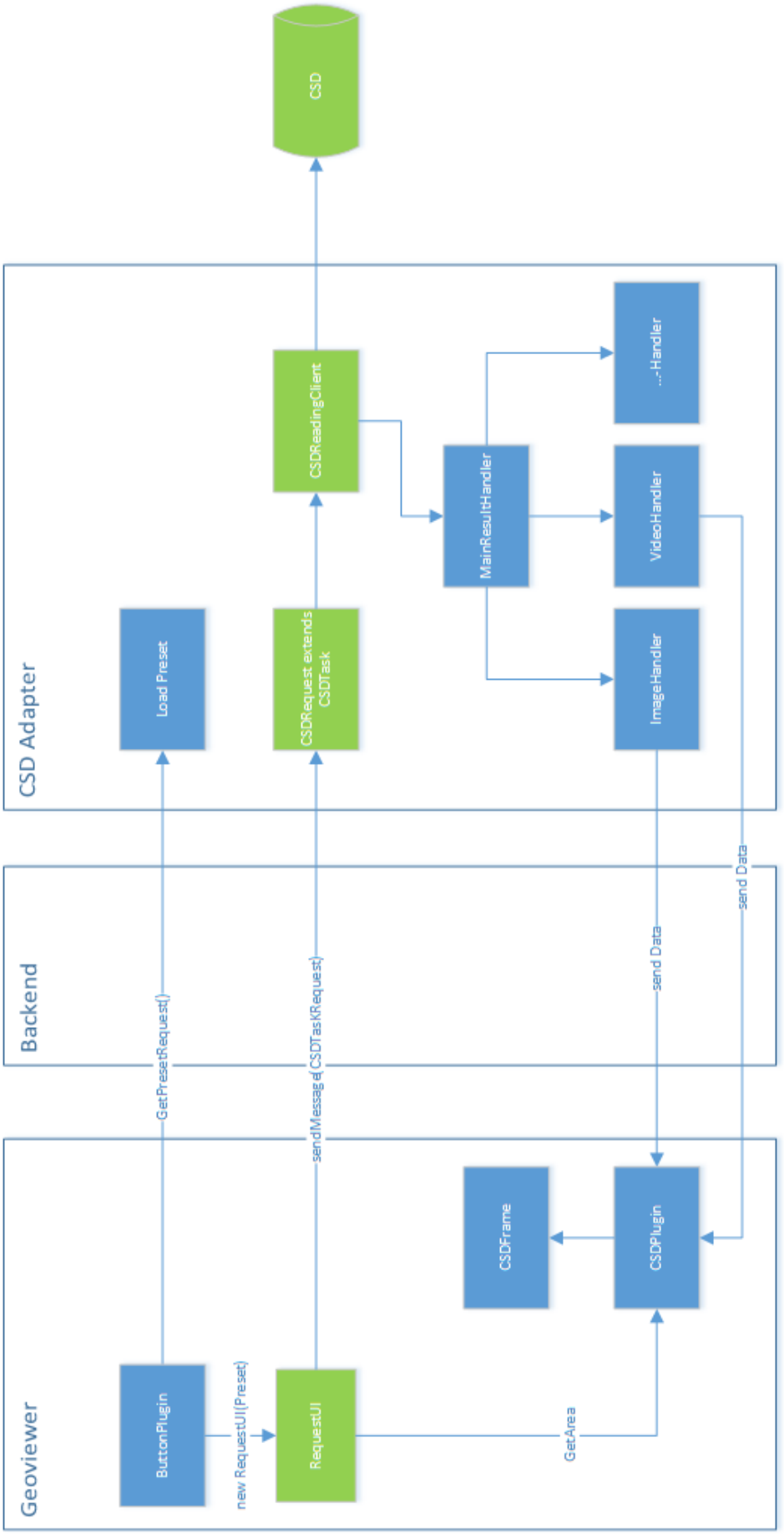

Abbildung 2.4.: Struktur der Anbindung an die CSD für den Geoviewer.

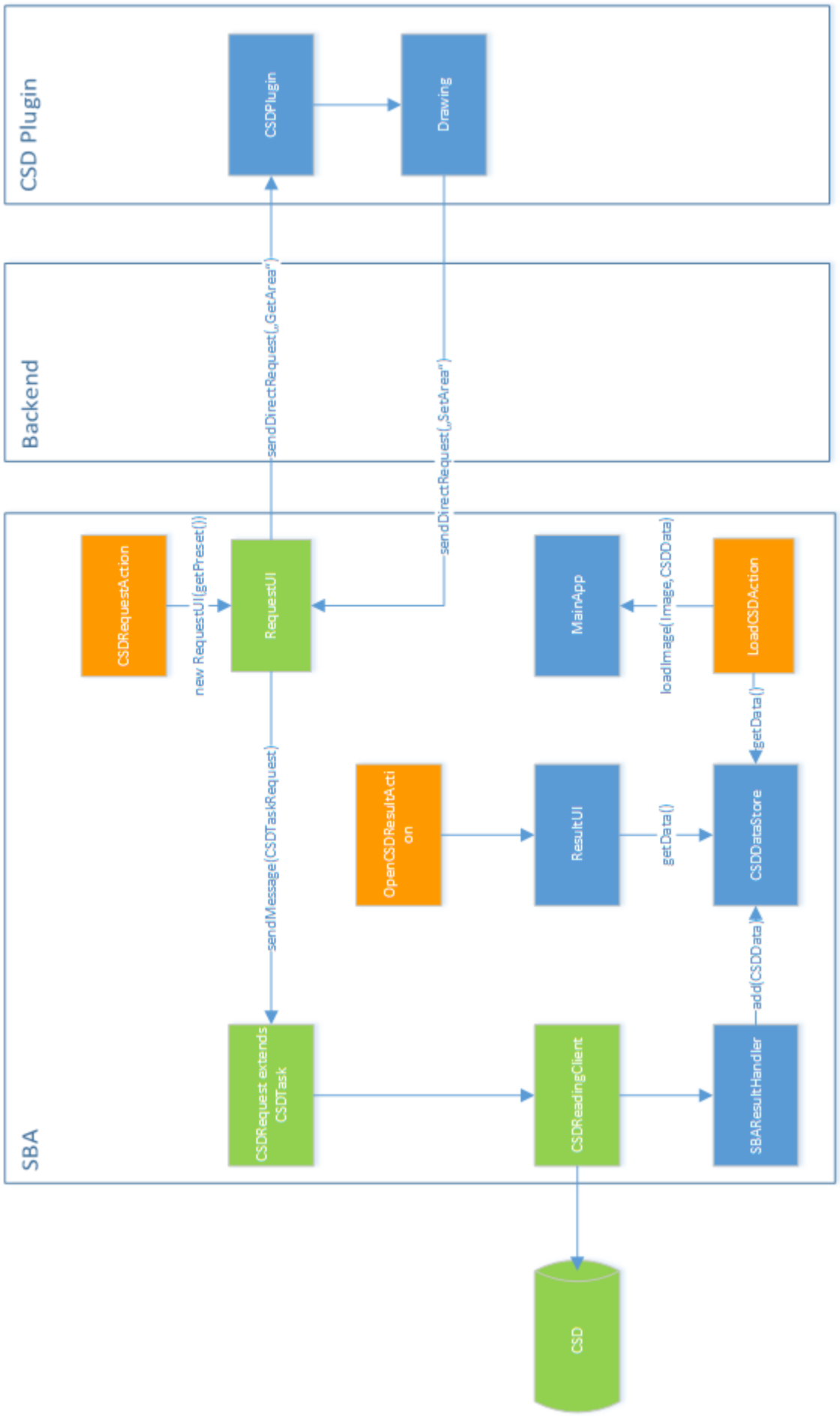

Abbildung 2.5.: Struktur der Anbindung an die CSD im Stereo-Bildauswerter. In gelb sind die Swing-Actions eingezeichnet, die der Nutzer aufrufen kann.

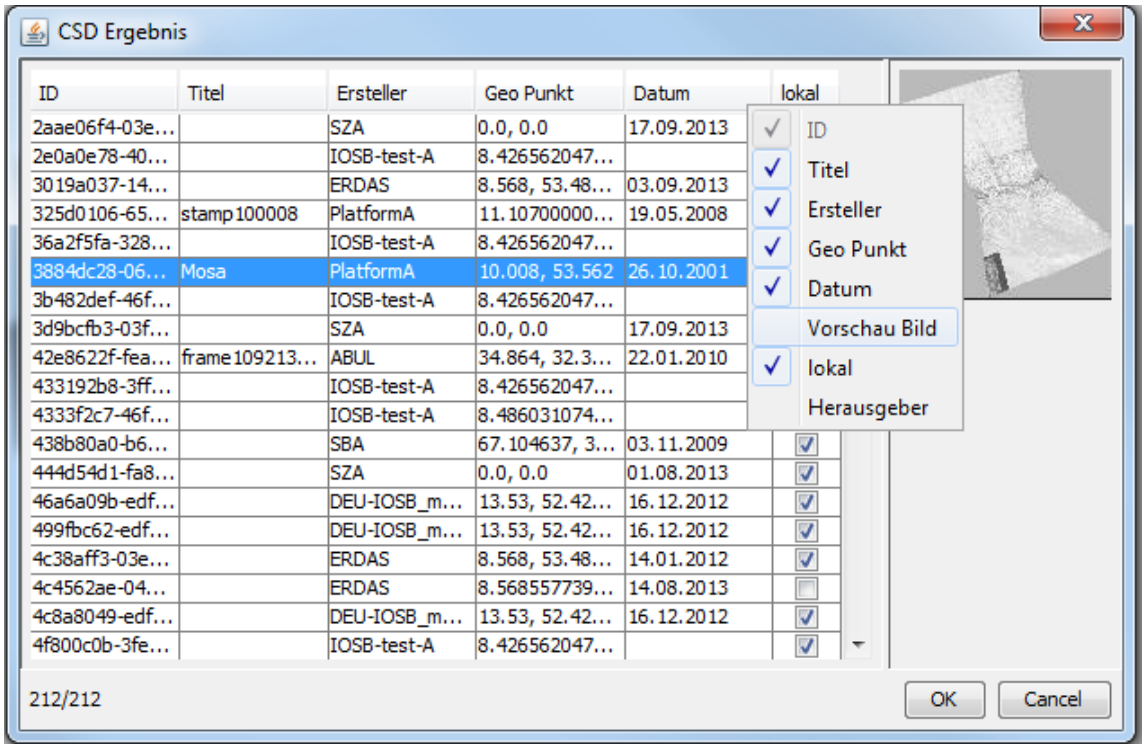

Abbildung 2.6.: Die Anzeige aller Ergebnisse im SBA. Die Tabelle ist sortierbar und beliebige Spalten lassen sich ein- und ausblenden. Falls ein Vorschaubild verfügbar ist, wird dieses an der Seite zusammen mit dem Kommentar des Bildes angezeigt.

2.4. Ausgewertete Daten des SBA speichern

2.4.1. Export in RecceMan®

Eine Funktion der Erkennungsassistenz RecceMan® ist das Hinzufügen von bisher unbekannten Fahrzeugen. Sind zum Beispiel Piraten mit selbstgebauten Booten unterwegs, sind diese höchstwahrscheinlich nicht im System. RecceMan® unterstützt das Hinzufügen von neuen Objekten und Bildern. In dem SBA wurde eine Funktion integriert, die das Exportieren eines Bildes für den RecceMan® vereinfacht. Durch einen Button wird der Rendervorgang des Bildes gestartet und in einen definierten Ordner abgespeichert. Diesen Ordner durchsucht RecceMan® regelmäßig und bietet einen Import für neue Objekte an (Abbildung 2.7);

2.4.2. Export in CSD

Nach einer erfolgreichen Auswertung soll das annotierte Bild wieder zu den CSD hinzugefügt werden. Hierfür wurde in den SBA ein schreibender Zugriff auf den Server implementiert. Zunächst kann der Nutzer die nötigen und bei Bedarf auch einige optionale Meta-Informationen eingeben. Hierfür wurde eine Oberfläche angelegt (siehe Abbildung 2.8). Falls das Bild aus der CSD geladen wurde, so wird das UI mit den Werten des geladenen Bildes gefüllt. Die Oberfläche überprüft, ob alle zwingenden Richtlinien eingehalten wurden. Das kann zum Beispiel die Prüfung von Pflichteingaben oder der maximalen Zeichenzahl für bestimmte Felder bedeuten. Buttons zur schnellen Datumsauswahl und zum Generieren von zufälligen IDs wurden hinzugefügt. Nachdem der Nutzer alle Informationen eingetragen hat, beginnt der eigentliche Schreibvorgang. Zunächst wird das Bild gerendert

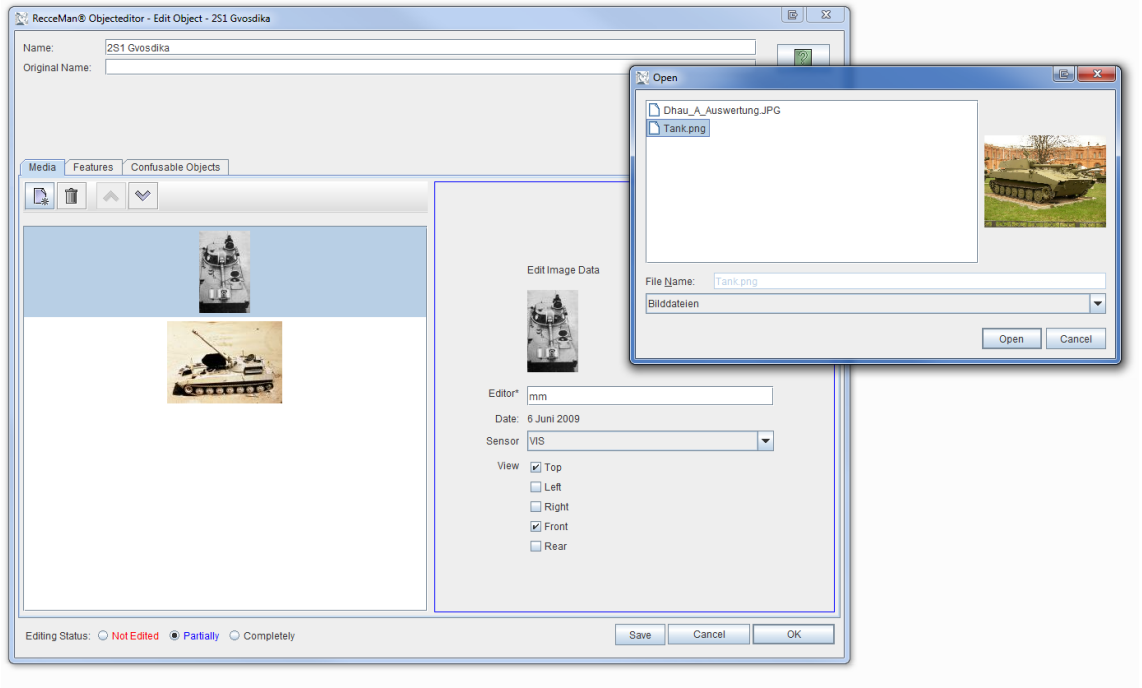

Abbildung 2.7.: Das importieren eines Bidles in den RecceMan®

und mit Hilfe des NSIF-Creators in ein NSIF-Bild umgewandelt. Der NSIF-Creator ist eine Bibliothek, die aus diversen herkömmlichen Bildformaten und einem Metadaten XML-File ein NSIF-Bild erstellt. Das Metadaten-XML-File wird mit Hilfe des CSD-Writing Clients der Isaac.lib an die CSD gesendet. Das Bild wird in einem HTTP Server, der in der Isaac.lib enthalten ist, zur Verfügung gestellt, sodass der CSD Server das Bild herunterladen kann.

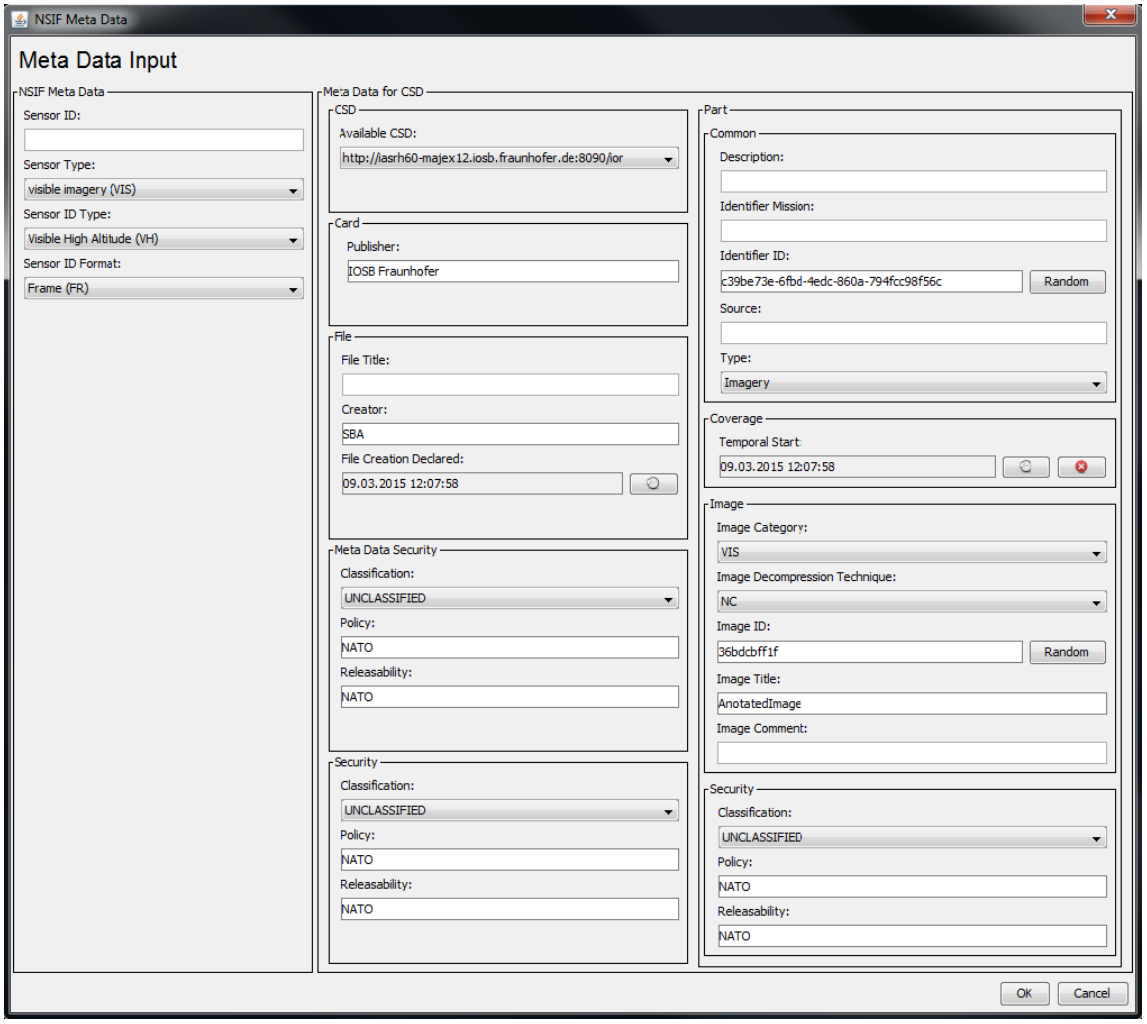

Abbildung 2.8.: Die Nutzeroberfläche zum Eingeben der Metadaten. Links die Informationen zum Erzeugen eines NSIF-Bildes und rechts die Informationen für die CSD.

3. Implementation

3.1. Laden von Daten

Als Erstes betrachten wir das Laden von Bilden im SBA. Bisher war im SBA nur das Laden von der Festplatte verfügbar. Im Folgenden wird die Anbindung an die CSD beschreiben, die in den SBA integriert wurde.

3.1.1. CSD-Task erstellen

Die Request UI des CSDPlugin soll im SBA wiederverwendet werden. Hierfür wurde eine gemeinsame Schnittstelle angelegt (siehe Abbildung 3.1). Diese Schnittstelle wird aufgerufen, sobald der Nutzer einen Task absendet und bei der Auswahl einer Region. Das CSD Plugin des Geoviewers implementiert diese Schnittstelle. Region Anfragen werden direkt bearbeitet und der Task Request wird an den CSD-Adapter gesendet.

Im SBA wird das Interface der CSDRequestAction implementiert. Der Aufruf der Region wird falls möglich an das CSD-Plugin weitergeleitet. Die Middleware verfügt neben Nachrichten und Anfragen an das Backend über DirectRequests zwischen den an die Middleware angeschlossenen Komponenten. Jede verbundene Komponente verfügt über eine eindeutige ID. Durch diese ID erfolgt die Zuordnung des DirectRequests zum jeweiligen Empfänger und den dieser mit einer DirectResponse beantwortet. Der SBA nutzt diese DirectRequests, um eine Region auf dem Geoviewer auswählen zu lassen. Das Problem hierbei ist, dass der Nutzer beliebig lange zum Einzeichnen brauchen kann und die Antwort eines DirectRequest innerhalb des Timeouts erfolgen muss. Deshalb antwortet der Geoviewer sofort auf den Request und lässt den Nutzer Einzeichnungen vornehmen. Ist die Regionsauswahl abgeschlossen, sendet der Geoviewer einen DirectRequest an den SBA, um das Ergebnis der Auswahl zu übermitteln. Im Sequenzdiagramm 3.2 sind beide Abläufe dargestellt.

3.1.2. Task ausführen

Die Isaac.lib unterstützt 2 Arten von Anfragen an die CSD. Ein CSD-Request ist eine einmalige Abfrage des Servers, die alle Treffer an einen im Request definierten ResultHandler übergibt und sich dann beendet. Eine Subskription liefert auch nach dem Abfragezeitpunkt Ergebnisse. Wird zum Beispiel ein neuer Datensatz hinzugefügt, der auf die Abfrage passt, so wird dieser auch an den ResultHandler übergeben. Die Verarbeitung der unterschiedlichen Abfragen unterscheidet sich nur in dem Aufruf des CSDReadingClients. Deshalb

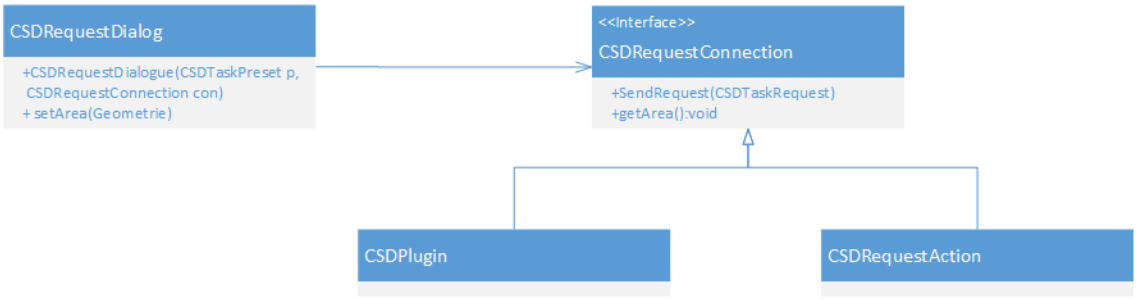

Abbildung 3.1.: Das Generieren von Tasks hat eine einheitliche Schnittstelle. Diese wird im SBA und CSDPlugin implementiert. So kann der RequestDialog wiederverwendet werden.

wurde die abstrakte Klasse CSDTask angelegt (siehe Abbildung 3.3). Die Methode startRequest startet die Abfrage in einen neuen Thread. Zuerst wird die abstrakte Methode beforeTask aufgerufen, um Vorbereitungen zu treffen, zum Beispiel Ergebnisse von einem alten Task zu löschen.

Die Ausführung der CSD-Abfrage erfolgt in 3 Schritten. Erst wird der CSD Querry String generiert (Siehe Querry 1). Anschließend folgt eine Abfrage über die Anzahl der erwarteten Ergebnisse. Die Methode isValidTaskSize überprüft nun, ob der Task ausgeführt werden soll und ob das Ergebnis zu groß und somit nicht sinnvoll ist. Die Standard-Implementierung vergleicht die erwarteten Ergebnisse mit einem Wert aus der Konfigurationsdatei. Ist der Task "klein genug", wird die abstrakte Methode queryCSD aufgerufen. Die Methode wird implementiert von den beiden Klassen CSDRequst und CSDSubscription, die vom CSDTask erben. In dieser Methode wird die Verbindung zur CSD aufgebaut und durchsucht. Die Ergebnisse der Abfrage werden alle an den übergebenden Result Handler weitergeleitet.

Um das System auch ohne CSD beim Kunden zu präsentieren, wurde die Möglichkeit eines lokalen Request aus dem CSD-Adapter übernommen. Dieser lokale Request lässt sich im Konfigurationsdatei aktivieren und führt dazu, dass die Ergebnisse aus einer Datei geladen und direkt an den ResultHandler übergeben werden.

```
((NSIL_COMMON.type = 'IMAGERY'))
AND NSIL_COVERAGE.spatialGeographicReferenceBox inside POLYGON
(49.0578,11.3417,49.0526,11.3407,49.0522,11.3543,
 49.0584,11.3553,49.0584,11.3553,49.0578,11.3417)
```

Querry 1: Beispiel eines CSD Querry Strings nach Bildern innerhalb einer geografischen Region. Die Koordinaten wurden aus Übersichtsgründen auf vier Nachkommastellen gekürzt

3.1.3. ResultHandling

Die Ergebnisse aus einer CSD-Abfrage werden als Liste aus XML-Dokumenten an die Methode newResults(List<Documents>) übergeben. Ein Beispiel XML Dokument ist im Anhang A. Diese ist ein Teil des Interfaces Isaac.lib ResultHandler. Für den SBA und den CSD-Adapter müssen zunächst einige allgemeine Informationen aus dem XML-Dokument ausgelesen werden. Deshalb wurde die abstrakte Klasse ResultHandler in der CSDCommon-Bibliothek angelegt (siehe Abbildung 3.4). Für jedes neue Ergebnis werden zunächst alle

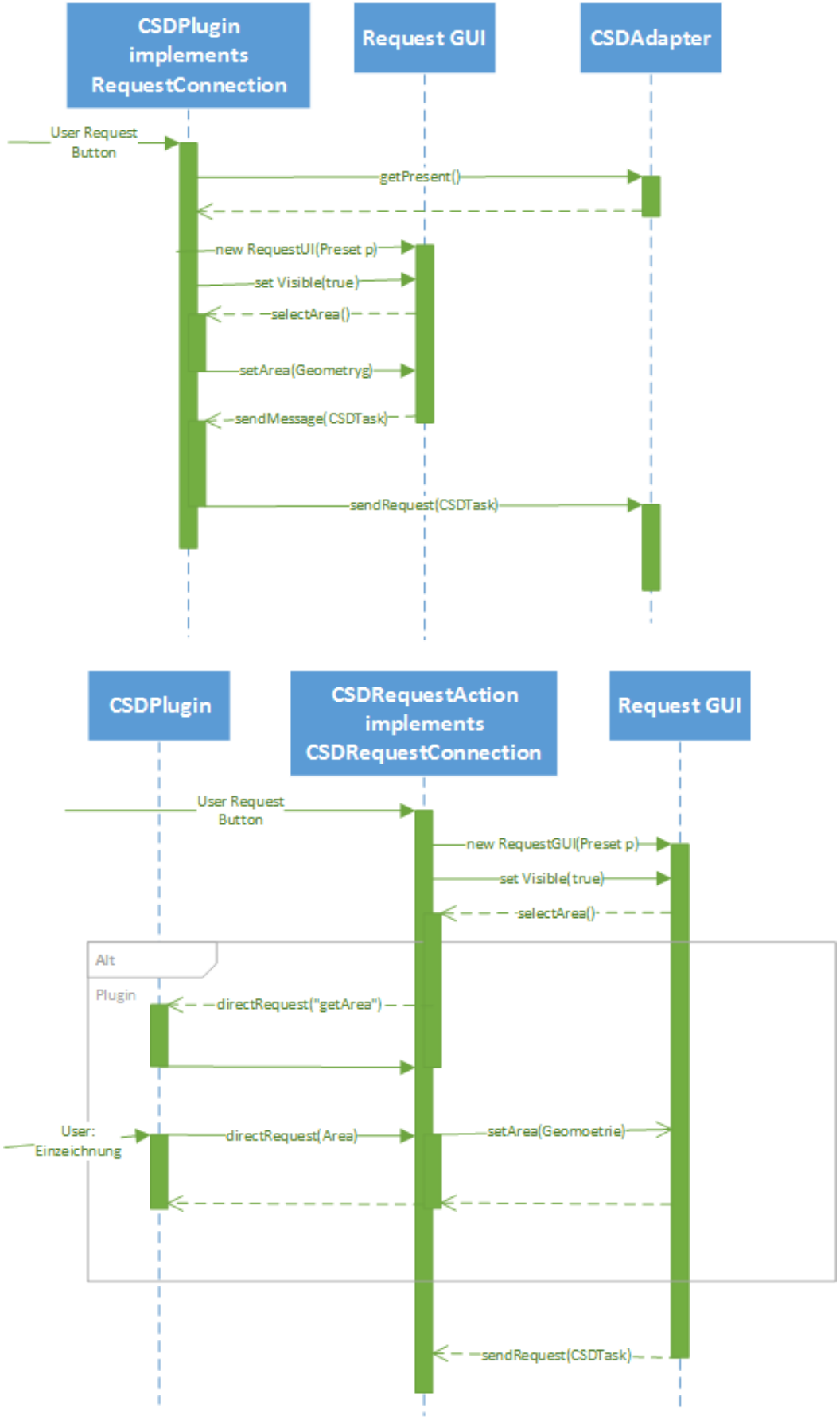

Abbildung 3.2.: Das Erstellen eines CSD Tasks als Sequenz Diagramm. Oben der Ablauf im CSDPlugin des Geoviewer; unten im SBA. Der alternative Block (Alt) steht nur zur Verfügung wenn der SBA mit dem Geoviewer verbunden ist.

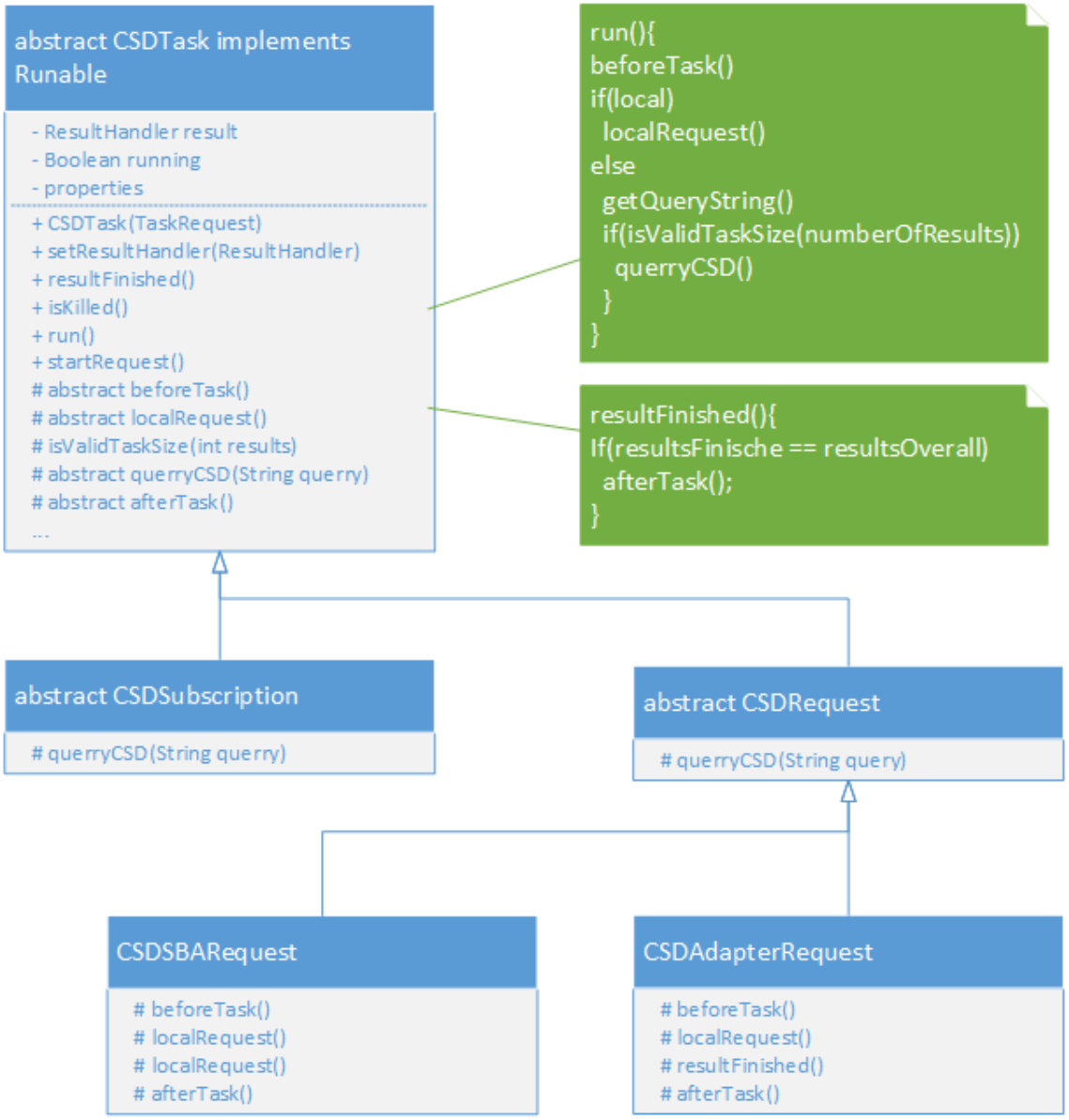

Abbildung 3.3.: Vererbungs-Hierachie des CSDTasks. Die Ausführung des Task wird in der Superklasse gestartet. Die jeweilige CSD Anfragen sind im CSDRequest und die CSDSubscription implementiert. Der SBA Request und der Adapter Request implementieren die Vor- und Nachbereitung des Tasks sowie das Userfeedback über den Fortschritt in den jeweiligen Programmen.

allgemeinen Felder ausgelesen und anschließend die abstrakte Methode newResult aufgerufen. Diese Methode wird von den spezifischen ResultHandlern implementiert, um ein Ergebnis zu verarbeiten. Außerdem werden alle Java Exceptions abgefangen, die bei der Auswertung eines Ergebnisses geworfen werden. Bricht die Verarbeitung eines Ergebnisses mit einer Exception ab, so wird die Methode storeErrorProduct aufgerufen. In der Standard-Implementierung wird das XML Dokument dessen Verarbeitung fehlgeschlagen ist, in einem Ordner abgelegt. Falls die Verarbeitung fehlschlägt, kann der Nutzer dieses XML Dokument einsehen um totzdem an die Informationen zu kommen.

Ist ein Ergebnis verarbeitet, wird im zugehörigen Task die Methode resultFinished aufgerufen. So kann der Task, nachdem alle Ergebnisse ausgewertet sind, die Methode afterTask aufrufen, um eine eventuelle Nachbearbeitung durchzuführen (Abbildung 3.3).

Abbildung 3.4.: Vererbungs-Hierachie der Result Handler

3.1.4. CSD-Adapter

Im CSD-Adapter wird ein CSDRequest genutzt. In beforeTask und resultFinished werden Nachrichten an den Goeviewer gesendet, um ein visuelles Feedback an den Nutzer zu geben, wie weit die Bearbeitung des Tasks vorangeschritten ist (siehe Abbildung 3.5)

Abbildung 3.5.: Links die Fortschrittsanzeige im Geoviewer, Rechts im SBA

Zur Kommunikation implementiert der CSD-Adapter einen ConnectionServer und der Geoviwer einen ConnectionManager aus den MiddlewareTools. Das heißt, die Kommunikation über neue Tasks und den Fortschritt läuft nicht über das Backend, sondern direkt zwischen den Programmen. Die erstellten Datenobjekte werden an das Backend gesendet, sodass diese im System sind und alle angeschlossenen Programme die Daten darstellen können.

Die Verarbeitung der Ergebnisse im ResultHandler wird zunächst aufgeteilt. Der MainResultHandler erhält alle Ergebnisse und gibt diese an einen auf den jeweiligen Datentyp spezialisierten ResultHandler weiter. Sowohl der MainResultHandler als auch die spezifischen Handler erben von der allgemeinen Implementierung in der CSDCommons Bibliothek. Im MainResultHandler wird die Methode newResults überschrieben, um die Ergebnisse weiterzureichen.

3.1.5. SBA

Im SBA ist die Auswahl-Oberfläche auf Bilder eingeschränkt. Andere Datentypen können nicht verarbeitet werden und sind deshalb auch nicht abfragbar. Bei der Verarbeitung der Bilder werden alle wichtigen Informationen in einem CSDData-Objekt gespeichert. Falls ein Vorschaubild vorhanden ist, wird dieses heruntergeladen. Auch Vorschaubilder liegen in der CSD als NSIF-Bilder vor und werden deshalb in PNG-Bilder umgewandelt. Alle Ergebnisse werden dem CSDDataStore hinzugefügt und der Nutzer kann sich in der ResultUI ein Bild aussuchen. Sobald der Nutzer ein Bild aus der CSD laden möchte, wird dieses heruntergeladen und aus dem NSIF-Format in den SBA importiert. Um das Bild zu importieren wird es zunächst in das PNG-Format umgewandelt.

3.2. Bilder exportieren

Das Exportieren von Daten auf das Dateisystem war im SBA vorhanden. Wie alle Funktionen im SBA wurde auch das Exportieren in einer Swing Action implementiert, die an mehrere Nutzereingaben wie z.B. Buttons verknüpft werden kann. Diese ExportAction fragt den Nutzer nach einem Speicherort, rendert das Bild und legt es als Datei an den vom Nutzer ausgewählten Ort.

3.2.1. Export in RecceMan®

Um den Export in den RecceMan® zu gestalten, wurde das Rendern der ExportAction umgebaut, so dass es wiederverwendbar ist. Die ExportAction in RecceMan® erbt von der Standard ExportAction, sodass die Hauptfunktionalität, das Rendern übernommen wird. Um das Bild dem RecceMan® zur Verfügung zu stellen, muss dieses in einem vordefinierten Ordner abgelegt werden. Dieser Ordnerpfad wird aus der Konfigurationsdatei gelesen. Der Nutzer muss nicht jedes Mal manuell den Pfad heraussuchen, sondern das Bild wird direkt an den richtigen Ort gelegt.

3.2.2. Export in CSD

Der Export in die CSD erbt auch von der ExportAction. Zunächst muss der Nutzer alle nötigen Informationen zum Speichern in die CSD eingeben. Die Informationen werden auf Richtigkeit und Notwendigkeit überprüft. Wurde das editierte Bild aus der CSD geladen, so wird die Oberfläche mit den Informationen aus dem geladenen XML-File ausgefüllt. Stammt das Bild nicht aus der CSD, werden falls möglich Standardwerte gesetzt, die in einer Konfigurationsdatei editierbar sind. Für das Einlesen und generieren der XML Files ist die XMLGenerator zuständig. Die Nutzereingaben erfolgen in der MetaDataUI (siehe Abbildung 3.6). Nachdem der Nutzer die Informationen editiert hat, wird das temporäre Bild mit dem NSIF-Creator zu einem NSIF-Bild konvertiert. Die nötigen Metadaten werden aus der Nutzeroberfläche geladen und in ein XML-Dokument geschrieben. Sind im SBA Geoinformationen vorhanden, werden diese ebenfalls angefügt. Das XML-Dokument für die CSD wird auch mit den Informationen aus der Nutzereingabe generiert und zusammen mit dem NSIF-Bild an die Isaac.lib übergeben. Ist das Bild aus der CSD geladen, wird darauf geachtet, dass alle Informationen aus dem alten XML-Dokument übernommen werden. Die Eingabe umfasst nicht alle Felder, die in dem XML-Dokument vorhanden sein können. Bei der Erstellung des neuen Dokuments wird in diesem Fall das alte kopiert, alle Felder, die vom Server generiert werden, gelöscht und alle Änderungen aus der Nutzereingabe eingetragen. Sind keine Informationen vom vorherigen Bild vorhanden, wird ein neues XML-Dokument generiert (Beispiel im Anhang B). Das NSIF-Bild wird nicht direkt übertragen, sondern wird in einem HTTP Server der Isaac.lib zum Download bereitgestellt. Die URL des Bildes wird in das XML Dokument eingetragen und dann mit dem CSDWritingClient an die CSD übertragen. Der CSD-Server lädt sich daraufhin das Bild aus dem HTTP-Server herunter.

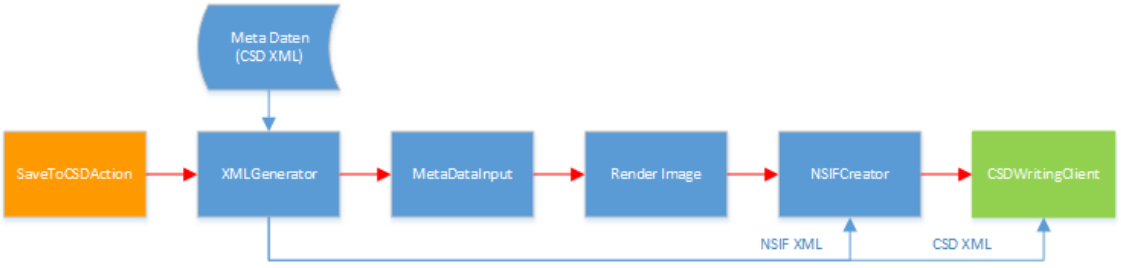

Abbildung 3.6.: Ablauf des Exportierens in die CSD. Die Action in orange ist die initiale Aktivierung des Nutzers. Die roten Pfeile zeigen die Abfolge der aufgerufenen Programmteile. In grün sind die Komponenten der Isaac.lib

4. Probleme und Code-Qualität

In diesem Kapitel wird zunächst auf einige Probleme bei der Implementierung eingegangen und anschließend die Code Qualität betrachtet.

4.1. Probleme

4.1.1. Maven und nicht Maven Bibliotheken

Alle Dependencies in den bestehenden Projekten werden durch Maven ([6]) aufgelöst. Maven ist ein Build-Management Tool, das auch die Dependency Verwaltung übernimmt. Die isaac.lib und der nisf-creator standen zum Beginn des Projektes nicht in einem Maven Repository zur Verfügung. Damit zukünftig keine lokalen Bibliotheken eingebunden werden müssen, wurden beide Bibliotheken und deren Abhängigkeiten in das Repository eingefügt. Dazu mussten die Abhängigkeiten der einzelnen Bibliotheken manuell aufgelöst werden, bevor diese in das Project Object Model (POM) der einzelnen Bibliotheken eingetragen werden können.

4.1.2. CSD upload

In der Implementierung sind beim Hochladen des Bildes einige Probleme aufgetreten. Der CSD wird mitgeteilt, wo das Bild herunterzuladen ist und das kann zu Problemen führen (Siehe Abbildung 4.1). Ein Problem, das hierbei auftreten kann, ist, dass der benötigte Netzwerk-Port nicht freigegeben ist. Um einen Port freizugeben, braucht man Administratorrechte, die nicht jeder Nutzer hat.

Außerdem ist darauf zu achten, dass die Infrastruktur erlaubt, dass der Server eine Rückverbindung zum Client aufbaut. Steht der Server in einem anderen Netz als der Client, kann diese Funktionalität durch eine Firewall blockiert werden, und das Bild lässt sich nicht übertragen.

Die Isaac.lib überträgt nicht die IP-Adresse des Clients als Download-URL, sondern denn Domain Namen des Clients. Kann der Server den Namen nicht auflösen, kann der Server das Bild ebenfalls nicht herunterladen.

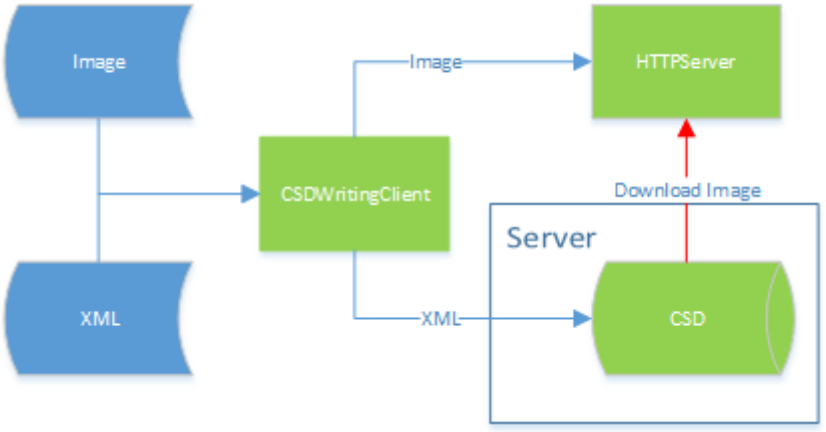

Abbildung 4.1.: Der Upload eines Bildes zur CSD erfolgt, indem der Server das Bild herunterlädt.

4.2. Code Qualität

4.2.1. Testing

Um die Anbindung an die CSD zu testen, wären umfangreiche Integrationstest nötig. Diese gestalten sich schwierig, da keine CSD Server als exklusive Testinstanzen zur Verfügung stehen.

Alle Komponenten wurden manuell getestet, um die Funktionalität zu überprüfen. Das bedeutet, dass beim Parsen der XML Dokumente im Result Handler der CSDCommons Bibliothek jedes ausgelesene Feld auf Korrektheit des Ergebnisses überprüft wurde. Es wurde darauf geachtet, dass einige Felder optional sind und dass das Fehlen dieser Informationen nicht zu Programmabstürzen führt.

Das Parsen der XML Dokumente im ResultHandler der CSDCommons Bibliothek ist besonders intensiv überprüft worden. Viele Felder sind optional. Sowohl CSDAdapter als auch SBA versuchen aber diese auszulesen. Es wurde darauf geachtet, dass dieses nicht zu Fehlern in der weiteren Verarbeitung führt.

Bei der Nutzereingabe zum Schreiben in die CSD wird die Richtigkeit und Notwendigkeit der einzelnen Felder direkt bei der Eingabe überprüft. Übergibt man ein unvollständiges XML Dokument an die Isaac.lib, so wirft diese Fehler. Um das zu vermeiden, kann der Nutzer die Anfrage erst absenden, wenn alle Eingaben korrekt erfolgt sind. Auch hier wurde für jedes Feld überprüft, ob die Eingabe den Richtlinien der CSD entspricht.

4.2.2. Find Bugs

Um in der Benutzung keine Programmabstürze durch Programmierfehler zu erhalten, wurde der Code mit Findbugs [7] überprüft. Findbugs untersucht den Java bytecode nach Bug Patterns. Vermutliche Fehler z.B. Nullpointer oder auch schlechter Stil wie == Operatoren anstelle der equals Methode werden von der Bibliothek erkannt und dem Nutzer zur Verbesserung vorgeschlagen. In der Implementierung wurden alle Anmerkungen von Findbugs behoben. Lediglich die Fehler in der JHotDraw GUI Bibliothek, auf dem der SBA aufbaut, wurden nicht behoben. Dafür wäre ein exaktes Verständnis des gesamten JHotDraw Frameworks notwendig.

5. Verbesserungen des Systems

5.1. Ausblick

5.1.1. Direktes Laden aus der CSD mit der UUID

Jedes Objekt in der CSD hat eine eindeutige UUID. Diese wird verwendet, um die Datensätze eindeutig zu identifizieren. Bekommt ein Bildauswerter den Auftrag ein bestimmtes Bild auszuwerten, bekommt er die UUID des Bildes mitgeteilt. Die Eingabeoberfläche unterstützt das Suchen nach der UUID, jedoch wird das Ergebnis anschließend in der Tabelle angezeigt. Da diese Suche nur ein einziges Ergebnis liefern kann, ist das erneute Anzeigen und Aussuchen des Nutzers überflüssig. Das direkte Laden eines Bildes anhand der UUID könnte die Arbeit des Bildauswerters verbessern.

5.1.2. SBA-Import durch Geoviewer

Bisher werden die Objekte aus der CSD auf dem Geoviewer dargestellt. Diese werden unter anderem verwendet, um die zugehörigen Metadaten auf dem MetaDaten Display anzuzeigen. Hier kann eine weitere Verbindung zwischen SBA und Geoviewer hergestellt werden. Durch Auswahl des Objektes im Geoviewer wird das Bild im SBA geladen und man erspart sich eine erneute Abfrage der CSD auf dem SBA.

5.1.3. Geoviewer Vorschau beim Laden in den SBA

Nachdem der Nutzer auf dem SBA eine CSD Abfrage ausgeführt hat, bekommt er alle Ergebnisse tabellarisch aufgelistet. Eine der Auswahlinformationen ist sind die Koordinaten des Bildes. An dieser Stelle könnte auf Anfrage des Nutzers die Karte des Geoviewers an die entsprechende Stelle bewegt werden. So kann sich der Auswerter die Umgebung der Ergebnisse ansehen, bevor er das Bild im SBA lädt. Damit nicht nur die Karte zu sehen ist, muss ein Data Objekt angelegt werden. Eine Möglichkeit wäre, hierfür einen Task an den CSDAdapter zu senden, der genau das eine Datenobjekt aus der CSD lädt und für den Geoviewer bearbeitet.

5.1.4. Assoziationen

Bisher werden beim Lesen und Schreiben der CSD die Assoziationen zwischen den Datensätzen nicht ausgelesen. Diese Zusammenhänge zwischen den Daten sind wichtige Informationen. Hier bieten sich zwei Verbesserungsmöglichkeiten an. Beim Schreiben in die

CSD sollte überprüft werden, ob eine neue Assoziation sinnvoll ist. Ist das Bild aus der CSD geladen, sollte das ausgewertet Bild auf das Original verweisen. Außerdem müssten diese Verknüpfungen beim Laden ebenfalls visualisiert werden, um anzuzeigen, wie die Bilder zusammenhängen. Ein Vorteil hierbei ist, dass man sofort erkennt, welches Bild noch nicht ausgewertet ist und welches bereits bearbeitet wurde. Für die sinnvolle Darstellung der Verknüpfungen müsste man eine geeignete Visualisierung erarbeiten. Eine Möglichkeit wäre eine Baumstruktur.

5.1.5. Reports

In dieser Arbeit haben wurde hauptsächlich das Laden und Speichern der Bilder für den SBA implementiert. Ein wichtiger Bestandteil der Bildauswertung ist der Bericht, der während der Auswertung angefertigt werden muss. Eine Verbesserung des Arbeitsplatzes kann erzielt werden, indem der Nutzer auch die Reports in die CSD eintragen kann. Idealerweise wird dieser Bericht dann mit dem Original und dem ausgewertetem Bild verknüpft. Eine Möglichkeit wäre, diesen Upload ebenfalls in den SBA einzubauen. Zusätzlich zu den Metainformationen gibt man den Dateipfad des Berichts an und dieser wird zusammen mit dem Bild in die CSD eingetragen. Alternativ könnte man evaluieren, ob eine direkte Integration eines Texteditors oder eines spezialisierten Werkzeugs in den SBA sinnvoll ist.

6. Anhang

A. CSD Reading Example XML Dokument

```
<?xml version="1.0" encoding="UTF-8"?>
<PRODUCT>
  <CARD>
    <dateTimeModified>2013-01-17T16:05:01.731Z</dateTimeModified>
    <identifier>a629bfac-60bf-11e2-bccb-0050568e0049</identifier>
    <numberOfParts>1</numberOfParts>
    <sourceDateTimeModified>2013-01-17T16:05:01.731Z</
        sourceDateTimeModified>
    <sourceLibrary>DEU-IOSB_majex12</sourceLibrary>
    <status>NEW</status>
  </CARD>
  <FILE>
    <archived>false</archived>
    <creator>IOSB</creator>
    <dateTimeDeclared>2011-10-02T09:26:27.000Z</dateTimeDeclared>
    <extent>1.751220703125</extent>
    <format>application/x-nitf</format>
    <formatVersion>02.10</formatVersion>
    <isProductLocal>true</isProductLocal>
    <productURL>http://153.96.15.182:8090/proxy/a5e9f786-60bf-11
        e2-bccb-0050568e0049</productURL>
    <title>blaue Moschee Uebersichtsbild</title>
  </FILE>
  <METADATASECURITY>
    <classification>UNCLASSIFIED</classification>
    <policy>DEU</policy>
    <releasability>XXN</releasability>
  </METADATASECURITY>
  <PART>
    <COMMON>
      <identifierMission>DNBL</identifierMission>
      <identifierUUID>00912402-0001-0002-0003-000120095423</
          identifierUUID>
```

```
      <type>IMAGERY</type>
    </COMMON>
    <COVERAGE>
      <spatialCountryCode>AFG</spatialCountryCode>
      <spatialGeographicReferenceBox>
        <lowerRightPoint>
          <latitude>36.703538</latitude>
          <longitude>67.118332</longitude>
        </lowerRightPoint>
        <upperLeftPoint>
          <latitude>36.714442</latitude>
          <longitude>67.104637</longitude>
        </upperLeftPoint>
      </spatialGeographicReferenceBox>
      <temporalStart>2009-11-03T13:43:41.000Z</temporalStart>
    </COVERAGE>
    <IMAGERY>
      <category>EO</category>
      <decompressionTechnique>NC</decompressionTechnique>
      <identifier>blueMMos#1</identifier>
      <numberOfBands>3</numberOfBands>
    </IMAGERY>
    <SECURITY>
      <classification>UNCLASSIFIED</classification>
      <policy>DEU</policy>
      <releasability>XXN</releasability>
    </SECURITY>
    <partIdentifier>1</partIdentifier>
  </PART>
  <RELATED_FILE>
    <URL>http://153.96.15.182:8090/proxy/a60c73a8-60bf-11e2-bccb
        -0050568e0049</URL>
    <creator>DEU-IOSB_majex12</creator>
    <dateTimeDeclared>2013-01-17T16:05:01.547Z</dateTimeDeclared>
    <extent>0.10154247283935547</extent>
    <fileType>THUMBNAIL</fileType>
    <isFileLocal>true</isFileLocal>
  </RELATED_FILE>
  <RELATED_FILE>
    <URL>http://153.96.15.182:8090/proxy/a62070da-60bf-11e2-bccb
        -0050568e0049</URL>
    <creator>DEU-IOSB_majex12</creator>
    <dateTimeDeclared>2013-01-17T16:05:01.679Z</dateTimeDeclared>
    <extent>0.599853515625</extent>
    <fileType>OVERVIEW</fileType>
    <isFileLocal>true</isFileLocal>
  </RELATED_FILE>
  <SECURITY>
    <classification>UNCLASSIFIED</classification>
    <policy>DEU</policy>
    <releasability>XXN</releasability>
  </SECURITY>
```

```
</PRODUCT>
```

B. CSD Writing Example XML Dokument

```
<?xml version="1.0" encoding="UTF-8"?>
<PRODUCT>
  <FILE>
    <creator>SBA</creator>
    <dateTimeDeclared>2015-03-11T14:52:14.000Z</dateTimeDeclared>
    <productURL>H:\SBA\temp\a629bfac-60bf-11e2-bccb-0050568
        e00494503771203963546572.nsif</productURL>
  </FILE>
  <METADATASECURITY>
    <classification>UNCLASSIFIED</classification>
    <policy>DEU</policy>
    <releasability>XXN</releasability>
  </METADATASECURITY>
  <PART>
    <COMMON>
      <identifierMission>DNBL</identifierMission>
      <identifierUUID>00912402-0001-0002-0003-000120095423</
          identifierUUID>
      <type>IMAGERY</type>
    </COMMON>
    <COVERAGE>
      <spatialCountryCode>AFG</spatialCountryCode>
      <spatialGeographicReferenceBox>
        <lowerRightPoint>
          <latitude>36.703538</latitude>
          <longitude>67.118332</longitude>
        </lowerRightPoint>
        <upperLeftPoint>
          <latitude>36.714442</latitude>
          <longitude>67.104637</longitude>
        </upperLeftPoint>
      </spatialGeographicReferenceBox>
      <temporalStart>2009-11-03T13:43:41.000Z</temporalStart>
    </COVERAGE>
    <IMAGERY>
      <category>EO</category>
      <decompressionTechnique>NC</decompressionTechnique>
      <identifier>blueMMos#1</identifier>
      <numberOfBands>3</numberOfBands>
      <numberOfColumns>1563</numberOfColumns>
      <numberOfRows>1563</numberOfRows>
    </IMAGERY>
    <SECURITY>
      <classification>UNCLASSIFIED</classification>
      <policy>DEU</policy>
      <releasability>XXN</releasability>
    </SECURITY>
    <partIdentifier>1</partIdentifier>
```

```
  </PART>
  <SECURITY>
    <classification>UNCLASSIFIED</classification>
    <policy>DEU</policy>
    <releasability>XXN</releasability>
  </SECURITY>
</PRODUCT>
```

Literaturverzeichnis

[1] Erich Gamma and Thomas Eggenschwiler, "Jhotdraw as open-source project," http://www.jhotdraw.org/, stand: 2015-03-03.

[2] Fraunhofer-Institut für Optronik, Systemtechnik und Bildauswertung, "Interaktive visualisierung integrierter geodaten (ivig)," http://www.iosb.fraunhofer.de/servlet/is/7583/Interaktive%20Visualisierung%20Integrierter%20Geodaten%20IVIG_de.pdf?command=downloadContent&filename=Interaktive%20Visualisierung%20Integrierter%20Geodaten%20IVIG_de.pdf, 2014, stand: 2015-03-03.

[3] ——, "Objektidentifikation (RecceMan®)," http://www.iosb.fraunhofer.de/servlet/is/4425/, stand: 2015-03-03.

[4] T. Reiter, "Innovative interaktionstechniken für einen multi-display-arbeitsplatz zur bildauswertung," Bachelor Thesis, KIT, 2015.

[5] Fraunhofer-Institut für Optronik, Systemtechnik und Bildauswertung, "Coalition shared data server," http://www.iosb.fraunhofer.de/servlet/is/4637/Produktflyer_CSD-Server_english.pdf?command=downloadContent&filename=Produktflyer_CSD-Server_english.pdf, 2011, stand: 2015-03-03.

[6] The Apache Software Foundation, "Welcome to apache maven," http://maven.apache.org/, stand: 2015-03-03.

[7] FindBugs Development Team, University of Maryland, "Findbugs - find bugs in java programs," http://findbugs.sourceforge.net/, stand: 2015-03-03.

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

Karlsruhe, 12.03.2015

.....

(**Kai Westerkamp**)