Web service-based exploration of Earth Observation time-series data for analyzing environmental changes

Dissertation

zur Erlangung des akademischen Grades
doctor rerum naturalium
(Dr. rer. nat.)

vorgelegt dem Rat der Chemisch-Geowissenschaftlichen Fakultät
der Friedrich-Schiller-Universität Jena

von M. Sc. Jonas Eberle
geboren am 05.12.1984 in Heidelberg

Gutachter:

1. Prof. Dr. Christiane Schmullius (Friedrich-Schiller-Universität Jena)

2. Prof. Dr. Lars Bernard (Technische Universität Dresden)

Tag der Verteidigung: 07.11.2019

Abstract

The increasing amount of freely available Earth observation (EO) data requires a tremendous change, in order to property handle the number of observations and storage size thereof used for the environmental monitoring of land surfaces. In the near future, the processing of EO data will need to be undertaken close to EO data archives as the downloading of large amounts requires too much time and storage capacity. It is not only scientists and geospatial processing specialists who work with EO data; stakeholders, thematic experts, and software developers do too. Due to open data strategies and the increasing size of data archives, a new market has been developed to provide analysis and application-ready data, services, and platforms. There is thus a great demand for improving the discovery, access, and analysis of EO data in line with the new possibilities of web-based infrastructures. With the aim of bridging the gap between users and EO data archives, various topics have been researched: 1) user requirements and their relation to web services and output formats; 2) technical requirements for the discovery, access, and analysis of multi-source EO time-series data, and 3) management of EO time-series data focusing on analysis and application-ready data.

State of the Art: Current web technologies enable the interactive exploration of EO data in distributed infrastructures based on web-service architectures. EO data has been made available through web services for search, download, exploration, and analysis. Standardized service specifications that support the interoperable use of the services exist (e.g., from the Open Geospatial Consortium and the World Wide Web Consortium). Although web services and cloud-based infrastructures are widely used in the EO domain for data discovery, access, and processing, data providers employ different service specifications, and response formats are mainly optimized for machine-to-machine communication, in contrast to formats suitable for non-technical end users (e.g., scientists and thematic experts). Although web technologies enable modern web services, there remains a gap between users and providers.

Review: Existing web services for EO data discovery and access, time-series data processing, and web-based EO platforms are reviewed and related to the requirements of user personas. For multi-source discovery and access services, a key issue is having to learn different service specifications. The diversity of data providers and web services available means that specific knowledge of these systems and specifications is required. Although standards for service specifications and brokering systems for the discovery of EO data exist, improvements are still necessary to meet the requirements of different user personas. For the processing of EO time-series data, various data formats and

preprocessing steps need to be considered. A conversion of the data format and structure is often required to conduct analyses in programming languages and with analysis tools. Today, there remains a gap between EO time-series data access and analysis tools, which needs to be addressed to simplify work with such data. A variety of cloud infrastructures and platforms for EO data processing have been launched in recent years (e.g., Google Earth Engine, Copernicus Data and Information Access Services), and software exists to provide web-based interactive platforms (e.g., Open Data Cube and Jupyter Notebook). These enable users to access and process EO data using web browsers rather than geographic information systems installed on the user's computer. However, each infrastructure or platform hosts different sets of EO data, analysis tools, algorithm development environments, and other functions. Thus, the decision regarding which of these to use depends on the specific knowledge of the user, the data, and the analysis to be conducted.

Concepts, methods, and applications: To bridge the gap between users and data providers, concepts and methods have been defined, focusing on service-based exploration, multi-source EO time-series data discovery and access, and uniform EO time-series data management and analysis. For each of these, the requirements of different user personas are considered, which leads to a uniform service specification, multiple and user-specific response formats, a harmonized data structure, and analysis tools directly linked to EO data. New concepts and methods for user-aligned platforms, services, and output formats are described. These are demonstrated in, for example, web and mobile applications with the aim of simplifying discovery, access, and analysis in order to focus on the exploration of EO time-series data.

Results: Results are presented and discussed in relation to user-specific exploitation of EO time-series data: 1) the centralization of EO time-series data at regional scales enables the development of user-specific platforms; 2) the harmonization of service interfaces makes data discovery, access, and analysis uniform; and 3) the standardization of EO time-series data structure and formats simplifies analysis and usage in geospatial tools.

Conclusions and outlook: The complete workflow of EO time-series data-handling, with a focus on user-aligned web services, is reviewed and new concepts are designed and developed. It can be concluded that there is a need for more user-driven design and development of services, which would lead to automated workflows, harmonized service interfaces, and user-aligned data formats. Research topics such as interoperable time-series data discovery and access, data cubes, cloud-based infrastructures, and analysis-ready data highlight the nature of the next generation of web-based EO data exploration.

Kurzfassung

Die zunehmende Menge an frei verfügbaren Erdbeobachtungsdaten (EO-Daten) erfordert eine Veränderung im Umgang mit der Anzahl an Beobachtungen sowie der Speichergröße der Daten, die für Umweltüberwachungen von Landoberflächen verwenden werden. Schon in naher Zukunft muss die Datenverarbeitung zu den Datenarchiven geschoben werden, da das Herunterladen großer Zeitreihendaten zu viel Zeit und Speicherkapazitäten erfordert. Darüber hinaus arbeiten nicht mehr nur Wissenschaftler und GIS-Entwickler mit den Daten, sondern unter anderem auch thematische Experten und allgemeine Softwareentwickler. Daher besteht ein hoher Bedarf an einer Verbesserung der Suche, des Zugriffs und der Analyse von EO-Daten mit Bezug auf neue Möglichkeiten in web- und cloud-basierten Infrastrukturen. Mit dem Ziel, die Lücke zwischen Nutzern und Datenprovidern zu schließen, wurden in dieser Arbeit verschiedene Aspekte untersucht und erforscht: 1) Benutzeranforderungen und deren Auswirkungen auf die Bereitstellung von Webdiensten und Datenformate. 2) Technische Anforderungen für die Suche, den Zugriff und die Analyse von EO-Zeitreihendaten. 3) Verwaltung von EO-Zeitreihendaten in Verknüpfung mit anschließender Datenanalyse.

Stand der Technik und Forschung: Aktuelle Webtechnologien ermöglichen die interaktive Erforschung von EO-Daten in verteilten Infrastrukturen auf Basis von Webdienst-Architekturen. Webdienste für die Suche, den Zugriff und die Analyse stehen zumeist über standardisierte Spezifikationen zur Verfügung, die eine interoperable Nutzung der Dienste ermöglichen. Obwohl Webdienste und Cloud-basierte Infrastrukturen für die Suche nach, den Zugriff auf und die Verarbeitung von EO-Daten weit verbreitet sind, nutzen Datenanbieter unterschiedliche Spezifikationen und Ausgabeformate. Bisher sind diese hauptsächlich für die Maschine-zu-Maschine-Kommunikation optimiert, die für nicht-technische Endanwender (z.B. Wissenschaftler, thematische Experten) allerdings nicht geeignet sind. Obwohl aktuelle Webtechnologien moderne Webdienste ermöglichen, gibt es immer noch eine Lücke zwischen Nutzern und Anbietern im Bereich der Erdbeobachtung.

Analyse: Eine Analyse vorhandener Webdienste für die Suche nach, den Zugriff auf und die Verarbeitung von EO-Daten sowie web-basierten Verarbeitungs- und Datenplattformen wurde durchgeführt und auf die Anforderungen verschiedener Benutzergruppen untersucht. Die Vielfalt der Datenanbieter und deren Webdienste erfordern spezifische Kenntnisse in diesen Systemen und Spezifikationen. Obwohl es Standards für Webdienste und Brokeringsysteme im Geodatenbereich gibt, ist noch Verbesserungsbedarf vorhanden, um den Anforderungen der Benutzer gerecht zu

werden. Für die Verarbeitung von EO-Daten sind unterschiedliche Datenformate und Vorverarbeitungsschritte zu berücksichtigen. Eine Konvertierung von Datenformat und -struktur ist oft notwendig, um Analysen in Anwendungen durchführen zu können. Noch heute besteht eine Lücke zwischen dem Zugriff auf EO-Zeitreihendaten und Analysewerkzeugen. Diese muss geschlossen werden, um die Arbeit mit EO-Daten für alle Benutzergruppen zu erleichtern. In den letzten Jahren wurde eine Vielzahl von Cloud-Infrastrukturen und Plattformen für die Verarbeitung von EO-Daten eingeführt und Software für webbasierte interaktive Plattformen zur Verfügung gestellt. Beide ermöglichen es den Benutzern, über Webbrowser auf EO-Daten zuzugreifen und diese zu verarbeiten, anstatt diese auf dem Computer des Benutzers durchzuführen. Jede Infrastruktur und Plattform enthält jedoch verschiedene Daten und Analysetools. Somit hängt eine Entscheidung, welches System verwendet werden kann, von den Kenntnissen des Benutzers und den zu analysierenden Daten und Analysen ab.

Konzepte, Methoden und Anwendungen: Um das Auffinden, den Zugriff und die Analyse von EO-Daten zu vereinfachen und damit die Lücke zwischen Nutzern und Datenanbietern zu schließen, wurden die Bereiche Webdienst-basierte Erforschung von EO-Daten, standardisierte Suche und Zugriff verschiedener EO-Daten sowie einheitliches Datenmanagement und -analyse erforscht. Auf Basis der Anforderungen verschiedener Benutzergruppen wurden einheitliche Webdienstspezifikationen, spezifische Datenformate sowie harmonisierte Datenstrukturen in Verbindung mit Analysewerkzeugen entwickelt. Diese wurden in Beispielanwendungen demonstriert.

Ergebnisse: Der gesamte Ablauf zur Verarbeitung von EO-Daten wurde im Rahmen dieser Arbeit analysiert sowie mit Schwerpunkt auf benutzerorientierten Webdiensten neu entworfen und entwickelt. Ergebnisse in Bezug auf die benutzerspezifische Erforschung von EO-Zeitreihendaten können wie folgt aufgezeigt werden: 1) Die Zentralisierung der EO-Daten auf regionaler Ebene ermöglicht die Entwicklung benutzerspezifischer Plattformen. 2) Die Harmonisierung der Webdienste vereinheitlicht die Datensuche, den Zugriff und die Analyse. 3) Die Standardisierung der EO-Datenstruktur vereinfacht die Analyse und die Anbindung an geografische Werkzeuge.

Schlussfolgerungen und Ausblick: Eindeutig erkennbar ist ein Bedarf an einer stärker benutzerorientierten Gestaltung und Entwicklung von Diensten und Plattformen, die zu automatisierten Arbeitsabläufen, benutzerorientierten Datenformaten und harmonisierten Webdiensten führen muss. Diese sind auch für zukünftige Forschungsarbeiten wie interoperable Zeitreihendatensuche und -datenzugriff, Datenwürfel, Cloud-basierte Infrastrukturen und die Bereitstellung von analysierbaren Daten relevant.

Publications

Scientific peer-reviewed paper publications

O. Semenova, L. Lebedeva, N. Volkova, I. Korenev, M. Forkel, **J. Eberle** & M. Urban (2015): Detecting immediate wildfire impact on runoff in a poorly-gauged mountainous permafrost basin. Hydrological Sciences Journal. 60:7-8, 1225-1241.

Urban, M., M. Forkel, **J. Eberle**, C. Schmullius & M. Herold (2014): Pan-arctic climate and land cover trends derived from multi-variate and multi-scale analysis (1981 - 2012). - Remote Sensing - Special Issue on Remote Sensing of Changing Northern High Latitude Ecosystems, 6, 3, 2296-2316.

Eberle, J., S. Clausnitzer, C. Hüttich & C. Schmullius (2013). Multi-Source Data Processing Middleware for Land Monitoring within a Web-Based Spatial Data Infrastructure for Siberia. ISPRS Int. J. Geo-Inf. 2013, 2, 553-576.

Urban, M., **J. Eberle**, C. Hüttich, C. Schmullius & M. Herold (2013). Comparison of Satellite-Derived Land Surface Temperature and Air Temperature from Meteorological Stations on the Pan-Arctic Scale. Remote Sensing. 2013, 5, 2348-2367.

Eberle, J. & C. Strobl (2012). Web-based Geoprocessing and Workflow Creation for Generating and Providing Remote Sensing Products. - In: Li, S., S. Dragicevic, B. Veenendaal & M. A. Brovelli (Hrsg.): Special Issue on "Analytical Geospatial Web Services". GEOMATICA. Vol. 66, No. 1, 2012, pp. 13 - 26.

Book chapter

Eberle, J., M. Urban, A. Homolka, C. Huettich & C. Schmullius (2016): Multi-Source Data Integration and Analysis for Land Monitoring in Siberia. In: Mueller, L., A. S. Sheudshen & F. Eulenstein (ed.): Novel Methods for Monitoring and Managing Land and Water Resources in Siberia. Springer Water, 471 - 488.

Selected conference proceedings

Eberle, J., T. Taylor and C. Schmullius (2016): Easy to use time-series data access and analysis tools using standard-based geoprocessing services. In: Proceedings of IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, 2016, pp. 3614-3617.

Eberle, J., C. Hüttich, C. Schmullius (2014): Operational Earth Observation data access for automated time-series monitoring based on OGC Web Processing Services. - Proceedings of the 2014 conference on Big Data from Space, 12-14 November 2014, Frascati, Italy.

Eberle, J. & C. Schmullius (2013). User-driven data-integration for multi-source time-series data. - Proceedings of Canadian Institute of Geomatics (CIG) Annual Conference and International Conference on Earth Observation for Global Changes (EOGC), 5-7 June 2013, Toronto, Canada.

Eberle, J., S. Hese & C. Schmullius (2012). Siberian Earth System Science Cluster - A Geoportal to provide user-friendly access and analysis for Earth Observation Products. - Proceedings of EnviroInfo, 29-31 August 2012, Dessau, Germany.

Selected contributions on scientific conferences

Eberle, J. (2018): The SWOS/GEO-Wetlands Community Portal. – EuroGEOSS Workshop, 12-14 September 2018, Geneva, Switzerland.

Eberle, J. & C. Schmullius (2018): Automated Earth Observation data discovery and access using open source web frameworks. – EGU General Assembly, 8-13 April 2018, Vienna, Austria.

Eberle, J. & C. Schmullius (2017): Standardized Access and Processing of Multi-Source Earth Observation Time-Series Data within a Regional Data Middleware. – AGU Fall Meeting, 11-15 December 2017, New Orleans, USA.

Eberle, J., C. Hüttich & C. Schmullius (2016): From global observations to local information: The Earth Observation Monitor - International Conference for Free and Open Source Software for Geospatial, 24-26 August 2016, Bonn, Germany.

Eberle, J. (2016): MySeasons - Community-based mapping and monitoring of vegetation phenology - GEO-XIII Plenary - Side-Event on Citizen Science, 7-10 November 2016, St. Petersburg, Russia.

Eberle, J., A. Strauch, F. Cremer, C. Hüttich, G. Menz & C. Schmullius (2016): The SWOS Data Portal & Brokering System - A next generation knowledge hub for free wetland data, information and analysis tools - ESA Living Planet Symposium, 9-13 May 2016, Prague, Czech Republic.

Eberle, J. & C. Schmullius (2015): Geoprocessing services for Earth Observation time-series data access as basis for web and mobile application. - Earth Observation Open Science 2.0, 12-14 October 2015, Frascati, Italy.

Eberle, J., C. Hüttich & C. Schmullius (2015): mobileEOM - Vegetation change analysis in the field based on standard-compliant web services. - 9th GEO European Projects Workshop, 15-16 June 2015, Copenhagen, Denmark.

Eberle, J., C. Hüttich & C. Schmullius (2015): Automated Earth Observation time-series monitoring with OGC-compliant web services. - 36th International Symposium on Remote Sensing of Environment (ISRSE), 11-15 May 2015, Berlin, Germany.

Eberle, J., C. Hüttich & C. Schmullius (2014): The Earth Observation Monitor - Automated monitoring and alerting for spatial time-series data based on OGC web services. - AGU Fall Meeting, 15-19 December 2014, San Francisco, USA.

Eberle, J., C. Hüttich, C. Schmullius (2014): Interoperable time-series tools and web services for local land monitoring using multi-source Earth Observation data archives. - Google Earth Engine Workshop, San Francisco, USA.

Eberle, J., C. Hüttich & C. Schmullius (2013). Web-based Multi-Source Data Processing Middleware for Land Observations and Monitoring. - ESA Living Planet Symposium, 9-13 September 2013, Edinburgh, Scotland.

Eberle, J., R. Gerlach, S. Hese & C. Schmullius (2012). Siberian Earth System Science Cluster - A web-based Geoportal to provide user-friendly Earth Observation Products. - EGU General Assembly, 22-27 April 2012, Vienna, Austria.

Table of contents

Abstract ..... **i**

Kurzfassung ..... **iii**

Publications ..... **v**

Table of contents ..... **vii**
List of figures ..... **x**
List of tables ..... **xii**
List of code listings ..... **xiii**
Abbreviations ..... **xiv**

Outline ..... **xvii**

Chapter 1: Introduction **1**
1.1 Motivation **3**
1.2 History **5**
1.3 Current State of Research **7**
1.4 Scientific Questions **9**

Chapter 2: Definitions and Requirements **11**
2.1 Case Study: Satellite-based Vegetation Monitoring **12**
2.1.1 Earth Observation data 13
2.1.2 Analysis tools 19
2.1.3 Conclusions 22
2.2 User Personas **23**
2.2.1 Scientists 23
2.2.2 Thematic experts 23
2.2.3 Software developers 24
2.2.4 Summary 24
2.3 Requirements **25**
2.3.1 Web platforms 25
2.3.2 Web technologies 26
2.3.3 Data formats 28
2.3.4 Summary and conclusions 30

Chapter 3: State of the Art **31**
3.1 Web Technologies **32**
3.1.1 Web service architectures 32
3.1.2 Standardization 35
3.1.3 Structured data formats 36
3.1.4 Web service software 38
3.1.5 Cloud-based infrastructures 42
3.1.6 Web platforms 44
3.2 EO Time-series Data Services and Formats **45**
3.2.1 Discovery 45
3.2.2 Access 48
3.2.3 Brokering 51
3.2.4 Visualization 52

3.2.5 Processing and analysis 54
3.2.6 Data formats 57
3.3 Summary and Evaluation 58

Chapter 4: Review of EO Web Services, Tools, and Platforms 61
4.1 Discovery of EO Time-series Data 62
4.1.1 Data provider solutions 63
4.1.2 Brokered web service solutions 65
4.1.3 Conclusions 67
4.2 Access to EO Time-series Data 69
4.2.1 Data access services 70
4.2.2 Data download links 71
4.2.3 Data extraction services 73
4.2.4 Conclusions 74
4.3 Processing of EO Time-series Data 75
4.3.1 Programming languages 75
4.3.2 Geospatial tools 79
4.3.3 Conclusions 82
4.4 Cloud-based EO Time-series Data Platforms 83
4.4.1 Virtual environments 84
4.4.2 Processing platforms 85
4.4.3 Service platforms 89
4.4.4 Conclusions 89
4.5 Recommendations 93

Chapter 5: Concepts and Methods 95
5.1 Service-based EO Time-series Data Middleware 96
5.1.1 Concept of a regional data middleware system 97
5.1.2 User-aligned web services 102
5.1.3 User-aligned output formats 105
5.1.4 Implementation: Middleware software architecture and web services 107
5.2 Service Brokering for Multi-source Data Discovery and Access 114
5.2.1 The concept of web service brokering 115
5.2.2 Metadata model 118
5.2.3 Discovery output formats 122
5.2.4 Implementation: EO time-series data discovery and access brokering 124
5.3 Unified EO Time-series Data Structure and Analysis 129
5.3.1 Common EO time-series data structure 129
5.3.2 Specifications for data processing tools 132
5.3.3 Specifications for data analysis 135
5.3.4 Implementation: EO time-series data processing and analysis 141

Chapter 6: Example Use Cases 147
6.1 The Siberian Earth System Science Cluster 148
6.1.1 SIB-ESS-C Middleware 148
6.1.2 Web portal 150
6.2 Earth Observation Monitor 152
6.2.1 EOM Middleware 152
6.2.2 webEOM 154

6.2.3 mobileEOM 155
6.2.4 MySeasons App 157
6.3 GEO-Wetlands Community Portal 158
6.3.1 Middleware 158
6.3.2 Web portal 160
6.3.3 Open Data Cube for Wetlands 161
6.3.4 Sentinel–1 Surface Water Dynamics Toolkit 162

Chapter 7: Results and Discussion 163
7.1 Centralization of EO time-series data at regional scales 164
7.1.1 Regional data middleware approach 164
7.1.2 Application development 165
7.1.3 Regional use cases 166
7.2 Harmonization of EO Time-series Service Interfaces 169
7.2.1 WPS-based EO web services 169
7.2.2 EO data discovery broker 170
7.3 Standardization of EO Time-series Data Structure and Formats 171
7.3.1 EO time-series data structure and format 171
7.3.2 User-aligned output formats 172

Chapter 8: Conclusions and Outlook 173
8.1 Responses to the research questions 173
8.2 Conclusions 174
8.3 Outlook 176

References 179

Appendix A 195
A.1 Discovery of EO time-series data 195
A.2 Access to EO time-series data 198

Selbstständigkeitserklärung 200

Acknowledgements 201

List of figures

Figure 1.1: Overall structure of the thesis. 3
Figure 1.2: Framework of web-mapping eras. The stars indicate the approximate commencement of the era. 6
Figure 1.3: Bridging the gap between user applications and EO data archives. 9
Figure 2.1: Multi-annual time-series of MODIS EVI of Stadtbruch in Anklam 12
Figure 2.2: MODIS sinusoidal tile grid. 14
Figure 2.3: Timeline of the satellites of the Landsat program. 16
Figure 2.4: WRS–2 tiles (ascending and descending) for parts of Thuringia. 16
Figure 2.5: EO data structures for individual Sentinel, Landsat, and MODIS scenes. 18
Figure 2.6: Greenbrown analyses for Stadtbruch Anklam. 21
Figure 2.7: BFAST analyses for Stadtbruch Anklam. 21
Figure 2.8: TIMESAT analyses for Stadtbruch Anklam. 21
Figure 3.1: Client-server model involving different services provided by the server and the interactions between client and server/service (request and response). 32
Figure 3.2: Google search trend (October 2018): SOAP vs. REST. 33
Figure 3.3: Tasks of the individual standardization organizations in relation to “De Jure—De Facto” and “Domain—Infrastructure” standards. 35
Figure 3.4: Structured data in XML and JSON formats. 37
Figure 3.5: Requests to EO data using individual services and service brokering. 51
Figure 3.6: Workflow of a web-based visualization of geospatial data. 52
Figure 4.1: Text file for spatial raster time-series data structure. 79
Figure 4.2: Results of the web-based NASA Giovanni tool analysis. 86
Figure 4.3: Results of the web-based MODIS Global Subsets tool from ORNL DAAC 86
Figure 4.4: Google Earth Engine Playground web application. 88
Figure 4.5: Screenshot of a Google Earth Engine App. 88
Figure 4.6: User interface of the CEOS Open Data Cube. 88
Figure 5.1: Main concepts and methods of this thesis. 95
Figure 5.2: Concept of the methodological development 98
Figure 5.3: System architecture of the regional data middleware system 100
Figure 5.4: Connections of client applications to the geospatial service infrastructure. 102
Figure 5.5: Traditional discovery, access, and analysis request/response compared to a uniform web service request. 104
Figure 5.6: Automated geoprocessing service chaining using data access and data analysis services. 105
Figure 5.7: Time-series plots provided as outputs while accessing the data. 106
Figure 5.8: Spreadsheet file output showing statistical summaries. 107
Figure 5.9: WPS discovery request and an example of CSV output. 109
Figure 5.10: WPS access/integration request, file directory output, and CSV output 111

Figure 5.11: WPS analysis requests for breakpoint and trend calculations. 113
Figure 5.12: Concept of the discovery and access broker for satellite data discovery. .116
Figure 5.13: Comparison of quick-look images of a Sentinel–1 scene. 117
Figure 5.14: Metadata mapping between multiple data providersl. 118
Figure 5.15: The data discovery broker integrates post-processing steps.....119
Figure 5.16: Cloud cover calculation for a Landsat-8 scene in an area of interest.....121
Figure 5.17: On-the-fly visualization of a satellite scene using the NDVI layer. 122
Figure 5.18: Spreadsheet file output showing scenes from a data discovery request ..123
Figure 5.19: Summary output generated from the discovery result list. 123
Figure 5.20: System architecture of the EO time-series data discovery and access broker. 124
Figure 5.21: Time-series observations extracted for an area of interest. 131
Figure 5.22: Screenshot of a Jupyter Notebook with preloaded time-series data. 134
Figure 5.23: Graphic plots from geospatial data layers. 141
Figure 5.24: Visualization of the postprocessed vector Shapefile based on BFAST.....141
Figure 6.1: Comparison of MODIS Terra and Aqua LST estimates and air temperature records from an individual meteorological station 149
Figure 6.2: Screenshots of the Siberian Earth System Science Cluster Geoportal.....151
Figure 6.3: Screenshot of the webEOM portal. 155
Figure 6.4: Interactions between the mobileEOM app and EOM middleware services.156
Figure 6.5: EOM middleware services are used in the MySeasons App. 157
Figure 6.6: Screenshot of the GEO-Wetlands Community Portal.....160
Figure 6.7: User interface of the CEOS Open Data Cube 161
Figure 6.8: Sentinel–1 Water Dynamics Toolkit. 162
Figure 7.1: Top five countries with the most users and most active users of webEOM and mobileEOM.....167
Figure 7.2: Combination of MODIS NDVI and MODIS Burned Area datasets 167
Figure 7.3: Twenty-six year Landsat time-series images showing a forest site near the village of Yelwa on Mambilla Plateau, Nigeria. Results of MODIS analysis show the pattern and years of deforestation. 168

List of tables

Table 2.1: Sentinel missions of the EU Copernicus program. 17
Table 2.2: Data formats for each of the satellite missions. 18
Table 2.3: Potential output and response requirements for service outputs. 29
Table 3.1: Comparison of the synchronous and asynchronous execution of services. 34
Table 3.2: Limits of the serverless tools from Amazon and Google. 43
Table 3.3: Core methods of the OGC CSW specification. 46
Table 3.4: Core methods of the OGC WCS specification and WCS-EO extension. 50
Table 3.5: Core methods of an OGC Web Map Service. 53
Table 3.6: Core methods of an OGC WPS. 56
Table 4.1: List of services for satellite data discovery for Landsat, Sentinel, MODIS. 62
Table 4.2: List of web services for Landsat, Sentinel, MODIS satellite data access. 69
Table 4.3: Data providers with services for the direct extraction of time-series data. 73
Table 4.4: EO time-series data infrastructure and web platforms. 83
Table 4.5: Comparison of user requirements between different web services providers for discovery and access of EO data. 91
Table 4.6: Comparison of user requirements between different processing and service platforms for access and analysis of EO data. 92
Table 5.1: Inputs and outputs for a user-specific EO time-series data discovery. 109
Table 5.2: Inputs and outputs for a user-specific EO time-series data access. 111
Table 5.3: Inputs and outputs for a user-specific EO time-series data analysis. 113
Table 5.4: Metadata items and their appearance in external metadata catalogues. 119
Table 5.5: Data inputs for the algorithms BFAST, Greenbrown, and TIMESAT 136
Table 5.6: Scientific time-series analysis tools and the resulting regular output files, as defined by the algorithm, and the postprocessed output files. 140
Table 5.7: Metadata schema for a raster time-series dataset. 145
Table 6.1: Data sources integrated in the SIB-ESS-C middleware system. 149
Table 6.2: Data sources within the EOM middleware. 153
Table 7.1: Platforms developed in relation to user requirements. 165
Table A.1: List of search parameters for Sentinel data. 195
Table A.2: List of search parameters for collections within USGS Earth Explorer. 195
Table A.3: List of search parameters for scenes within USGS Earth Explorer. 196
Table A.4: List of search parameters for collections within NASA CMR. 196
Table A.5: List of search parameters for satellite scenes within NASA CMR. 196
Table A.6: List of parameters to use the Sentinel-Hub WFS service. 197
Table A.7: List of parameters to order pre-processed satellite data using ESPA. 198
Table A.8: List of parameters to use the Sentinel-Hub WCS service for download. 199
Table A.9: List of parameters to use the Sentinel-Hub FIS for data extraction. 199

List of code listings

Listing 3.1: Implementation of a process using PyWPS. 39
Listing 3.2: Implementation of a process using OpenCPU. 40
Listing 3.3: Provision of a web service using Python Flask. 41
Listing 4.1: Python source code for working with *Pandas* 76
Listing 4.2: Example work with the *xarray* library using raster time-series data.....76
Listing 4.3: Example of working with multi-band raster data using *RasterBrick*. 78
Listing 4.4: Example of working with single time-series data using the *TS* function.....78
Listing 4.5: Example of working with spatial time-series data using *Raster*TS* object....78
Listing 4.6: Example of working in GRASS GIS with spatial time-series data.....80
Listing 4.7: Example workflow for ingesting raster time-series data into Open Data Cube. 80
Listing 4.8: Example configuration file for ingestion into Rasdaman.....81
Listing 5.1: Python-based source code for undertaking data discovery.....125
Listing 5.2: Python code to download all scenes found during the discovery search. ...128
Listing 5.3: Python-based source code for conducting data extraction for Sentinel. 128
Listing 5.4: Example VRT files for two bands referencing external GeoTIFF files. 131
Listing 5.5: RasterBrick output in R using the VRT dataset. 131
Listing 5.6: The ingested data can be directly loaded using the *datacube* library. 135
Listing 5.7: Processing log file with download and processing steps. 138
Listing 5.8: Integration of MODIS Vegetation Index data for an area of interest with further processing parameters applied.....143
Listing 5.9: Accessing pixel-based Landsat-8 and MODIS Vegetation index data for a given point of interest from Google Earth Engine. 143
Listing 5.10: Using the DataManagement class of pyEOM 144
Listing 5.11: Using TIMESAT analysis tool from the pyEOM library. 146
Listing A.1: Filtering Sentinel-1 Collection using the Python Earth Engine library.....196
Listing A.2: Data access for point-based extraction of time-series. 199

Abbreviations

API	Application Programming Interface
ARD	Analysis-ready data
AWS	Amazon Web Services
BFAST	Breaks For Additive Seasonal and Trend
CEN	European Committee for Standardization
CEOS	Committee on Earth Observation Satellites
CMR	NASA Common Metadata Repository
CMS	Content Management System
CODE-DE	Copernicus Data and Exploitation Platform – Germany
CRS	Coordinate Reference System
CSV	Comma-separated values
CSW	OGC Catalogue Service for Web
CWIC	CEOSS WGISS Integrated Catalog
DAAC	Distributed Active Archive Center
DIAS	Copernicus Data and Information Access Services
EO	Earth Observation
EOM	Earth Observation Monitor
EROS	Earth Resources Observation and Science
ESA	European Space Agency
ESA RSS	ESA Research & Service Support
ESA TEP	ESA Thematic Exploitation Platform
ESPA	EROS Science Processing Architecture
ETM+	Enhanced Mapper Plus
EU	European Union
EVI	Enhanced Vegetation Index
FedEO	ESA Federated Earth Observation Gateway
FIS	Feature Information Service
GDAL	Geospatial Data Abstraction Library
GEE	Google Earth Engine
GEO	Group on Earth Observation
GEOSS	Global Earth Observation System of Systems
GHCN	Global Historical Climate Network
GIS	Geographic Information System
GML	Geography Markup Language
GRASS	Geographic Resources Analysis Support System
GSOD	Global Surface Summary of the Day
HDF	Hierarchical Data Format
HDF-EOS	Hierarchical Data Format-Earth Observing System
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
INSPIRE	Infrastructure for spatial information in Europe
ISO	International Organization of Standardization
JEODPP	JRC Earth Observation Data and Processing Platform
JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation

LPDAAC	Land Processed Distributed Active Archive Center
LST	Land Surface Temperature
MODIS	Moderate Resolution Imaging Spectroradiometer
MODIS VI	MODIS Vegetation Index
MSS	Multi-Spectral Sensor
NASA	National Aeronautics and Space Administration
NDVI	Normalized Difference Vegetation Index
NOAA	National Oceanic and Atmospheric Administration
NSIDC	National Snow and Ice Data Center
OASIS	Organization for the Advancement of Structured Information Standards
ODC	Open Data Cube
OGC	Open Geospatial Consortium
OLCI	Ocean and Land Colour Instrument
OLI	Operational Land Imager
ORNL	Oak Ridge National Laboratory
PNG	Portable Network Graphics
QA	Quality Assurance
RDF	Resource Description Framework
REST	Representational State Transfer
RPC	Remote Procedure Calls
SDI	Spatial Data Infrastructure
SIB-ESS-C	Siberian Earth System Science Cluster
SLD	OGC Styled Layer Descriptor
SLSTR	Sea and Land Surface Temperature Radiometer
SNAP	Sentinel Application Platform
SOAP	Simple Object Access Protocol
SOS	OGC Sensor Observation Service
STAC	SpatioTemporal Asset Catalog
TIRS	Thermal Infrared Sensor
TM	Thematic Mapper
UI	User Interface
URL	Uniform Resource Locator
USGS	United States Geological Survey
USGS EE	USGS Earth Explorer
UTM	Universal Transverse Mercator
UUID	Unique identifier
VRT	Virtual Raster Table
W3C	World Wide Web Consortium
WCPS	OGC Web Coverage Processing Service
WCS	OGC Web Coverage Service
WFS	OGC Web Feature Service
WGISS	CEOS Working Group on Information Systems and Services
WKT	Well-known-text
WMS	OGC Web Map Service
WPS	OGC Web Processing Service
WRS	Worldwide Reference System
XML	Extensible Markup Language
YAML	YAML Ain’t Markup Language

Outline

Chapter 1: Introduction 1
1.1 Motivation 3
1.2 History 5
1.3 Current State of Research 7
1.4 Scientific Questions 9

Chapter 2: Definitions and Requirements 11
2.1 Case Study: Satellite-based Vegetation Monitoring 12
2.2 User Personas 23
2.3 Requirements 25

Chapter 3: State of the Art 31
3.1 Web Technologies 32
3.2 EO Time-series Data Services and Formats 45
3.3 Summary and Evaluation 58

Chapter 4: Review of EO Web Services, Tools, and Platforms 61
4.1 Discovery of EO Time-series Data 62
4.2 Access to EO Time-series Data 69
4.3 Processing of EO Time-series Data 75
4.4 Cloud-based EO Time-series Data Platforms 83
4.5 Recommendations 93

Chapter 5: Concepts and Methods 95
5.1 Service-based EO Time-series Data Middleware 96
5.2 Service Brokering for Multi-source Data Discovery and Access 114
5.3 Unified EO Time-series Data Structure and Analysis 129

Chapter 6: Example Use Cases 147
6.1 The Siberian Earth System Science Cluster 148
6.2 Earth Observation Monitor 152
6.3 GEO-Wetlands Community Portal 158

Chapter 7: Results and Discussion 163
7.1 Centralization of EO time-series data at regional scales 164
7.2 Harmonization of EO Time-series Service Interfaces 169
7.3 Standardization of EO Time-series Data Structure and Formats 171

Chapter 8: Conclusions and Outlook 173
8.1 Responses to the research questions 173
8.2 Conclusions 174
8.3 Outlook 176

References 179

Appendix 195

Chapter 1: Introduction

Changes of the environment are ongoing and need to be observed on an ongoing basis to achieve environmental and sustainable development goals (Reid et al. 2010). Earth observation (EO) data can foster monitoring activities to analyze these environmental changes. The importance of EO data has been recognized by diverse research communities, funding agencies, and governments, with new satellites continuing to be launched (Berger et al. 2012). Archives of EO data enable the derivation of information about the changing planet. Many of these data archives have been made available to the public in recent years in terms of a free and open data policy (Woodcock et al. 2008; Aschbacher & Milagro-Pérez 2012). This addresses the increasing demands of users analyzing environmental changes and leads to research and the capability to establish monitoring services for environmental purposes using EO data (Wulder et al. 2012; Roy et al. 2014; Turner et al. 2015).

A wide range of activities is based on the analysis of EO time-series data as these can be used on an ongoing basis to identify trends and changes of the environment (Gutman & Masek 2012; Lasaponara & Lanorte 2012; Kuenzer et al. 2015). Using EO time-series data allows the study of environmental changes around the world. EO data is accessible as near real-time data, though it also comprises archival material dating back more than 40 years. This allows the derivation of ready-to-use information for several subjects (e.g., biodiversity, ecology, and agriculture) in support of science and policy-making (Atzberger 2013; Kuenzer et al. 2014; Pettorelli et al. 2014; Turner et al. 2015). Research on the analysis of vegetation time-series data has been widely conducted, resulting in analysis tools that can be used in operational land-monitoring systems and connected to EO data archives (Verbesselt et al. 2010a; Forkel et al. 2013; De Jong et al. 2013).

With new satellites being launched regularly and the increasing amount of EO data that is based on open data licenses, big data technologies have been introduced into the geospatial domain (Birkin 2013). As the use of EO data in research and industry has increased tremendously (Ma et al. 2015; Nativi et al. 2015), new technologies need to be considered for handling EO time-series data. With the increasing size and number of EO data archives, the processing of these data needs to be adjusted in order to lead to automated workflows, distributed processing, parallelization, and scalable cloud-based infrastructure (Yang et al. 2017a). Data cubes and web-based processing systems have recently been combined with large EO data archives, enabling users to conduct processing in close proximity to data archives (Pagani & Trani 2018; Yang et al. 2017b).

The geospatial web has evolved rapidly in recent years, allowing the development of interactive web-based tools, applications for mobile devices, and web services (Veenendaal et al. 2017). Brokering systems and data cubes have been introduced to reduce limiting factors in relation to data discovery, access, and processing. Web-based tools can be developed with commonly used scripting languages without the need to conduct web development (Swain et al. 2016; Yin et al. 2017).

However, there are many challenges in terms of the discovery, access, and analysis of EO time-series data that need to be improved: Data discovery and access of EO time-series data still involve complex data processing tasks. For further analysis, EO data needs to be prepared in order to be used in geographic information systems and analysis tools. Although geospatial standards are available to provide discovery of and access to EO time-series data (Nativi & Bigagli 2009), users need to learn many specifications to handle the data provided by distributed environments. To answer a question such as “Was my area of interest affected by floods in recent years?” the user has to find EO data for the relevant area and conduct downloading and processing steps for each date. Many of these technical challenges can be resolved and automated. Access to EO data and analysis tools must be simplified in order to make it more user-friendly. Thus, in this thesis, new approaches for providing user-aligned and standardized web services, which enable a simplified exploration of EO time-series data, are investigated.

This thesis is structured as follows (also see Figure 1.1): An introduction is provided in this chapter, including the motivation for this study, the history of digital EO, the current state of research, and scientific questions. A scientific overview that includes definitions and requirements, the state of the art, and a review of EO web services, tools, and platforms is presented in Chapters 2–4. The case study, user personas, and user requirements are described in Chapter 2. The state of the art (Chapter 3) focuses on web technologies and EO time-series data services and formats. These are evaluated according to the user requirements. An overall review is conducted in Chapter 4, which describes how data providers offer web services and what kinds of cloud-based EO platforms are available. The own research, developments, and results are described in Chapters 5–7, including the concepts and methods, example use cases, and the results. Methodological concepts and methods are defined and grouped into three fields which are described in Chapter 5: 1) service-based EO time-series data middleware, 2) service brokering for multi-source data discovery and access, and 3) unified EO time-series data structure and analysis. These methods have been implemented in several applications, which are described in Chapter 6. The results are summarized and discussed in Chapter 7. Finally, Chapter 8 closes the thesis with responses to the research questions, conclusions, and outlook.

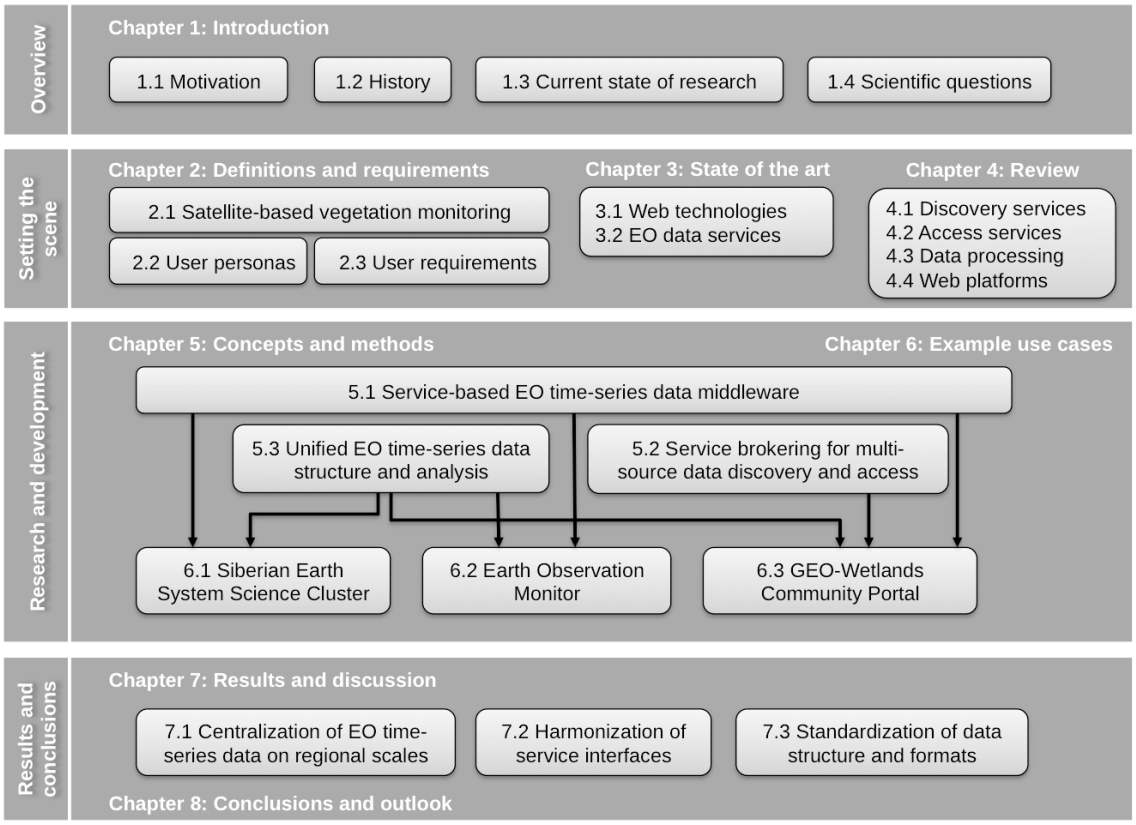


Figure 1.1: Overall structure of the thesis.

1.1 Motivation

For both the increasing amount of freely available EO data and the increasing demand for ready-for-use information, new technologies need to be researched to improve data discovery, access, and analysis. Although most of the data can already be accessed through the World Wide Web, there remain limiting factors to making such data discoverable and available with user-aligned and automated approaches. To benefit from increasing data availability, a lowering of barriers to discovering, accessing, and analyzing EO data is required. Scientists would then be able to focus on the use of analysis tools and the interpretation of the results of analysis. Thematic experts, students, and citizens, who mostly do not have a deep knowledge of geospatial data processing and web services, would be able to make use of modern web and mobile applications with a direct visualization of EO data and the results of analysis.

In recent years, the World Wide Web has been transformed from presenting only static information to including dynamic applications and web services. Web technologies have been improved with the further development of geospatial web and mobile applications based on web services (Veenendaal et al. 2017). Combined with the processing capabilities of centralized server infrastructures, EO data archives can be made available using user-aligned services and applications. With the use of web services, the complex

processing of data in general can be hidden from users. There is an increasing demand for the development and provision of web-based applications and services to support scientists, thematic experts, stakeholders, decision-makers, and citizens with information about the changing environment. Applications for environmental subjects, such as deforestation, wetland observation, vegetation change, and disaster management, can be based on EO data and provide useful information about environmental changes. To ensure the easy use of EO data for all kind of users, applications need to be developed by means of user-aligned approaches.

Providing geospatial data has been an important task in recent years. Several initiatives, such as the Infrastructure for Spatial Information in Europe (INSPIRE; European Commission 2018), have commenced to ensure that official geospatial data is being made available in a standardized manner. The Group on Earth Observation (GEO) published its Global Earth Observation System of Systems (GEOSS; Shibasaki & Pearlman 2008) to provide a centralized entry point for EO data. Thus far, it comprises a vast range of metadata for geospatial and EO data in a metadata catalogue. Within GEOSS, data access, especially for time-series data, remains behind the possibilities of current web technologies. As many data providers make their data publicly available, everyone is able to access this data, derive information, and distribute it to other users. However, in most cases, users need to have knowledge of data handling and processing in order to provide data and derived information. Although interoperability has been established with standardized specifications, the focus has primarily been on machine-to-machine interaction. Nevertheless, with many existing tools, the requirements of multiple user personas (e.g., scientists, thematic experts, stakeholders) have not yet been fully considered. Domain experts, decision-makers, and general users are not familiar with data-processing techniques and always need a considerable amount of time to understand how they can retrieve and process EO data.

Today, many EO data archives are available on the World Wide Web and can be accessed free of charge. Metadata catalogues exist for searching and finding data, which, in many cases, is published as web services to be integrated in applications. Web portals are available to search, find, and download data. In addition, web-based tools are available for analyzing datasets. However, most of the tools are standalone software and cannot be used or integrated in other applications. Such integrations are necessary to fulfill the requirements of different users and to allow them to build their own applications. The tools available often focus only on specific user personas and are not flexible enough to support multiple personas. Integration in applications and the standardized provision of these tools in particular need therefore to be considered in current research.

1.2 History

In the context of a digitizing Earth, former vice-president of the United Sates Al Gore made a clear statement in 1998 about how to proceed with handling spatial data on environmental change:

> *A new wave of technological innovation is allowing us to capture, store, process and display an unprecedented amount of information about our planet and a wide variety of environmental and cultural phenomena. Much of this information will be geo-referenced […] I believe we need a Digital Earth. A multi-resolution, three-dimensional representation of the planet, into which we can embed vast quantities of geo-referenced data.* (Gore 1998, p. 89)

Several international actions followed, such as measuring the Earth in the course of digital elevation model datasets, the launch of operational satellites to monitor the Earth (e.g., NASA's Terra and Aqua satellites), the development of international and national spatial data infrastructures, and the establishment of sensor networks.

A review of geospatial web mapping is provided by Veenendaal et al. (2017), starting with the beginning of the World Wide Web in 1989. Twenty-eight years of history is split into nine web-mapping eras: static, dynamic, services, interactive, collaborative, digital globe, mobile, cloud, and intelligent (Figure 1.2). Plewe (2007) and Tsou (2011) describe five generations of web mapping, from static through dynamic, interactive, virtual globes, to cloud computing. Both studies conclude that there needs to be a greater focus on the needs of users and greater engagement of users in designing web-based applications and services. Further directions for web mapping are towards the intelligent use of data for knowledge generation for diverse users and applications, in addition to the generation of better information and services focused and filtered to their needs (Veenendaal et al. 2017).

In the EO domain, the increasing availability of EO satellites fosters the analysis of environmental change on Earth. The recently launched Sentinel satellites of the European Copernicus program come with a free and open data policy and a wide range of new satellites and data products. In addition, the extensive archives of the National Aeronautics and Space Administration (NASA) and the United States Geological Survey (USGS) are open to the public and widely used in research, and the satellites of these organizations are still operational, with up-to-date datasets. With such free and open data policies, many datasets are available to be used in research and decision-making processes related to environmental and climate change. The demand of information about the changing

Earth—such as that pertaining to climate change, vegetation change, and disaster management—is increasing. EO data can be used to fulfill many of these demands. The recent era of EO-based web mapping comprises cloud-based processing (Evangelidis et al. 2014), cloud-optimized data formats (COG 2018), data cube paradigms (Baumann & Rossi 2016; Strobl et al. 2017), containerized data processing (Tao et al. 2018), raster processing in browsers (EOX IT Services GmbH 2018), and scalable web services and tools, which allow the generation of global thematically relevant products (Gorelick et al. 2017).

Several projects, such as the Future Earth initiative, require transferring EO-based information on environmental and climate change into action to deal with societal challenges. Future Earth, which was launched in 2012, was founded by international research funding organizations with the aim of supporting interdisciplinary collaboration in the area of global environmental change research and to address critical questions. In addition to questions regarding how the Earth is changing and how knowledge in this area can be used to move towards a sustainable future, an important “cross-cutting capability” is the focus on data and observing systems. This aims to make “the research more useful and accessible for decision makers” and to make it “accessible to all parties” (Future Earth 2013, p. 12,21). This program seeks to determine the best practices for integrating user needs and for understanding research needs that are to be fostered by “developing and diffusing useful tools for applying knowledge” (Future Earth 2013, p. 21).

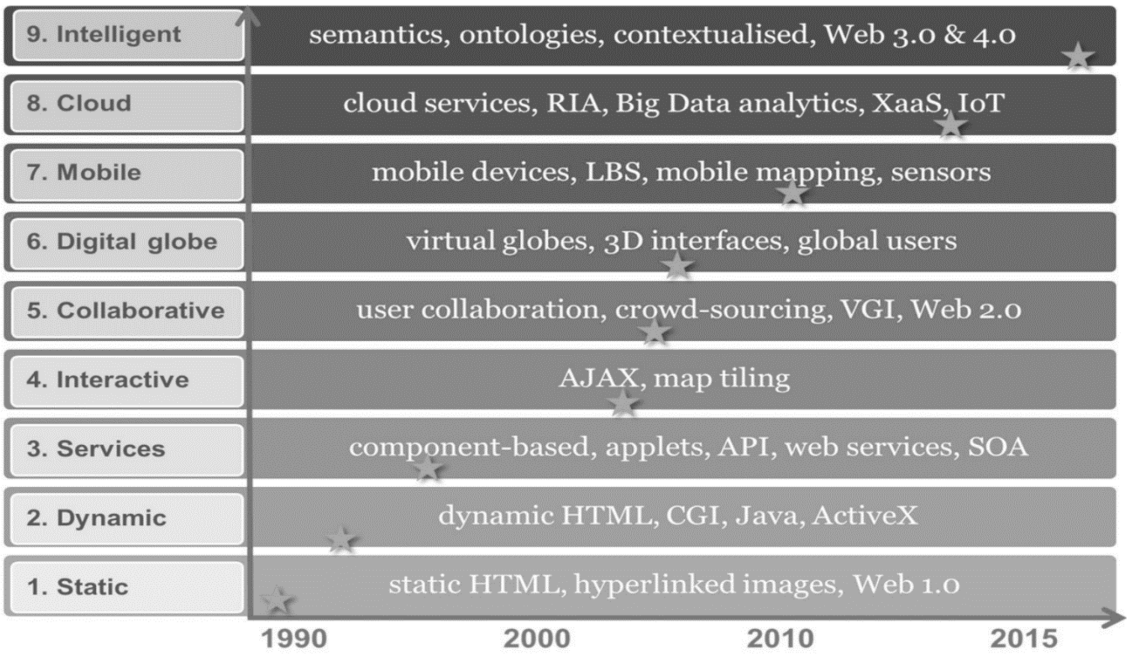


Figure 1.2: Framework of web-mapping eras. The stars indicate the approximate commencement of the era (Veenendaal et al. 2017, p. 7).

1.3 Current State of Research

Current research agendas and visions of a Digital Earth show trends in various topics, such as spatial data infrastructures (Adams & Gahegan 2014; Bernard et al. 2014; Granell et al. 2016), citizen science and crowd-sourced data (See et al. 2016; Higgins et al. 2016), big data and cloud computing (Schade 2015; Guo et al. 2017; Koubarakis et al. 2017; Boulton 2018), geospatial web services (Vinhas et al. 2016; Wagemann et al. 2018), the semantic web (C. B. Tan et al. 2017; Janowicz & Hitzler 2017), and event-driven infrastructures (Yue et al. 2015; Zhang et al. 2017; Rieke et al. 2018).

Spatial data infrastructures (SDIs) serve as fundamental systems for publishing geospatial datasets and related meta-information. Research agendas have covered the needs and further development of SDIs: Díaz et al. (2012) provide a comprehensive overview of the state of SDIs with the conclusion that "SDIs have so far failed to achieve the desired level of impact and penetration in the geospatial community" (Díaz et al. 2012, p. 380). Adams and Gahegan (2014) see next-generation SDI becoming one that acts as mediator, harmonizing data generated from heterogeneous sources. Tsinaraki and Schade (2016) discuss the implications of big data on SDIs. They conclude that although existing technologies can still be used, the growing user base needs to be better served. In particular, more users lacking knowledge of geospatial fields are expected. They tend to require service interfaces that are ready to be used and, thus, new interface specifications need to simplify the use of the current services for application developers.

Standard-compliant web technologies have been used and studied in Earth system science for many years now. Service-oriented architectures were provided in the early years of the Internet. Today, there are several discussions of the use of web technologies and web service methods (Vitolo et al. 2015; Baresi et al. 2016; Wagemann et al. 2018). The provision of interoperable web services (Nativi et al. 2012; Miura 2016) for data discovery, visualization, and access is an inherent component of an SDI. While there were early approaches that involved standard-compliant visualization and discovery of geospatial data, services for data access and data processing have improved in recent years (C. B. Tan et al. 2017; Hempelmann et al. 2018; Herle & Blankenbach 2018; Wagemann et al. 2018). Interoperability for the EO domain specifically has been investigated by Mazzetti and Nativi (2012), who state three principles: 1) build on existing capacities, 2) address different interoperability levels, and 3) lower the entry barriers for both users and providers. Recent activities in the EO domain investigate the provision of spatial data as time-series services. EO data can be provided either with the Web Coverage Processing Service query language (Baumann 2009b; Karantzalos et al. 2015) of the Open Geospatial Consortium (OGC) or using the OGC Sensor Observation Service,

which was originally designed for in-situ measurements, for raster data (Sorg & Kunkel 2015). When geospatial processes are made available using the standard-compliant OGC Web Processing Service specification, several scientific aims can be investigated, such as process orchestration (De Jesus et al. 2012; Kiehle et al. 2007), process mediation (Giuliani et al. 2012), distributed processing (Bychkov et al. 2015; Tan et al. 2015), and shared geoprocessing logic (Müller et al. 2013).

The challenges and opportunities associated with big data in the geospatial domain have been discussed regularly in recent years (Lee & Kang 2015; Li et al. 2016; Guo et al. 2017; Boulton 2018). In general, challenges related to the organization of data (Kiemle et al. 2016), big data analytics (Kambatla et al. 2014), and the use thereof combined with web technologies (Vitolo et al. 2015) still need to be further investigated. Closely related to this is the introduction of paradigms, software, and standards relating to data cubes. Baumann and Rossi (2016) define "datacubes as a service paradigm," concluding that they "are a convenient model for presenting users with a simple, consolidated view on the massive amount of data files gathered" (Baumann & Rossi 2016, p. 188). The "Datacube Manifesto" (Baumann 2017) defines a data cube as a "multi-dimensional array" and presents the requirements that lead to services that are "user-friendly, faster, and [more] scalable than typical classical services" (Baumann 2017, p. 2). Strobl et al. (2017) describe the "six faces of data cubes," which are defined as the technical aspects that are required to allow data ingestion, storage, provision, and the analysis of structured geospatial data within a geospatial data cube (e.g., data organization, data processing levels defined as analysis-ready data, infrastructure, user interfaces, and interoperability). Pagani and Trani (2018) compare traditional and data cube approaches for geospatial computation, concluding that "offering a ready-to-use data cube as-a-service might be a great value and save resources avoiding multiple storage of data, eliminate the creation of multiple ad hoc ways to access data, and make data easier to access" (Pagani & Trani 2018, p. 298).

Although there are standards for geospatial data, data management issues need to be investigated especially for raster time-series data in order for them to be provided in a user-aligned architecture. Big EO data management has recently been researched by different organizations (Kiemle et al. 2016; Xie & Li 2016; Z. Tan et al. 2017). The term "analysis-ready data" is defined by the Committee of Earth Observation Satellites (CEOS) as "satellite data that have been processed to a minimum set of requirements and organized into a form that allows for immediate analysis with a minimum of additional user effort" (CEOS 2018a). Analysis-ready data have been applied and researched in the EO domain by several researchers (Davis et al. 2015; Wyborn & Evans 2015; Szuba et al. 2016; Giuliani et al. 2017; Dwyer et al. 2018).

1.4 Scientific Questions

Based on the motivation and current state of research described in the previous sections, three scientific questions with the overall objectives of investigating the use of web technologies for EO time-series data exploration and bridging the gap between user applications and EO data archives (Figure 1.3) are posed and answered in this thesis.

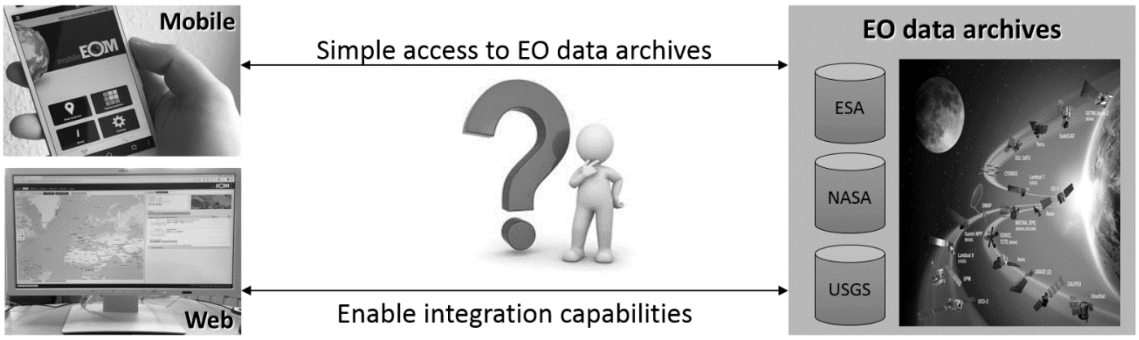


Figure 1.3: Overall objective: Bridging the gap between user applications (left) and EO data archives (right) (Image on the right: Courtesy NASA/JPL-Caltech).

1. **How to design a user-aligned discovery, access, and analysis for EO time-series data based on standard-compliant web services?**

The amount of EO data that is available for environmental analyses increases daily. New satellites are continually being launched and various users are interested in using EO data. With the increasing size of EO data archives, today especially for Landsat and Sentinel satellites, data downloads in the near future to local computers will take too much time and storage capacity to be practicable. Solutions are required that provide a simple method to combine data discovery, access, and analysis. Although much EO data is available at no cost, the discovery, access, and processing thereof need to be simplified to meet the requirements of different users (e.g., scientists, thematic experts, and developers). Thus, increasing user demands and data availability need to be brought together, leading to the design of user-aligned web services for EO time-series data exploration. In this thesis, system architectures and service specifications are explored and investigated based on the use of standardized web technologies in order to foster this new behavior.

2. **What are the technical requirements for accessing and processing multi-source EO time-series data?**

Much of the EO data is available in a variety of data formats and data structures. Users always need to work with multiple data formats and need to prepare data for their analysis tools. Working with EO data from different satellite missions and data providers, in particular, requires knowledge of different data formats and of data processing in general.

The processing and analysis of EO time-series data requires further knowledge of automated data processing workflows to conform with large EO time-series data archives. Accessing EO time-series data involves a complex workflow because different processing steps need to be undertaken. Not only is the EO data provided in different data formats, but each data provider also uses different technologies and specifications to provide the data (e.g., protocols, such as HyperText Transfer Protocol and File Transfer Protocol or specifications, such as Open Data Protocol and OGC Web Coverage Service). Any one of these differences need to be considered when discovering and accessing EO time-series data. To lower this barrier, a common, easy-to-use specification needs to be investigated to make EO time-series data more easily discoverable and accessible within a standardized approach. A uniform data format and the uniform handling of time-series data are necessary and need to be directly linked to use in analysis tools.

3. What are the data requirements for analysis- and application-ready formats and how must EO time-series data hence be organized?

In these times of large EO data archives, analysis tools need to be moved to the data, rather than EO data being downloaded to local computers. Although data and analysis tools can be used together, in most cases, further data processing steps need to be undertaken before data analysis can be conducted. In recent research, the new approach of taking algorithms or analysis tools to data has been defined as “moving code” (Müller et al. 2010, 2013); however, so far, this has focused on general tools within geographic information systems. The analysis of EO time-series data requires that tools be moved and deployed close to the data. This allows users to conduct analyses without the need to download data and install software. However, within the EO domain, EO time-series data need to be available in a data format and structure that can be handled by each analysis tool. Thus, the moving code approach needs to be connected to EO time-series data. In addition, the results of analysis need to be further processed to meet the needs of users (e.g., direct visualization of the results of analysis). As EO time-series data needs to be prepared for the use of analysis tools, a direct link between these and access to time-series data needs to be established to enable users to employ analysis tools without prior data download and preparation.

Chapter 2: Definitions and Requirements

The overall framework of this thesis is defined and described by means of a thematic case study, specific user personas, and specific functional and technical requirements. The case study on satellite-based vegetation monitoring includes a description of EO data and time-series analysis tools. Three different scientific algorithms that have been linked to the EO time-series data are described. In order to focus on user-specific developments, three user personas are defined and the requirements for web platforms, web technology, and data formats are presented.

In the sections that follow, the following definitions and requirements are presented:

- Case Study: satellite-based vegetation monitoring (Section 2.1)
- Definition of user personas (Section 2.2)
- Description of functional and technical user requirements (Section 2.3)

2.1 Case Study: Satellite-based Vegetation Monitoring

Climate-induced and anthropogenically induced changes in land cover and vegetation—in particular, gradual changes in vegetation—can be identified based on EO time-series data in order to analyze numerous land change processes, such as pan-arctic climate and land cover trends (Urban et al. 2014), trends of light use and inherent water use efficiency in African vegetation sensitive to climate and atmospheric carbon dioxide concentrations (Traore et al. 2014), the detection of indicators of change processes in forest ecosystems (Hüttich et al. 2007), and the derivation of the phenological metrics of vegetation types (Jönsson & Eklundh 2004). Near real-time change detection—for example, to detect deforestation or other events—are also based on EO time-series data (Verbesselt et al. 2012; Verbesselt et al. 2010a; DeVries et al. 2015; Dutrieux et al. 2015).

Indices for vegetation monitoring that use optical EO data are based on radiometric measurements of photosynthetically active radiation in the leaves of the vegetation. The index is based on different reflectance in red and near-infrared bands in the optical spectrum. These act as a proxy for photosynthetic activity and the vitality of plants. Time-series of vegetation indices, such as the Normalized Difference Vegetation Index (NDVI; Tucker, 1979) and the Enhanced Vegetation Index (EVI; Huete et al., 2002), have proven to be important data sources for vegetation change and dynamics analyses. Figure 2.1 shows an example of a regularly observed EVI for the “Stadtbruch” in Anklam, Germany, from 2000 to 2016. A change of the vegetation index over time, which is linked to renaturation (flooding of former moor area), can be identified from 2010 onwards.

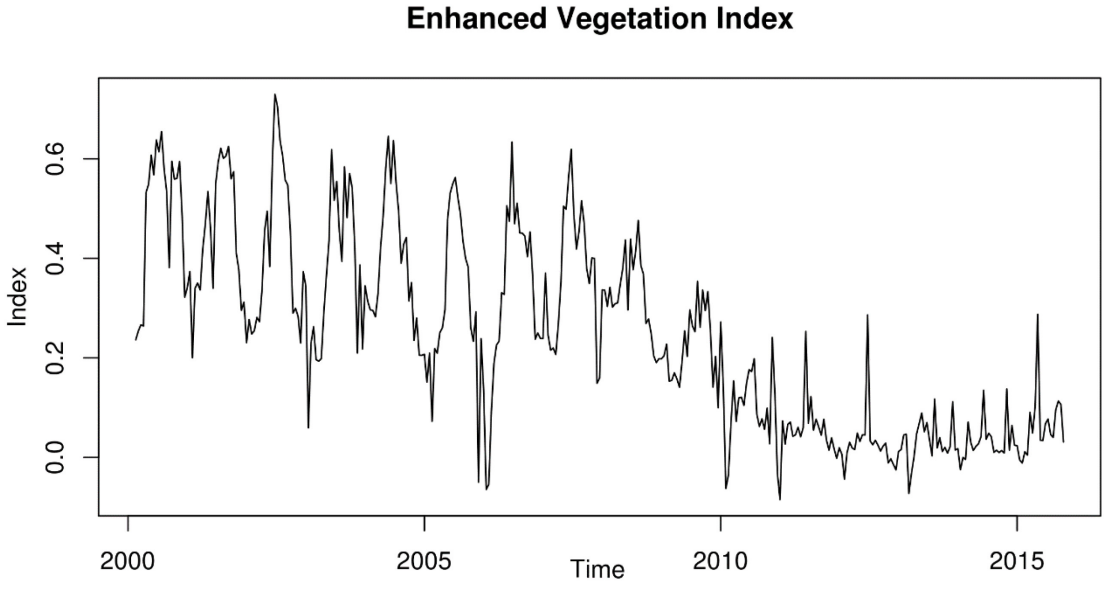


Figure 2.1: Multi-annual time-series of MODIS EVI of Stadtbruch in Anklam (Germany).

2.1.1 Earth Observation data

A number of EO satellites exist, which provide free global time-series data in different spatial and temporal resolutions. For vegetation analyses, satellites with optical sensors are widely used and further derived vegetation datasets are provided automatically. For example, NASA's Moderate Resolution Imaging Spectroradiometer (MODIS) provides daily to bi-weekly global data with a spatial resolution of 250 m to 1 km (Huete et al. 2002). High-resolution USGS Landsat data (30 m spatial resolution) has also been used to study changes in the vegetation (Zhu 2017), though the temporal resolution of only 16 days repeating rate often leads to gaps in the time-series because of cloud cover. Sentinel-2 of the European Space Agency (ESA) data provides up to 20 m spatial resolution and up to 6 days repeating rate from two satellites in orbit (Sentinel-2 A and B). It thus provides a better temporal resolution than Landsat though the data has only been available since 2016. Vegetation index datasets are available from several other satellites, such as SPOT-Vegetation, Proba-V, and the Advanced Very High Resolution Radiometer (Xie et al. 2008). In what follows, MODIS, Landsat, and Sentinel data, which have been used in this thesis, are described.

MODIS

Data from MODIS is received from two satellites, Aqua and Terra. Both of these were launched through NASA's Earth Observation System program. Continuous observations for Terra have been available since 2000 and for Aqua since 2002. The aim of the MODIS sensor is to monitor the land surface, oceans, and atmosphere. Data from MODIS is provided as land, atmospheric, and ocean products on a systematic basis (Justice et al. 2002). These standard products contain information about atmospheric profiles, surface reflectance, clouds, land and sea surface temperature, thermal anomalies, vegetation indices, snow cover, sea-ice cover, and the like (Xiong et al. 2009). The land products are available in several spatial (0.05 deg, 1 km, 500 m, and 250 m) and temporal (monthly, 16-day, eight-day, four-day, and daily) resolutions.

Data is provided in the Hierarchical Data Format-Earth Observing System (HDF-EOS). Based on the spatial resolution, data is available as 5 minute swaths, as tiles with a width and height of 10 degrees (tile grid in Figure 2.2), and as a Climate Modeling Grid (CMG) (Wolfe et al. 1998). Tiles have a sinusoidal projection; the CMG is based on the World Geodetic System (WGS) 84 / Lat-Long projection. A detailed quality layer is available for each standard product from MODIS. In the vegetation index product (MOD13), a "pixel reliability" layer summarizes the information quality with ranked values describing the overall quality of each pixel.

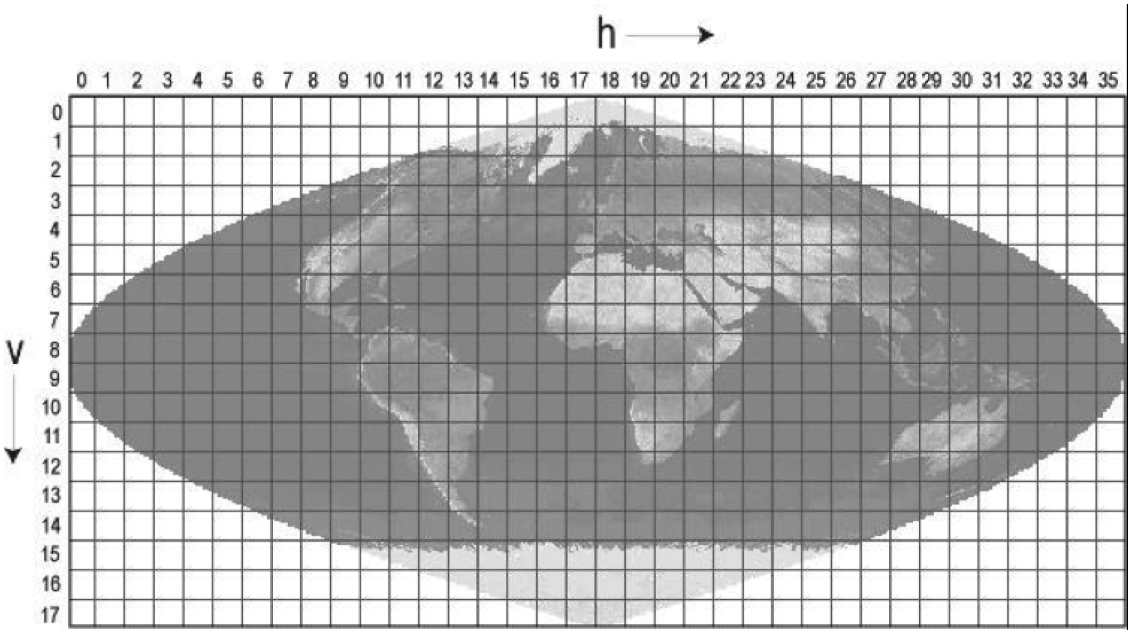


Figure 2.2: MODIS sinusoidal tile grid (NASA LPDAAC 2014).

The NASA Land Processes Distributed Active Archive Center (LPDAAC) is one of the official sources for obtaining MODIS data produced by the MODIS Land team (NASA 2019b). The data is provided in the original HDF-EOS data format on the file servers[1]. Users need to know which specific MODIS tile they need for their area of interest. The HDF-EOS file includes several layers that need to be extracted for further processing. To reduce storage space, individual layers with floating-point values have been stored as integers and need to be scaled to the original data range in the processing chain. Others, like the detailed quality flag, are encoded as integers and need to be decoded bit-wise. Thus, the layer name, units, bit type, no data (fill) value, valid data range, and scale factor need to be known for further data processing.

In addition to the data being provided by NASA LPAAC, MODIS data is also available from other data providers. The Distributed Active Archive Center at Oak Ridge National Laboratory (ORNL DAAC) provides a MODIS Web Service[2] based on various web technologies, which support several request methods, such as lists available products and the dates the products will be available, subset extraction, and subset order. Google Earth Engine[3] provides access to several MODIS Level 3 products, such as a 16-day vegetation index. This data can be directly processed within Google Earth Engine (e.g., quality masking, clipping to the area of interest, and individual calculations) and either further analyzed or directly downloaded by users.

[1] https://e4ftl01.cr.usgs.gov
[2] https://daac.ornl.gov/modiswebservice/
[3] https://earthengine.google.com

Landsat

The Landsat program of the United States of America (USA) was established in 1966, and the first satellite was launched in 1972. Landsat–8, launched in 2013, is the latest satellite from this program. A graphic overview of all Landsat satellites is shown in Figure 2.3. In contrast to MODIS, Landsat satellites provide data with up to 30 m spatial resolution but with a lower temporal resolution (16-day global coverage). Since 1972, Landsat satellites have used different sensors, starting with the Multi-Spectral Sensor (MSS) on Landsat 1–3, followed by the Thematic Mapper (TM) on Landsat 4–5, the Enhanced Thematic Mapper Plus (ETM+) on Landsat–7, and the Operational Land Imager (OLI) with the Thermal InfraRed Sensor (TIRS) on Landsat–8 (USGS 2019c).

Landsat data is stored as tiles within the Worldwide Reference System (WRS), a global notation system for Landsat data (Figure 2.4). Each tile of the WRS consists of a center coordinate as well as path and row numbers. Each Landsat scene can be identified using these numbers. Based on the Level–1 processing, different "Landsat Collection 1 Tiers" are available: Tier 1 has the highest available data quality, including Precision and Terrain corrected data; Tier 2, which does not meet the data quality criteria from Tier 1, includes Systematic Terrain and Systematic processed data and real time data with additional post-processing (USGS 2019b). Beyond this standard processing to Level–1, the Level–2 product, "Surface Reflectance," is processed, which includes atmospheric corrections. Level–3 products (e.g., surface water extent, burned area) are currently under development. Landsat files are available as a zipped archive containing files for each band. All Landsat data is provided in GeoTIFF data format, with a Universal Transverse Mercator (UTM) map projection and a WGS-84 datum (USGS 2019b). The NDVI product needs to be calculated separately using the red and near-infrared bands once the Landsat data has been downloaded.

The web-based Earth Explorer[4] application from the USGS is the first site that can be used to search for and order Landsat images. In addition, other applications for EO data searching and visualization (e.g., NASA Earthdata and Landsat Look) link to the Earth Explorer when providing data download functions. In addition to data from the USGS, collections from the NASA LPDAAC and a range of further datasets are available in the Earth Explorer. With a user account, data can be downloaded directly or added to a cart, which can be downloaded later with the bulk downloader tool. The USGS Earth Resources Observation and Science Center provides a new service-based data access system[5]

[4] https://earthexplorer.usgs.gov
[5] https://espa.cr.usgs.gov

(USGS ESPA). It provides both a simple web-based interface and a service interface to be used with programming languages. In addition to data access to the Landsat TM, ETM+, and OLI sensors, processing tools can be applied automatically before downloading the datasets. These processing tools include re-projection, spatial sub-setting, and pixel resizing as well as various thematic products (e.g., top of atmosphere reflectance, brightness temperature, cloud mask, and surface reflectance) and output formats (e.g., GeoTIFF, ENVI binary, NetCDF, HDF-EOS2). Users of ESPA need to know the Landsat scene identities prior to using the tool. An order first needs to be placed, and the resulting data can be downloaded once the processing is done. Landsat 4 to 8 data have been made available on the Google Cloud infrastructure[6] and on Google Earth Engine. When using Google Cloud, the individual scenes can be directly accessed on cloud-based virtual machines. Amazon Web Services[7] provides access to Landsat 8 data, which are already unzipped and accessible by band.

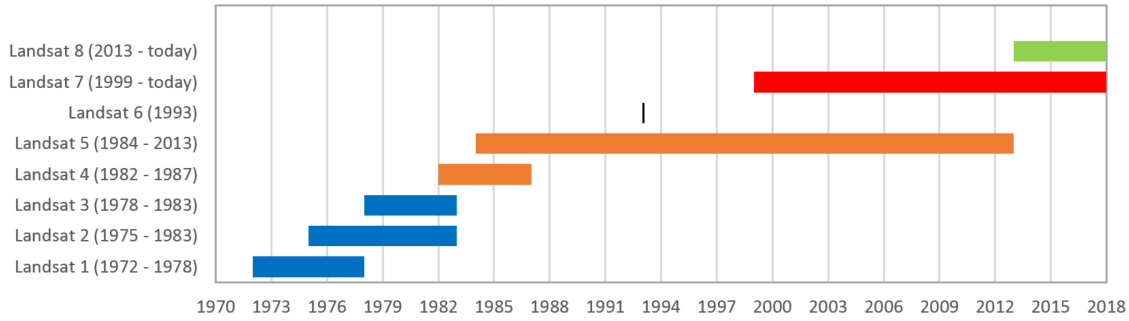


Figure 2.3: Timeline of the satellites of the Landsat program: Landsat MSS (blue), Landsat TM (orange), Landsat ETM+ (red), Landsat OLI (green). Visualization based on USGS (2019b).

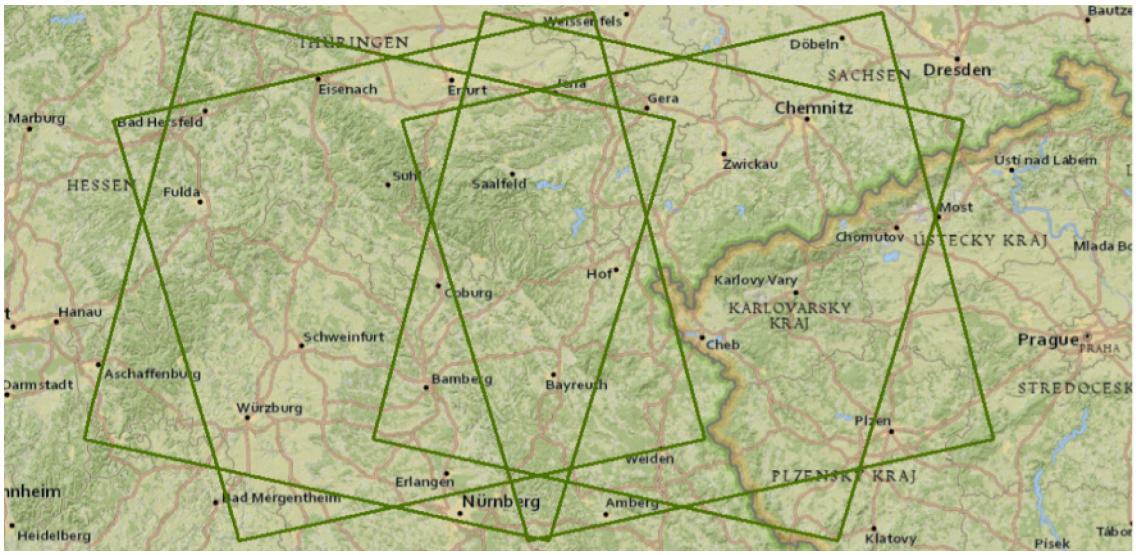


Figure 2.4: WRS–2 tiles (ascending and descending, green rectangles) for parts of Thuringia, Germany (own visualization, National Geographic World map as background).

[6] https://cloud.google.com/storage/docs/public-datasets/landsat
[7] https://aws.amazon.com/de/public-data-sets/landsat/

Sentinel

The Copernicus program of the European Union (EU) has launched Sentinel satellites since 2014. These provide EO data from a variety of sensors for numerous thematic purposes (Aschbacher & Milagro-Pérez 2012; Berger et al. 2012). Table 2.1 lists the first three Sentinel satellite missions, along with their sensor types, spatial and temporal resolution, and launch dates. The Copernicus program provides satellites with high (10 m) and medium (300 m) spatial resolution. The temporal resolution ranges from daily to 6-day. As this is a long-term satellite mission, the European Commission has already been ordered two more identical satellites for Sentinel 1, 2, and 3 (ESA 2016; Magan 2016).

The satellite data from the Copernicus program is made available through an open data license (Aschbacher & Milagro-Pérez 2012). The data is available from the official Copernicus Open Access Hub[8] (SciHub) provided by ESA, national ground segments (e.g., the German Copernicus Data and Exploitation Platform), and cloud-based data providers (e.g., Amazon Web Services, Google Cloud, Google Earth Engine, and Copernicus Data and Information Access Services (DIAS)). The ESA SciHub can be used with a web portal or with standardized web service interfaces. Depending on the EO mission, the data is available in various formats, such as netCDF, GeoTIFF, and JPEG2000.

From Sentinel–2, the vegetation index can be calculated using red and near-infrared bands. Specific vegetation data is provided by Sentinel–3[9]. The Ocean and Land Colour Instrument provides a Level–2 Land Full Resolution (300 m) product, which includes the Fraction of Absorbed Photosynthetically Active Radiation as Global Vegetation Index. The NDVI can be calculated individually using the red and near-infrared bands. The Sea and Land Surface Temperature Radiometer provides a land surface temperature product, which includes a pre-processed NDVI on 1 km spatial resolution.

Table 2.1: First three Sentinel missions of the EU Copernicus program (ESA 2019).

Satellite	Sensor	Resolution	Launch dates
Sentinel–1	Radar imaging mission for land and ocean services	5 m–40 m 2–6 days	3 April 2014 25 April 2016
Sentinel–2	Multispectral high-resolution imaging for land monitoring	10 m–60 m 2–5 days	23 June 2015 7 March 2017
Sentinel–3	Multi-instrument for surface temperature, land color, etc.	300 m 1–4 days	16 February 2016 25 April 2018

[8] https://scihub.copernicus.eu/
[9] https://sentinel.esa.int/web/sentinel/missions/sentinel-3/data-products

EO data formats and data structures

Much of the data from different satellites is provided with an individual data structure and data formats. Table 2.2 shows original data formats for MODIS, Landsat, and Sentinel. While data from Landsat satellites has been in the same data format since the beginning, many of the Sentinel satellites provide different data formats according to their specific user community. Thus, users need to know how to work with those data formats or at least need to convert data into other more commonly used formats. Although software exists that can handle all of these data formats (e.g., GDAL, SNAP), users need to know exactly how to handle the data with the software to be used within their individual processing workflows (e.g., within Python or R).

Table 2.2: Data formats for each of the satellite missions.

Dataset	Format
Sentinel-1	SAFE / GeoTIFF
Sentinel-2	SAFE / JPEG2000
Sentinel-3	netCDF
Landsat 1-8	GeoTIFF
MODIS (Level-3 products)	HDF-EOS

Figure 2.5 shows the data structures for Sentinel, Landsat, and MODIS Level-3 data: MODIS Level-3 data is provided within a single file including the measurements as bands accompanied with additional files for metadata and quick-look image. Sentinel and Landsat data is originally provided as zipped archive file. Measurements from Sentinel and Landsat data is provided in individual files within the archive file. Additionally, files for metadata, quick-look images, etc. are available.

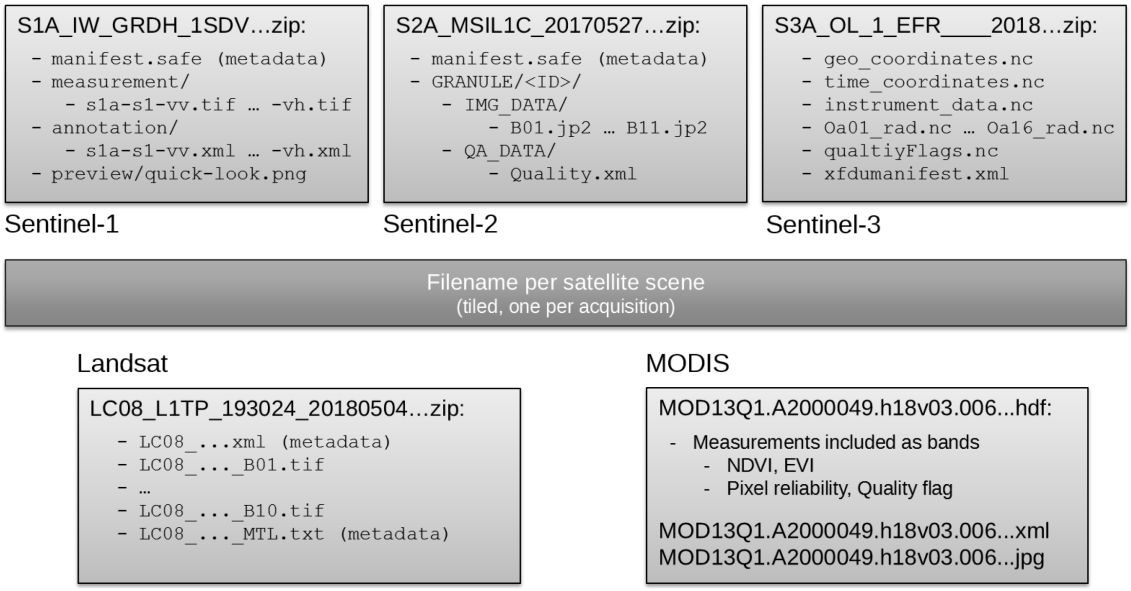


Figure 2.5: EO data structures for individual Sentinel, Landsat, and MODIS satellite scenes.

2.1.2 Analysis tools

Evolving methods of vegetation change characterization include breakpoint detection techniques that allow users to detect the point in time when a change event occurred. Studies observing the increasing or declining net primary productivity of vegetation (Verbesselt et al. 2010a; Verbesselt et al. 2010b) have fostered the development of statistical tools for detecting trends such as greening or browning. Three analysis tools based on vegetation time-series data are used in this thesis: Greenbrown for trend calculations, BFAST for breakpoint detection, and TIMESAT for phenological analyses.

Greenbrown

The 'Greenbrown' software (Forkel et al. 2013) is a collection of functions designed to analyze trends and trend changes in a gridded time-series, as with those from satellite observations or climate model simulations. In the EO community, vegetation greenness can be divided into positive (greening) and negative (browning) trends. The Greenbrown methods are provided as software packages in the R statistical language. The methods can be used within R, based on time-series objects. In addition to trend calculations for individual pixels (Figure 2.6, left), it is possible to derive greening and browning classifications for raster time-series data (Figure 2.6, right). Analyzing the increasing and decreasing trends of the vegetation can help to identify regions of long-term vegetation change. This can be demonstrated in the area of Anklam, where a former moor area was recultivated following controlled water logging (Figure 2.6). This results in a decreasing vegetation index, which can be analyzed using 'Greenbrown.' In addition, the significance of the trends is available as result and can be used to classify trends into greening and browning.

BFAST

The 'Breaks For Additive Seasonal and Trend' (BFAST) method (Verbesselt et al. 2010a) allows the identification of changes in land cover by detecting phenological changes in the inter-annual time-series. The BFAST software integrates the decomposition of time-series into trend, seasonal, and remainder components and provides the times and number of changes in the time-series (Figure 2.7, left). This method has been widely used to detect deforestation and other disturbances (e.g., fire and floods) based on MODIS Vegetation and Landsat data. BFAST is available as a method in the R statistical language. As with Greenbrown, the methods are based on R time-series objects. In contrast to Greenbrown though, only methods for single pixels are available; however, they can be iterated over a spatial time-series object using standard functions in R. Other tools, such as '*bfastSpatial*' (Dutrieux & DeVries 2014) based on BFAST, have been published to overcome this issue, but have not been used in this thesis.

Figure 2.7 shows an example output of BFAST, which demonstrates the advantages of using BFAST in vegetation time-series analyses. In addition to the different breaks in a trend, which have also been shown with the Greenbrown analysis, a break in the seasonality of the vegetation was identified in 2008 followed by a visual change in the vegetation index. With such breakpoint detection, the point at which changes in the time-series occur can be identified automatically. The example shows the previously flooded moor area that has been recultivated in the recent years (Stadtbruch Anklam).

TIMESAT

In addition to trend and breakpoint analyses, the derivation of phenological metrics (phenometrics) for vegetation characterization and classification is another method used to monitor changes in vegetation based on EO time-series data. TIMESAT software is used for satellite-based phenological characterizations that analyze the seasonality of EO time-series data and their relationship with the phenological cycles of vegetation (Jönsson & Eklundh 2004). It furthermore enables the extraction of time-related phenological metrics (e.g., start of season and length of season) and biomass-related metrics. The extraction of phenometrics allows the analysis of short- and long-term changes in land cover and the structural and species composition of vegetation types. TIMESAT is provided as executable on Linux and Windows operating systems. It provides several time-series fitting functions (e.g., a Savitzky-Golay filter, least-squares fitted asymmetric Gaussian, and double logistic smoothing) that are undertaken automatically before the phenological analysis is done. Time-series data need to be pre-processed to match the TIMESAT input file format. The file format is based on flat binary files for spatial analyses and text files for single pixel analyses.

The example in Figure 2.8 shows the seasonal parameters for the start, length, and end of the vegetation season calculated based on the vegetation index time-series data. Using the yearly chart (Figure 2.8, right), the change in the vegetation season can also be identified easily by non-expert users.

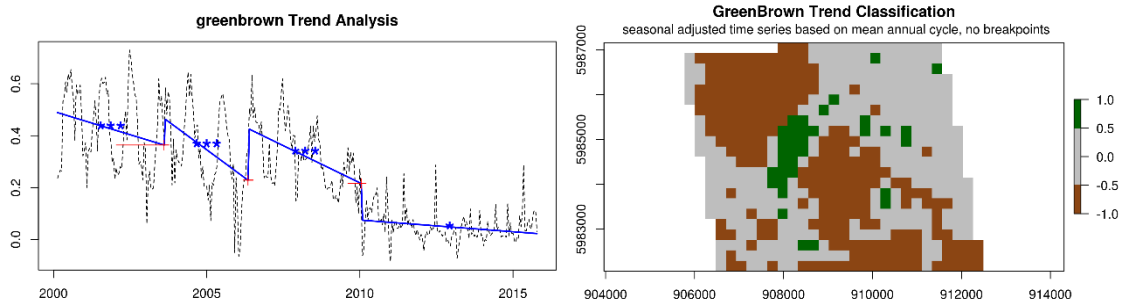


Figure 2.6: Greenbrown analyses for Stadtbruch Anklam: on the left, an analysis of a single pixel showing the vegetation observations and multiple trend calculations based on breakpoints; on the right, a spatial analysis showing the greening and browning trend classification (own visualization).

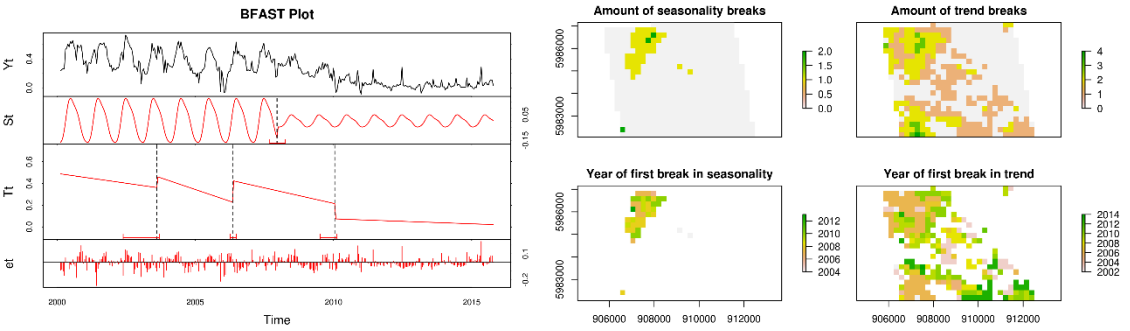


Figure 2.7: BFAST analyses for Stadtbruch Anklam: on the left, an analysis of a single pixel showing the vegetation observations (first line), the seasonality with breaks (second line), the trend with breaks (third line), and the remainder signal (last line). The four graphics on the right show a spatial analysis with number and year of the breakpoints in seasonality and trend (own visualization).

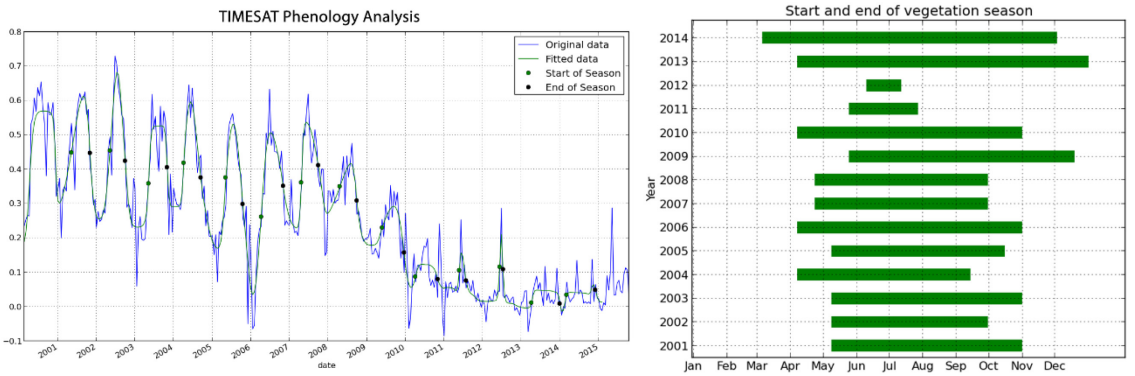


Figure 2.8: TIMESAT analyses for Stadtbruch Anklam: on the left, an analysis of a single pixel showing the vegetation observations (blue line), the fitted data (green line), and the calculated start and end of seasons (green and black points); on the right, the chart shows the different start, length, and end of the vegetation season year by year (own visualization).

2.1.3 Conclusions

For users who focus on vegetation time-series data, freely available EO data can be used in combination with scientific analysis tools to study changes in vegetation. A specific vegetation product with a coarse spatial resolution is available only from the MODIS sensor. Vegetation indices from higher resolution data (e.g., Landsat, Sentinel–2) need to be processed prior to analysis. From the point of view of an end-user (e.g., decision-maker, stakeholder, or scientist), access to ready-to-use analysis tools to derive information from EO time-series data is a relevant issue. Some of the tools that are currently available can provide information about breakpoints in trends and seasonality, which can be used as indicators of changes in vegetation. A great amount of information can also be retrieved from phenological derivations, such as the start and end of a season, the seasonal amplitude, and seasonal integral. However, to date, none of this information is available as an on-demand processing service in web-based environments.

In summary, service-based access to time-series vegetation data is available only within specific limits. Several key points can be stated after reviewing the available EO data and analysis tools for EO-based vegetation time-series data analyses:

- The analysis of vegetation time-series based on EO data is relevant to studying changes of the environment.
- Much data is freely available, but data processing is still too complex and too diverse.
- Often, analysis tools require a specific data format and data structure, which is challenging for users who are not familiar with time-series data processing.

2.2 User Personas

Various aspects need to be considered in order to provide tools for diverse kinds of users, such as scientists, thematic experts, and software developers. Specific user types can be represented as personas. User personas define “hypothetical archetypes of actual users” in order to develop a precise description of users and their goals (Cooper 2004, pp. 123–124). To conform with the different types of users in this thesis, three user personas and their objectives have been defined in the following subsections.

2.2.1 Scientists

Scientists work with EO data to develop new algorithms to derive further information. They require control over all processed data and need to be able to use analysis tools with different parameters. Scientists need to download the datasets relevant to their study area and conduct the analysis on their own computing infrastructure. Although scientists are able to download data from the data provider, various processing steps are always necessary to extract the relevant data and convert it to a data format that fits as an input to the algorithms used by the scientists (e.g., GeoTIFF format instead of HDF format). These steps can be automated and annotated by means of processing commands, which have been undertaken on the data downloaded. Using such annotations, for example, within log files, scientists are able to reproduce the processing done by an automated processing service. Thus, scientists can be supported with data retrieval, clipping to the area of interest, checking quality flags, and converting data formats. In addition, the preparation of data for specific algorithms supports scientists by allowing them to focus on algorithm parameters and the interpretation of the results of analysis.

The following key aspects for scientists have been identified:

- Easy-to-use EO time-series data discovery and access services,
- Reproducible and automated data access and extraction for a given pixel or area of interest, and
- Executable algorithms linked with automated data extraction.

2.2.2 Thematic experts

Thematic experts have knowledge of specific environmental issues and understand the general concepts of geographic information systems (GIS) and EO data. However, in most cases, they are not familiar with complex data processing and the different EO data formats. Data needs to be in common data formats (e.g., GeoTIFF or ESRI Shapefile) and prepared for direct visualization and use in common GIS software. Web and mobile applications can support thematic experts in obtaining access to EO data and derived information, as well as to the results of analysis, either in the field using mobile applications

or in the office using web applications. The post-processing of the results of analysis in order to match the needs of the thematic experts is welcomed, for example, by making data visualization and understanding easier due to familiar data formats and other tools (e.g., interactive charts and pregenerated figures).

The following key aspects for thematic experts have been identified:

- On-the-fly visualization of the results of analysis,
- Web-based visualization of thematic products,
- Interactive exploration tools for EO data and data analysis,
- Mobile devices with on-the-fly extraction of EO time-series data, and
- Post-processing of the results of analysis to match common data formats.

2.2.3 Software developers

Software developers need ready-to-use services to build web and mobile applications. In general, software developers for web and mobile applications do not have knowledge of geospatial data processing. As such, ready-to-use services need to provide all the processing steps needed by the application. In a best case scenario, these service-based infrastructures make use of standard-compliant specifications so that existing software tools can be reused by the software developer.

The following key aspects for developers have been identified:

- Service-based infrastructure for data discovery, access, extraction, and processing based on state-of-the-art web technologies,
- Process chaining capabilities to combine different services (e.g., data extraction and analysis tools), and
- Ready-to-use web services with data processing already included.

2.2.4 Summary

The following key aspects can be summarized and need to be investigated to simplify work with EO time-series data for scientists, thematic experts, and software developers:

- Discovery, access, extraction, and processing tools provided by web services,
- Interactive exploration of EO time-series data and analysis tools,
- Service-based visualization of time-series data and the results of analysis, and
- Data and algorithms need to be linked directly and used together.

2.3 Requirements

Based on the various aspects of the user personas described in the previous section, several requirements have been defined to ensure the user-aligned exploration of EO time-series data. These functional and technical requirements have been grouped into the following subsections:

1) Web platforms (Subsection 2.3.1)
2) Web technologies (Subsection 2.3.2)
3) Data formats relating to user aspects (Subsection 2.3.3)

2.3.1 Web platforms

As regards to future users of web platforms, various aspects, such as the availability of data, functionality, services, and user management, need to be considered, investigated, and defined when designing a web-based EO platform. These are described in the following paragraphs.

Data availability

The availability of geospatial data is the fundamental basis of each platform. Thus, various EO time-series and additional geospatial data need to be available in easy-to-use data formats. The platform provider should be able to integrate further data, such as time-series data from climate stations, which can be analyzed together with EO time-series data.

Functions

In terms of functionality, various tools for geospatial data are required, such as for data visualization, data download, and pixel extraction. Visualization tools allow users to easily work with data and analysis tools. Data download tools are important for allowing users to work with the data on their own computers. Users need to be able to define an area of interest or individual pixel for further exploration of the available geospatial data. Tools for the analysis of time-series data are useful for exploring the data available on the platform. As such, besides the provision of specific analysis tools, external applications and interactive development environments for different programming languages (e.g., Python and R) can be linked to the platform and made available to users.

Web services

All the functions of a platform can be presented as web services for use by external applications (e.g., web and mobile applications and desktop software). Standardized web services can be provided for data discovery, visualization, download, and analysis. These can be used by external applications, such as desktop GIS tools or other web-based applications, without any adjustments. Geospatial web services are necessary for

geospatial data in order to allow the interactive exploration of data, the provision of metadata catalogues, and for geoprocessing and analysis tools. In order to provide standardized and geospatial web services, a few requirements need to be considered when comparing software solutions:

1) Geospatial data needs to be published automatically as visualization and download services with a pre-defined style for visualization,

2) Metadata needs to be published automatically for new datasets as output from data integration and data analysis, and

3) Users working on their own areas of interest need their own metadata catalogue and geospatial data-service instance separated from services for other users.

User management

The platform stores and manages various items of information about connected data providers, available datasets, and users of the system. All this data must be made available via web services, as various applications require access to this data. In addition, user-specific data, such as areas of interest, data integration that has been undertaken, and analytic tasks, can be stored and managed.

2.3.2 Web technologies

Different aspects of technologies can be considered and evaluated when providing web services, such as their architecture, long-running processes, service chaining, standard-compliant services, and uniform web service specifications. These are described in the following paragraphs.

Web service architecture

Web services are available with various architectural styles and protocols for distributed service architectures. As such, the evaluation of web service architectures is an important component of the setup of web-based platforms.

Long-running processes

Both synchronous and asynchronous execution of web services can be considered for user-aligned web services. The difference in both of these—whether or not to wait for the result—depends on the application that makes use of the web service. When executing web services asynchronously, the client needs to regularly check the status of the process. In most cases, both execution types are relevant to the application and depend on each individual process that is being provided. Thus, client and server applications need to support both execution types. Asynchronous execution is especially important for long-

running processes, such as data analysis and large data-integration tasks. It allows users to close the application and check the status later. In addition, the failure of an Internet connection does not stop the web service executed by the user. A further issue in terms of service execution involves allowing the web service to provide access to outputs immediately, when it is ready, without waiting until all the outputs have been generated. This allows the client application to show the results as soon as they are available. For example, a time-series plot of the input data can be shown to the user although a further analysis task is still running as part of the same process request.

Service chaining

Service chaining often includes only the direct use of a web service as input from another web service. Thus, the output of the first web service is directly an input for the second. As this is useful for many purposes, with the use of a large time-series data output it is more complex to send this to the next web service. Therefore, a concept is necessary that allows subsequent web services to know where to find the data of a previously executed web service for further processing on the same server. For example, a data access request is followed by a time-series analysis request that is conducted on the data of the previously conducted access web service. In most cases, users are not only interested in obtaining access to data but also in conducting analyses. If both web services are available on the same infrastructure, discovery, access, and analysis services can be linked. Thus, the output data of the first process needs to be available for the second process. In addition, algorithms provided as web services (e.g., the breakpoint analysis service) need to know where the data has been stored and how it has been managed. If data location and structure are considered within these processes, the execution of processing workflows—from data discovery through data access to data analysis—can be realized.

Standard-compliant services

Standard-compliant web service specifications are available from the OGC. These can be used by most GIS software and geospatial web applications. In comparison to a self-developed web service specification, standardized specifications allow the use of web services from standardized programming libraries and a wide range of geospatial tools. Standards are available for various geospatial tasks, such as data visualization, discovery, download, and processing.

Uniform web service specification

Web service specifications can be diverse as different tasks are provided (e.g., data discovery, access, and analysis), and different data providers share their services with

various service specifications (e.g., self-developed vs. standard-compliant interfaces). Although standardization organizations, such as the OGC, provide different service specifications for data discovery, access, and analysis, they share common methods and data formats that are optimized for machine-to-machine communication with single output results. A uniform web service specification is envisaged with the objective of providing multiple output formats at the same time for all of the web service tasks (discovery, access, and analysis).

2.3.3 Data formats

In contrast to current web service specifications for data discovery and data access, multiple output formats are required to fulfill the needs of different user personas. This allows the optimizing of the outputs individually for each service in respect of the user personas and their requirements. Examples of output formats for data discovery, access, and analysis are provided in the following paragraphs and summarized in Table 2.3.

Data discovery

Increasing amounts of EO data and the different requirements of users are challenges when providing EO data archives. A complex task is to find suitable EO data for specific areas of interest, time, and specific parameters (e.g., cloud coverage and sensor type). Considering the user personas described in the previous section, different aspects need to be evaluated. A thematic expert may be more interested in how many satellite scenes are available in the area of interest (e.g., as provided in a figure or a summarized table), while a researcher is interested in an output file, which can be processed by any other software (e.g., by providing CSV or Shapefile formats). A developer is interested in a web service feed, which can directly act as input to further processing workflows. It is not only standardized web formats that are suitable for data discovery. In particular, additional commonly used data formats, such as those of spreadsheets or geospatial data formats increase the ability to use discovery results. Enabling discovery services to comply with the different needs of users leads to the need for multiple data formats for the resulting outputs. Table 2.3a shows the individual requirements of output formats for fostering their re-use by different user personas.

Data access and extraction

Although the principle of “algorithm to data” is advanced today, many users still download and process data on their own infrastructure. Therefore, EO data access is still an important issue for discussion and evaluation. To comply with the needs of different user personas, besides the requested EO data, it is necessary to provide different outputs (Table 2.3b), such as statistical summaries (e.g., the mean minimum and maximum

standard deviation) for each date or chart of time-series data of individual pixels. In addition, the data access service needs to provide processing capabilities to ensure the delivery of pre-processed data that is ready for analysis, including the consideration of quality masks and scaling factors applied to data. Furthermore, the resulting data can be directly converted into formats that can be read by other software, such as the Rasdaman database or Open Data Cube software. In addition, specific data formats for data extraction services requested directly by applications are necessary (Table 2.3c).

Data analysis

The ability to provide analysis services enables users to convert data into information. However, in many cases, the algorithms for analyzing time-series data need to have input data in a specific data format and structure. In addition, in most cases, there are several dependencies when setting up the algorithm on local computers. Thus, web processing services, which allow the conducting of analyses in addition to accessing data, are an important step forward in traditional EO data analysis. Furthermore, as the resulting output formats of the algorithms are not always considered to be user-friendly, additional post-processing steps are necessary to provide specific data formats, services, and tools (Table 2.3d). This allows users direct visualization and interpretation of the results of analysis without the use of additional software.

Table 2.3: Potential output and response requirements for data discovery (a), data access (b), data extraction (c), and data analysis (d) service outputs.

	a) Data discovery	**b) Data access**	**c) Data extraction**	**d) Data analysis**
Data formats				
Vector (e.g., Shapefile)	☑			☑
Raster (e.g., GeoTIFF)		☑		☑
Figures (e.g., PNG, JPEG)	☑	☑	☑	☑
CSV	☑		☑	☑
JSON	☑		☑	
Services				
Web service feed	☑		☑	
OGC download service		☑	☑	☑
OGC visualization service	☑	☑		☑
Tools				
Application-Ready-Data*		☑		
Downloads	☑	☑		☑
Summaries	☑	☑		☑

* Output is directly transferred into an application or database

2.3.4 Summary and conclusions

Several functional and technical requirements for user-aligned exploration of EO time-series data have been explored and categorized into web platforms, web technology, and data formats. These need to be considered when designing user-aligned services for the exploration of EO data. The foundational basis of user-aligned exploration of EO data is the availability of various, individual EO data sources as well as individual analysis tools linked to the data. From a technical point of view, web services based on current web technologies, user-specific output formats, and uniform and standardized specifications are important requirements.

In this thesis, the following requirements are further considered and investigated:

Web platforms

- Multi-source EO and geospatial data
- Data visualization
- Data download
- Analysis tools
- Visualization of the results of analysis
- Support for various programming languages and tools
- Metadata for geospatial data
- User management
- Self-hosting
- Service interfaces

Web technology

- Service architectures
- Asynchronous execution
- Process chaining
- OGC standardization
- Uniform specifications
- Multiple output formats

Data formats

- Commonly used data formats
- Analysis and application-ready data
- Summarized results
- OGC web services

Chapter 3: State of the Art

Based on the user requirements presented in the previous chapter, the state of the art of the principal web technologies, specifications, and formats for EO time-series data is described in this chapter. The following main topics are explored and evaluated for their use with EO time-series data:

1) Web technologies, including web service architectures, standardization, web data formats, software for web services and web processing applications, cloud-based infrastructures, and web platforms (Section 3.1).

2) Service and format specifications for EO time-series data, including data discovery, access, visualization, brokering, and processing (Section 3.2).

Finally, an evaluation regarding the state of the art in relation to user requirements is presented.

3.1 Web Technologies

The Internet enables the retrieval of distributed web resources and the execution of tasks on distributed computers. Modern web-based systems make use of web services and interactive applications. The enhancements of browser applications and mobile devices allow easy exploration of geospatial data in general. Advanced web development methods can be used within the geographic domain to increase the availability and handling of EO data archives. In recent years, more EO and geospatial data have been made available through web services for data discovery, access, and analysis in combination with web- and cloud-based applications (Vitolo et al. 2015; Wagemann et al. 2018). In the following sections, basic information how to design and provide web services based on state-of-the-art web technologies is presented and current solutions for web technologies are evaluated in relation to user requirements.

3.1.1 Web service architectures

The provision of web services for client applications, such as web- or mobile-based applications, is based on web service architectures that are compliant with the client-server model (Svobodova 1985). Based on this model, a server can provide different services requested by client applications. After requesting a service, the server delivers a response according to the request parameters. Figure 3.1 shows the different components and their interactions: The services provided by the server can be based on different standards, such as the File Transfer Protocol for data exchange or the HyperText Transfer Protocol (HTTP) for the World Wide Web. The response format may be diverse, depending on the protocol and the service (e.g., text, images, or binary).

A uniform resource locator (URL) is used to identify the server, the service, and the requested resource on the server (e.g., *http://artemis.geogr.uni-jena.de:80/myfeed*). In this example, the URL is composed of the protocol identifier ("*http*" for HTTP-based resources), followed by the hostname of the server ("*artemis.geogr.uni-jena.de*"), the port number pointing to a specific service (this can be omitted for standard port 80), and the resource requested ("*myfeed*").

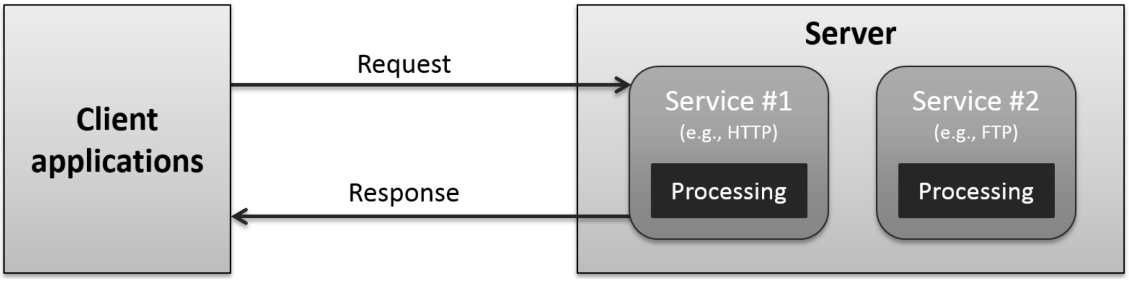


Figure 3.1: Client-server model involving different services provided by the server and the interactions between client and server/service (request and response).

Web-based services and resources

Web services provide machine-to-machine communication by transferring machine-readable file formats. The World Wide Web Consortium (W3C) defines a web service as follows: “A Web service is a software system designed to support interoperable machine-to-machine interaction over a network” (Booth et al. 2004).

Although in general web services have been based on the Simple Object Access Protocol (SOAP), in recent years, the focus for providing service-based solutions has changed—the principles of the World Wide Web have been adapted using resources based on the Representational State Transfer (REST) specification. The W3C has identified two major approaches to how web services can be built (Booth et al. 2004):

1) REST-compliant web services for manipulating representations of web resources using a uniform set of stateless operations, and
2) Arbitrary web services providing a set of operations (e.g., using SOAP).

While SOAP provides an object-oriented approach with clearly defined web service interfaces, REST is a simple approach with no interface specifications and is based on the basic technologies of the Web. That is, two service architectures are available to publish services, service-oriented architecture (MacKenzie et al., 2006) and resource-oriented architecture (Overdick, 2007). Both of them can also be used for geospatial services. Mazzetti et al. (2009) state that “it is not possible to say that one architecture is better than the other. The selection of the most effective system architecture depends on application requirements” (Mazzetti et al. 2009, p. 46). These authors selected the REST approach for their Earth System Science applications. Figure 3.2 shows the increasing trend for REST compared to the decreasing trend for SOAP within Google Search (Google 2018b).

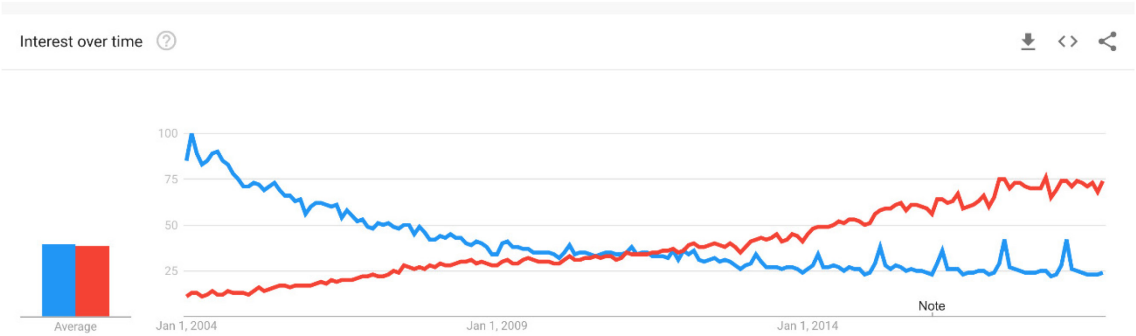


Figure 3.2: Google search trend (October 2018): SOAP (blue) vs. REST (red) (Google 2018b).

In addition, the “Programmable Web” as a directory of web service applications shows the importance of REST-based web services relative to other architectural styles, such as Remote Procedure Calls (RPC), which is used by SOAP: “As expected, REST is by far

the most commonly used architectural style given its prominence in API design over the last decade plus. RPC is the next most commonly used, owing to the many SOAP and XML-RPC-styled APIs that appeared, especially in the early days of our directory." (Santos 2017).

Synchronous and asynchronous service execution

Web services can be executed either synchronously or asynchronously. The latter means that a response is given immediately, while the service is still processing the request. A comparison of the synchronous and asynchronous execution of web services is listed in Table 3.1. Whereas synchronous execution is less complex and produces no communication overhead in server and client implementation, asynchronous execution provides useful features, such as the independent provision of outputs, independence from network failures, and "do not wait until execution is finished" behavior. Thus, asynchronous execution is an important property for many processing tasks—especially long-running tasks.

Table 3.1: Comparison of the synchronous and asynchronous execution of web services.

	Advantages	**Disadvantages**
Synchronous execution	• Immediate response includes the requested output • All output results at once • Less complex in terms of server and client implementation • Low overhead	• Connection to service required until completion • Server timeouts need to be properly configured
Asynchronous execution	• Provides outputs when ready • Possible to close application and check status later • Network independent (in case of failures)	• More complex in server and client implementation • Large overhead • Client needs to regularly check the status of the execution

Evaluation: A major advantage of RESTful web services is their ease of use in web browsers, various applications, and programming languages. Thus, the provision and use of RESTful web services has been increased in recent years relative to SOAP-based services. Although the decision about which architecture to use depends on the application, RESTful web services need to be considered. In addition, both synchronous and asynchronous execution are relevant depending on the use case of the individual web service. Long-running data integration or analysis tools should be supported by asynchronous executions.

3.1.2 Standardization

Standardization of data and services is relevant to enable the exchange of data and the use of services. Figure 3.3 shows the relationship between various standardization organizations. Standards and specifications for basic web technologies are mainly provided by the W3C, the Internet Engineering Task Force (IETF), and the Organization for the Advancement of Structured Information Standards (OASIS). These provide regulations for infrastructure and languages that are generally used for web-based communication and data exchange (e.g., the Extensible Markup Language). For geospatial purposes, the OGC specifies software interfaces and encodings with a domain- and infrastructure-specific emphasis. Legal and domain-specific standardizations are provided by the International Organization of Standardization (ISO) and the European Committee for Standardization (CEN).

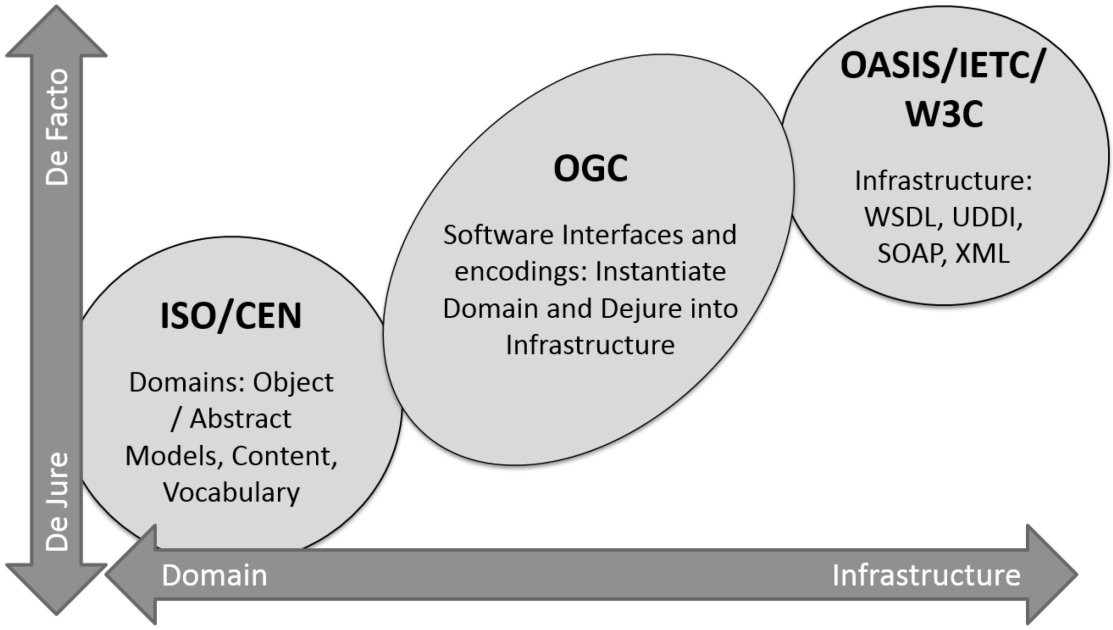


Figure 3.3: Tasks of the individual standardization organizations in relation to “De Jure—De Facto” and “Domain—Infrastructure” standards (after Trakas 2010).

Within ISO, “Technical Committee 211— Geographic Information/Geomatics” develops standards in the field of digital geographic information. These geographic information standards are published in the ISO 19100 series. The specifications describe the standards for basic geographic concepts, metadata, data content and definitions, core data model, data exchange formats, data interchange and services, data quality, spatial referencing, and imagery data. ISO specifies abstract models as well as their structures and content. The technical implementation of geo-related models and services is mostly specified by the OGC. Some of the specifications of the OGC have also been adopted by ISO (e.g., ISO 19128 Geographic information—Web map server interface).

The OGC comprises organizations and individuals from companies, governmental agencies, and research institutions. The aim of the OGC is the development of standards and technical specifications in the geospatial domain. The standards support interoperable solutions to geo-enable the Web, wireless and location based services, and "mainstream" IT (OGC 2018a). In contrast with standards published by ISO, OGC provides technical implementation specifications for interfaces and encodings. Developers of software can use these specifications to build and use services, which are compliant with the standards of ISO and the OGC. Some of these specifications are described in Section 3.2. Furthermore, specifications for data formats, such as the Geography Markup Language (GML) are defined by the OGC.

Evaluation: Several international organizations exist to standardize web technologies, geospatial data, and geospatial web services. The implementation specifications of the OGC especially are important, as software exists that supports data and web services that are provided with OGC specifications. Thus, the support of standardized specifications can only be recommended.

3.1.3 Structured data formats

Structured data formats provide the possibility of transferring structured data over web services based on text. Structured data are mostly provided in formats such as Extensible Markup Language (XML) and the JavaScript Object Notation (JSON). Both XML and JSON representations are shown in Figure 3.4, which describes structured data of a specific person.

Extensible Markup Language

XML is a markup language developed by W3C and defined by W3C's XML 1.0 specification (Bray et al. 2008). The specification consists of a set of rules for encoding documents in a format that is human- and machine-readable (Figure 3.4, left). XML was designed to work across the Internet. An XML document consists of elements (e.g., person and address) and attributes (e.g., firstName). The content of an XML document is not determined but can be defined in schemas. Schema systems, such as the XML Schema (Fallside & Walsmley 2014), provide information about what kind of XML elements and attributes are allowed in an XML document. This allows the standardizing of XML documents for individual purposes and standard-compliant web services.

JavaScript Object Notation

JSON is a text format for structured data interchange between programming languages defined by the JSON Data Interchange Standard of ECMA International (ECMA

International 2017). It was inspired by objects from the web programming language JavaScript and has been adopted by other programming languages (e.g., Python). JSON consists of name-value pairs in a structured format (Figure 3.4, right) and supports multiple data types (e.g., String, Number, Boolean, Array, and Null value). Today, the JSON format is preferred for modern web technologies as the data structure can be directly used as an object in various programming languages.

```xml
<person firstName="Jonas" lastName="Eberle">
  <age>99</age>
  <address>
    <streetAddress>Mainstreet 15</streetAddress>
    <city>Jena</city>
    <postalCode>07745</postalCode>
  </address>
  <phoneNumbers>
    <phoneNumber>
      <type>home</type>
      <number>03641 999999</number>
    </phoneNumber>
    <phoneNumber>
      <type>fax</type>
      <number>03641 999998</number>
    </phoneNumber>
  </phoneNumbers>
  <gender>
    <type>male</type>
  </gender>
  <children/>
</person>
```

```json
{
  "firstName": "Jonas",
  "lastName": "Eberle",
  "age": 99,
  "address": {
    "streetAddress": "Mainstreet 15",
    "city": "Jena",
    "postalCode": "07745"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "03641 999999"
    },
    {
      "type": "fax",
      "number": "03641 999998"
    }
  ],
  "gender": "male",
  "children": []
}
```

Figure 3.4: Structured data in XML (left) and JSON (right) formats.

Evaluation: Both XML and JSON formats provide machine-readable structured data. The structure of XML documents can be exactly defined (e.g., by the XML Schema). Although XML is widely used, XML parsing, especially in web browsers, is always a complex task. Thus, a major advantage of JSON is its ease of use in programming languages, as it can be directly converted into regular objects. With JSON, individual document parsing is not necessary.

3.1.4 Web service software

Many types of software can be used to provide web services. Depending on the task of the web service, different software may be of relevance. In light of this, web service software for data processing and content management are distinguished in this section.

Processing web services

Processing web services can be provided with already-existing software or the implementation of a self-developed web service. The OGC provides the Web Processing Service (WPS) standard to specify the provision, description, and execution of processing tools as interoperable web services. Based on the requirements described in Section 2.3, several criteria for the evaluation of software are considered:

- Support for asynchronous execution
- Multiple output formats
- Support for technology (REST/SOAP)
- Output provision as inline and reference
- Standardization
- Reproducibility
- Support for JSON output format

Three software packages are compared: *PyWPS* as representative of standard-compliant WPS software, *OpenCPU* as nonstandard-compliant software, and *Python Flask* as a self-developed web service solution.

Standard-compliant web service software

PyWPS is a standard-compliant implementation of the OGC WPS specification (Schut 2007) written in Python and released with an open-source license (Čepický & Becchi 2007; Čepický & De Sousa 2016). As the software uses the WPS specification, the provision of web services is exactly defined.

Any Python library can be used in a process. In addition, command-line executable tools can be used. Analysis tools from R or GRASS GIS can be used directly within Python using the rpy2 or GRASS GIS Python libraries. For each of the processes, a description is necessary for the process itself and for their input and output parameters that are provided to client applications. The interaction with the service is based on the methods defined in the WPS specification (GetCapabilities, DescribeProcess, and Execute). Listing 3.1 shows an example process implemented using PyWPS.

Based on the OGC WPS specification, PyWPS supports the following aspects (Čepický & Becchi 2007):

- Encode requests for process execution and responses from process execution
- Embed data and metadata in process execution inputs/outputs
- Reference web-accessible data inputs/outputs
- Support long-running processes
- Return process status information
- Return processing errors
- Request storage of process outputs

Furthermore, multiple output formats and the provision of outputs as URL references are features supported by PyWPS. Support of SOAP is available in PyWPS 3, but not in the newly written PyWPS 4 due to a lack of interest from the user community. Any of the executions of PyWPS are conducted with simple HTTP requests and REST-based technology. XML is the execution response format compulsory in the WPS specification. Unless the source code of the processes is not shared, reproducibility is not given. Although links to external websites and documents can be integrated in the process description (e.g., a link to the source code of the process), this is not mandatory for processes. Furthermore, in many cases, reproducibility relies on specific versions of tools installed on the server on which the process is running. These are not shown to the user.

The OGC WPS specification is also provided by other software, such as ZOO WPS and 52°North WPS. The main differences are the programming languages supported. Overviews are provided by Zhao et al. (2012) and Lopez-Pellicer et al. (2012).

```
01 class SayHello(Process):
02     def __init__(self):
03         inputs = [LiteralInput('name', 'Input name', data_type='string')]
04         outputs = [LiteralOutput('response','Output, data_type='string')]
05 
06         super(SayHello, self).__init__(
07             self._handler,
08             identifier='say_hello',
09             title='Process Say Hello',
10             abstract='Returns a string with Hello plus the inputted name',
11             version='1.3.3.7',
12             inputs=inputs,
13             outputs=outputs,
14             store_supported=True,
15             status_supported=True
16         )
17 
18     def _handler(self, request, response):
19         response.outputs['response'].data = 'Hello ' +
               request.inputs['name'][0].data
20         return response
```

Listing 3.1: Implementation of a process using PyWPS.

Nonstandard-compliant web service software

OpenCPU focuses on making scientific computing reproducible based on an interoperable HTTP web service interface for data analysis within R (Ooms 2014). Any standard R function as well as specifically developed functions can be handled within a unique web service interface. The parameters of the functions are the input parameters of the process. The return values of the functions are the output data of the process. In addition, figures plotted in the function are stored automatically and can be retrieved as further output data. Although a list of processes can be retrieved, no further descriptions of the process, input parameters, and output datasets are provided. Listing 3.2 shows an individually developed function in R with input parameters ‘x’ and ‘y’, which can be requested using the web service interface provided by OpenCPU.

```
1 extractWQ <- function(x,y) {
2   library(raster)
3   r <- brick("SWOS_WQ_CDOM_FUB_Monthly_France_Berre.vrt")
4   data <- extract(r,SpatialPoints(cbind(x,y)))
5   plt <- plot(c(data), type="l")
6   result <- list("values" = data)
7   return(result)
8 }
```

Listing 3.2: Implementation of a process using OpenCPU.

Multiple outputs as well as different execution response formats (e.g., text/plain, JSON, CSV, graphics, and R-workspace) are supported but no description of each of the inputs and outputs is provided. As the source code of the function can be retrieved, the names of the input parameters are available. To keep the interface and server implementation simple, only synchronous execution has been made currently available. This is not suitable for long-running processes although the timeouts of the server can be configured. OpenCPU follows its own standardization and thus client applications need to adapt the building of execution requests and parsing of responses. The technology is based on HTTP requests with a RESTful architecture; support for SOAP is not available.

Individual web service solutions

Processing tools can be published using various other forms of software, such as Flask for Python and *servr* for R. In contrast to the previously services described, the methods of web services need to be implemented by each application. A service example using the Python Flask framework is shown in Listing 3.3 with a REST-based service providing access to the resource `/entries/<post_id>` using methods of HTTP (e.g., GET, PUT, and DELETE). Within this source code, any Python library and command-line tool can be used. The same can be implemented using the servr library for R.

```
@app.route('/entries/<int:post_id>', methods=['GET','PUT','DELETE'])
def handle_post(post_id):
    if request.method == 'PUT':
        return change_post(post_id, request.data)
    elif request.method == 'DELETE':
        return delete_post(post_id)
    else:
        return get_post(post_id)
```

Listing 3.3: Provision of a web service using Python Flask.

Content management

Content management is important for providing functions for user authentication, available datasets and analysis tools, or areas of interest specified by individual users. Several open source software packages exist to provide functionalities for content management, such as popular web content management software like *Drupal CMS* and *Django Web Framework*. Both allow data management structured in individual "content types" with "content fields," which can be set up individually for each content type. While Drupal CMS is based on Hypertext Preprocessor (PHP) scripting language, Django Web Framework is based on Python scripting language. Both kinds of software support the management of different data types, allow for user registration and authentication, and provide RESTful web services.

Evaluation: In general, each of the software packages available has advantages and disadvantages, which are more or less relevant for specific purposes. In most cases, the programming language of software is the most important factor for selecting it, in addition to the functions that the software provides.

While web services for content management are in most cases too application-specific, services for geospatial processing tasks can easily be provided with standardized service specifications (e.g., the OGC WPS). This processing service specification already supports many of the features required to enable user-aligned web services. Others, such as OpenCPU, are only suitable for specific purposes as it depends on the functions they offer. For example, OpenCPU is better suited for prompt responses through the lack of asynchronous support, for example, the extraction of a time-series from a raster time-series stack. As OpenCPU as an application written in R, it is simple to provide R-functions as web services. Therefore, R-functions can be provided and used as web service using a simple approach.

Although anything can be provided with a self-developed web service, each of the functions (e.g., asynchronous processing) needs to be implemented by the developer him- or herself. As such, the use of already existing web service solutions (e.g., PyWPS or OpenCPU) is preferred for the provision of processing services.

3.1.5 Cloud-based infrastructures

Web services provide existing tools and applications to users. However, scientists and developers especially are interested in applying algorithms to the data, in addition to data discovery and data access. As the amount of available EO data increases daily, solutions are needed to move algorithms to data. Cloud providers offer different kinds of technologies to make use of their infrastructure. In addition to virtual machines that can be launched and accessed via remote virtual environments, new technologies enable the use of cloud infrastructures with serverless applications. Both are described and evaluated in what follows.

Remote virtual environments

Virtual environments focus on providing a virtual operating system to users. Users are able to use and install software tools, such as GIS and remote sensing-specific tools. EO data archives can be directly accessed and used in the software packages. Two different approaches can be distinguished, those that use either a virtual desktop or merely a virtual command-line environment. With both, users connect using a remote accessing tool and have the same experience as working on their local computer. Remote virtual environments can either be used with local virtual servers or cloud-based infrastructures. The connection to remote environments is either based on secure shells or remote desktop tools, which provide access to command line and desktop applications. With support of specific applications (e.g., Apache Guacamole), virtual desktops are available through web browsers. Besides large cloud infrastructure platforms, such as Amazon Web Services or Google Cloud, ESA and the European Union have also launched platforms, such as ESA RSS Toolbox, EU RUS Copernicus, and the recently launched EU Copernicus DIAS platforms. All of these provide on-demand virtual environments close to EO data archives.

Serverless web services

Serverless infrastructure describes a technology for undertaking tasks for which the underlying server infrastructure is set up and used only when the tasks are running. Computational tasks can be triggered by an event, such as a request conducted by a web service application. This allows the saving of resources and costs as the server infrastructure is set up automatically only on demand. Due to the infrastructural concept underlying Amazon Web Services (AWS) and Google Cloud, both support serverless computing with their products, AWS Lambda and Google Cloud Functions. Limits pertaining disk space, execution duration, request body size, memory allocation, and software package size need to be considered when designing such web services (Table 3.2). Examples can be found on the remotepixel.ca website, which provides a Landsat

scene viewer (landsat-tiler[10]) and Landsat NDVI extraction (remotepixel-api[11]). Both web applications make use of AWS Lambda web services to search for Landsat data, return visualization images in different band combinations, and calculate NDVI data for the area of interest.

Evaluation: Any application hosted on local servers can be transferred and hosted on cloud-based infrastructure close to EO data archives. Various aspects need to be considered when using virtual remote environments:

- No data download is needed if the data is already available on the same cloud.
- Costs for download bandwidth and storage when data required by the user is not available on the server infrastructure.
- Costs for the 24/7 runtime of the processing infrastructure depends on performance.

Serverless web services provide simple tools based on data that already is hosted by the service provider. As such, data can be made available, but long-running data processing or analysis tools are not supported due to the limitations of the serverless architecture. Although it is not necessary to rent a server for the 24/7 time range, the cost is calculated based on the number of requests and the computation resources used.

Table 3.2: Limits of the serverless tools from Amazon and Google (Status: Nov. 2018).

Limit	**AWS Lambda[12]**	**Google Cloud Functions[13]**
Package size	50 MB	100 MB
Disk space	512 MB	See memory allocation
Execution duration	300 seconds	540 seconds
Request body size	6 MB	10 MB
Response size	6 MB	10 MB
Memory allocation	128–3,008 MB	128–2,048 MB

[10] https://viewer.remotepixel.ca
[11] https://remotepixel.ca/projects/landsat8ndvi.html
[12] https://docs.aws.amazon.com/lambda/latest/dg/limits.html
[13] https://cloud.google.com/functions/quotas

3.1.6 Web platforms

Individual processing platforms automatically combine data access and data processing tools. In contrast to virtual environments (see Subsection 3.1.5), only tools that are published by the platform provider can be used. The general objectives of web-based processing platforms are the provision of ready-to-use data in combination with convenient tools and useable interfaces.

Different architectures and tools can be considered when designing infrastructures for web-based EO data exploration. Soille et al. (2018) have compared several multi-petabyte platforms for geospatial data processing and analysis, including NASA's Earth Exchange, the Australian Geoscience Data Cube, Google Earth Engine, and JRC Earth Observation Data and Processing Platform (JEODPP). Different properties have been assessed: 1) the capability to address users with remote desktop access, analysis software, and interactive visualization and analysis tools; 2) access to its different services through a web browser client; and 3) the possibility of running existing scientific workflows in a virtualized or containerized environment. These criteria lead to the conclusion that a self-developed infrastructure, such as the JEODPP, is best for use in terms of flexibility for users. Similar web-based applications and infrastructures have been set up by the ESA Thematic Exploitation Platforms (e.g., Esch et al., 2016) and the CEOS Open Data Cube initiative (Ariza-Porras et al. 2017; Giuliani et al. 2017). Both of these can be hosted either on own infrastructure or within commercial cloud providers close to EO Data archives, e.g., using AWS, Google, Copernicus DIAS, or Terradue Cloud (Caumont et al. 2014). In contrast, Google Earth Engine (Gorelick et al. 2017) provides highly scalable infrastructure that is free to use after registration and has petabytes of EO and geospatial data. However, own algorithms, for example, those available as an R package or command-line tool, need to be rewritten with functions provided by Google. Although own datasets can be uploaded, these are limited to raster and vector data.

Evaluation: Various web-based processing platforms exist that have different EO data, analysis tools, service interfaces, and infrastructures. Although the development and use of a self-developed platform is flexible for most of the requirements, it needs either to be hosted on a cloud platform close to the data (e.g., Amazon, Google, and Copernicus DIAS) or EO data needs to be downloaded. However, the use of existing platforms (e.g., Google Earth Engine) can be limited in terms of the available EO data and analysis tools. A final decision often depends on several aspects: the functionality of the web platform, the EO data that is needed, or the own processing capabilities, and the like.

3.2 EO Time-series Data Services and Formats

Services for EO time-series data discovery, access, brokering, visualization, and processing allow data exploration in various user-driven applications. In the following subsections, state-of-the-art specifications and research on each of these services are described and evaluated in relation to user requirements.

3.2.1 Discovery

Requests for geospatial data discovery include location and optional query parameters to filter down the search result; the responses include the resulting metadata in the data format requested (e.g., XML or JSON). In general, metadata cataloguing focuses on individual geospatial data—not only that related to time-series data. A hierarchical composition of satellite missions containing millions of satellite scenes has been demonstrated by discovery brokers, such as the ESA Federated Earth Observation Gateway (FedEO) and GEOSS metadata broker (Nativi et al. 2014; Craglia et al. 2017).

To allow interoperability between distributed metadata catalogues, standard-compliant service specifications are available. The OGC has published the Catalogue Service for Web (CSW) specification, which contains rules for querying metadata catalogues and accessing metadata records. Other catalogue specifications, such as OpenSearch, which was initially developed by Amazon, are available for general data discovery. The OGC developed an extension for OpenSearch that specifies features relevant to geospatial data, such as spatial and temporal properties and spatial filter methods. Metadata (e.g., title, abstract, and contact information) enables the discovery of geospatial datasets in catalogues. Given this, ISO published the ISO 19115 specification (ISO 2003) to determine what kind of information can be included. The structure of the metadata files is defined in ISO 19139 as a standardized XML encoding schema. EO-specific metadata can be made available using additional extensions of the existing metadata catalogue standards, such as the OGC OpenSearch Extension for Earth Observation (Gonçalves & Voges 2016) or the OGC EO Application Profile for CSW 2.0 (Gilles 2006).

Research has been conducted on semantic annotation of satellite data to automatically enhance metadata, either by automated feature extraction (e.g., Cui et al. 2014) or information linked to the footprint of each satellite scene extracted from additional data, such as land cover, population, and the digital elevation model (Gasperi 2014). Semantic information can be integrated using common discovery standards, such as OpenSearch and OGC CSW, or based on a linked data approach using the Resource Description Framework (RDF). As a result, a standardized query language can be used to search within EO data archives for additional annotations and linked data (Koubarakis et al. 2012;

Arenas et al. 2017). A combination of raster time-series discovery and access has been researched within the EarthServer project (Baumann et al. 2016a). Based on the OGC Web Coverage Processing Service (WCPS) specification, the XPath-enabled WCPS extension has been developed to semantically search within the metadata of each raster in the time-series (Liakos et al. 2015). To increase interoperability when searching for spatial time-series data, the SpatioTemporal Asset Catalog (STAC, Holmes 2017b) specification has been initiated by various international organizations. It aims to standardize how geospatial data is made available online and can be queried using up-to-date and modern web technologies. Currently, only an early version of the specification is available and, hence, there is potential for some major changes (Radiant Earth 2018).

OGC Catalogue Service for Web

The CSW (Nebert et al. 2007) is a standard specification of the OGC for data discovery. It uses a simple HTTP interface to make metadata discoverable and accessible. Within the specification, requests, responses, and filtering parameters of the web service are defined. Table 3.3 lists the core methods, which can be described as follows: The "GetCapabilities" response lists methods with their parameters as well as filter capabilities, such as geometry operands, spatial operators, comparison operations, and arithmetic operators. The "DescribeRecord" method provides information about the metadata fields that can be retrieved. The "GetRecords" method can be used to search the metadata catalogue using keywords and other filters. Various parameters can be set to define the resulting list of metadata (e.g., short or detailed). The "GetRecordById" method provides the complete metadata record according to the identifier of the metadata record set in the request.

Table 3.3: Core methods of the OGC CSW specification (Nebert et al. 2007).

Request method	Description	Output format
GetCapabilities	Lists all available methods, parameters, and filter capabilities	XML
DescribeRecord	Describes the metadata elements available	XML
GetRecords	Discovers metadata catalogue	XML
GetRecordById	Retrieves metadata entry by unique metadata file identifier	XML

Example of a CSW request (GetCapabilities):

```
http://artemis.geogr.uni-jena.de/pycsw/csw-
projects.py?service=CSW&request=GetCapabilities
```

Example of a CSW request (GetRecords):

```
http://artemis.geogr.uni-jena.de/pycsw/csw-
projects.py?service=CSW&request=GetRecords
```

OpenSearch

A non-domain-specific catalogue service specification is OpenSearch (Clinton 2018), which uses standardized web output formats based on XML technologies. The JSON format is additionally available. Depending on the configuration of OpenSearch, either only a limited set of metadata or all metadata can be integrated in the output of the data discovery request. However, another URL within the OpenSearch result can retrieve the full metadata. OGC published a spatial and temporal extension to meet the needs of the geospatial community (Gonçalves 2014).

Example of an OpenSearch description request:

```
http://artemis.geogr.uni-jena.de/ec/pycsw/swos/products.py?
service=CSW&request=GetCapabilities&mode=opensearch
```

Example of an OpenSearch search request:

```
http://artemis.geogr.uni-jena.de/ec/pycsw/swos/products.py?
service=CSW&version=2.0.2&request=GetRecords&mode=opensearch
&elementsetname=full&typenames=csw:Record&resulttype=results
&q=Azraq
```

Evaluation: Although the OGC CSW and OpenSearch specifications are used as standards for data discovery, they can be improved for specific cases. More user-aligned output formats for data discovery, which can be understood by a wide range of users, need to be provided. Examples include summarized figures and charts as well as commonly used data formats (e.g., Shapefile and CSV). To meet the needs of different users, more than a single output needs to be available.

3.2.2 Access

Geospatial raster data can be provided as direct file download or by using a web service interface. The provision of data downloads through web services allows on-demand requests of different projections, file formats, and subsettings. In addition to file download and web services, new technologies allow the reading, writing, and visualizing of GeoTIFF raster data in the browser (EOX IT Services GmbH 2018). An initiative to provide cloud-optimized GeoTIFF data, which can be accessed by means of subsets when requesting the data, has been initiated recently (COG 2018). This allows downloading of only parts of a raster file without using a web service interface, rather than the user having to download the complete raster file. In the following paragraphs current research and standard-compliant specifications for raster time-series access are described.

Although single observations can be provided using the OGC Web Coverage Service (WCS) specification, it is not optimized for large time-series data management and access. The provision of time-series data is in general provided by the OGC Sensor Observation Service (SOS) specification. Users can access data for a sensor of interest and further user-specific parameters, such as the selection of a measurement or a temporal range. This approach is mainly used for in-situ measurements when observing physical phenomena (e.g., temperature or precipitation), which are always related to a measurement station—a geographic point. In contrast, satellite-based EO delivers data related to a certain geographic area. This has been addressed by Sorg and Kunkel (2015), who published raster time-series data with the use of the OGC SOS specification. To allow for efficient storage and search, which have been major challenges, the raster data is made available with different grid solutions. The authors conclude that the “SOS approach is predominant compared with the common WCS-EO) approach due especially to its temporal and thematic filtering capabilities, but in particular due to the possibility of describing measurement equipment, measurement processes, and observations in detail by metadata standards, which are exactly defined for this purpose” (Sorg & Kunkel 2015, pp. 1093–1094). Other access possibilities include the research of databases, such as Rasdaman and SciDB (Baumann et al. 1998; Planthaber et al. 2012; Appel et al. 2018).

OGC Web Coverage Service

The WCS specification (Baumann 2012) allows the publishing of geospatial raster data using an HTTP interface. Multiple output formats for raster data can be retrieved, such as GeoTIFF or geo-referenced JPEG and PNG images. Using request parameters, users can request specific projections and subsets of the original data. Table 3.4 lists the core methods of the OGC WCS specification and the WCS-EO extension. A WCS-compliant service can contain several geospatial data listed in the “GetCapabilities” response. In

addition, available projections and interpolation methods are listed. A dataset can be further described with the “DescribeCoverage” request, which lists information about data properties (e.g., pixel size, columns, rows, or no data value) and dimensions (e.g., bands and time). The TIME extension of the WCS allows multi-temporal filtering before the resulting data is returned as a response to the user. Based on WCS, it is possible to integrate raster data into client applications, such as the GIS desktop system or web applications using state-of-the-art browser technologies. Furthermore, it is possible to download either the full dataset or only selected parts thereof by using subsetting and filter functions.

An EO application profile meets the needs of the EO community. This profile adds the methods “DescribeEOCoverage” and “GetEOCoverage,” with specific focus on multi-bands and multi-temporal satellite data (Baumann et al. 2014). As such, temporal and spatial information about various EO missions are included in the response formats.

OGC Web Coverage Processing Service

Another extension is the OGC WCPS, which allows the processing of raster data while requesting it(Baumann 2009a). Baumann (2009a) developed a raster database combined with the OGC WCPS specification to serve multi-dimensional raster data, which allows the extracting and processing of data when conducting a service request. Although the WCPS specification allows subsetting and processing of data while requesting it, the output of the multi-temporal dataset consists of a multi-dimensional raster dataset. In most cases, further metadata is necessary to relate each band to a date in the EO time-series data. WCPS is reasonable for the retrieval of a single dimension; accessing a time-series of raster data is more challenging because data management issues still need to be solved by the user even if some of the steps (e.g., merging and clipping) have been undertaken automatically by the web service. In the following examples, it is demonstrated how the WCPS query language can be used to retrieve a vegetation index from satellite data without the need to compute the index beforehand.

Evaluation: Existing data access services only partially meet the needs of users. They mainly focus on machine-to-machine interaction and raw data access. Several requirements, such as clipping and merging data to the area of interest, using quality masks and scaling factors, and converting data to user-defined output formats have not yet been fully addressed. Therefore, individual data integration and processing services are necessary. While it is possible to provide the raster data for each date individually with the OGC WCS and WCPS specifications, the handling of the requesting and further processing of all datasets according to the needs of a user is left to the client application.

Table 3.4: Core methods of the OGC WCS specification and the WCS-EO extension (Baumann 2012; Baumann et al. 2014).

Request method	Description	Output format
GetCapabilities	Lists information about operations and coverages available	XML
DescribeCoverage	Retrieves information about a single coverage	XML
DescribeEOCoverage	Retrieves information about single EO dataset collection	XML
DescribeEOCoverageSet	Retrieves information about zero or more EO datasets that meet the request parameters (e.g., dataset IDs and spatial/temporal dimensions)	XML
GetCoverage	Downloads geospatial coverage based on specific parameters	GeoTIFF/others
GetEOCoverage	Downloads EO coverage based on specific parameters	GeoTIFF/others

Example of a WCS request (GetCapabilities):

```
http://artemis.geogr.uni-jena.de/sibessc/modis?service=WCS&
version=2.0.0&request=GetCapabilities
```

Example of a WCS request (GetCoverage):

```
http://webgis.essi-services.org:8080/geoserver/ows?
service=WCS&version=2.0.1&request=GetCoverage&coverageid=web
gis__tuebingen-landcover_300m_band1
```

Example of a WCS request (GetCoverage with subsetting):

```
http://webgis.essi-services.org:8080/geoserver/ows?
service=WCS&version=2.0.1&request=GetCoverage&coverageid=tue
b_srtm_30m&subset=Long(9.05,9.15)&subset=Lat(48.50,48.55)&
subsettingcrs=http://www.opengis.net/def/crs/EPSG/0/4326
```

Example of a WCPS (ProcessCoverage) request calculating the NDVI for a subset of a single image within a multi-temporal dataset (LT5):

```
http://yourserver/rasdaman/ows?service=WCPS&version=2.0.1
  &request=ProcessCoverage
  &query=for c in (LT5) return encode( (((float)c.b4-
(float)c.b3)/((float)c.b4+(float)c.b3))[E(700000:784815),N(5
104285:5158115),ansi("1984-10-18")],"tiff")
```

3.2.3 Brokering

Brokering systems for geospatial data have been introduced by Nativi et al. (2012) to achieve interoperable web services across multi-disciplinary systems and standards. Nativi et al. (2012, p.6) defines a broker as follows: “A solution to reduce the interoperability burden on data providers and applications is to introduce a third party to interconnect the different service buses, mediating their existing (and future) models and interface specifications.”

Figure 3.5 shows both approaches: The traditional approach with requests to each individual data provider (right) and the service-brokering approach with a centralized request to the broker instead of to each individual service of the data provider (left). The service broker takes over the communication with each connected web service and adapts requests from users to the service specifications of the data provider.

While the service-brokering approach has been implemented for data discovery by various international organizations (e.g., GEOSS), the brokering of data access is more complex. Although Nativi et al. (2013) propose service brokering for data access, it has been introduced for individual geospatial data and does not focus on time-series data. Based on the same approach, a brokered virtual hub layer for historical maps has been developed (Previtali & Latre 2018). The study investigates how to remove the barrier introduced by data and services by different user communities to ensure the effective reuse and integration of geospatial data by software developers. It concludes that “the capability to integrate different informative layers, both historical and modern, can be an important opportunity of development with application areas still largely unexplored” (Previtali & Latre 2018, p. 19). In addition, architectures with only a single point of access ensure easy interoperability between different sources.

Evaluation: A brokering approach enables the use of a unified and harmonized interface for the different services that are connected. Users only need to use a single service.

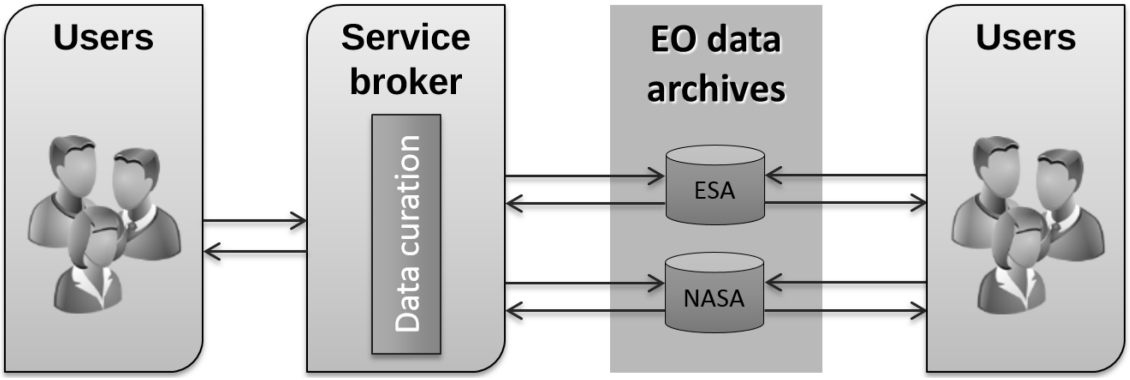


Figure 3.5: Requests to EO data using individual services (right) and service brokering (left).

3.2.4 Visualization

Geospatial raster and vector data are visualized in the web browser by converting them to RGB or greyscale images. Raw values from raster data and features from vector data hence need to be converted into color schemes (Figure 3.6). The OGC allows visualization using the Styled Layer Descriptor specification. For raster data, individual pixel values are converted either into discrete colors or into a continuous range of colors. For vector data, the color assignment is based on a column in the attribute table (e.g., different colors for land-cover types specified in the attribute table).

OGC Web Map Service

The OGC Web Map Service (WMS, Beaujardiere 2006) specification provides a simple HTTP interface to convert geospatial data to images in order to make them available on the Web. Several parameters can be added when retrieving an image, such as a bounding box or a specific styling. If data is provided using the OGC WMS specification, it can be easily integrated in existing GIS desktop and web-mapping software. Within a web-based system, geospatial data created on demand (e.g., outputs of the results of analysis) can be directly visualized in an interactive map viewer when a WMS is provided. No data download or additional software is required by the user. Table 3.5 lists the core methods of the WMS specification. A WMS can contain several geospatial data, which are listed as layers in the “GetCapabilities” request. In addition to the visualization of geospatial data, a WMS provides on-demand projection in order to transfer geospatial data into other projections within the “GetMap” request. It also supports the request for raw values of the geospatial dataset for individual pixels in raster or features in vector datasets (GetFeatureInfo). A legend graphic containing explanations of what the colors represent can be requested using the “GetLegendGraphic” request.

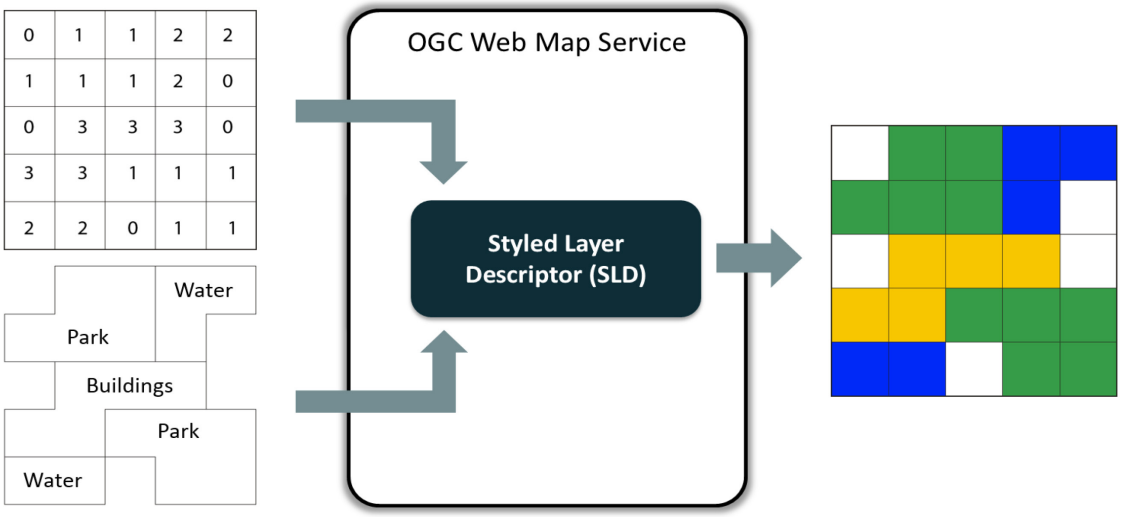


Figure 3.6: Workflow of a web-based visualization of geospatial data: Either raster data (top left) or vector data (bottom left) can be transferred into images (own visualization).

Table 3.5: Core methods of an OGC Web Map Service (Beaujardiere 2006).

Request method	Description	Output format
GetCapabilities	Lists all available datasets	XML
GetMap	Retrieves images from given datasets in a geographic area	Image
GetLegendGraphic	Retrieves a legend graphic from given dataset	Image
GetFeatureInfo	Retrieves raw values from a specific dataset and location	HTML/others

Example of a WMS request (GetCapabilities):

```
http://artemis.geogr.uni-jena.de/sibessc/modis?service=WMS&
version=1.3.0&request=GetCapabilities
```

Example of a WMS request (GetMap):

```
http://artemis.geogr.uni-jena.de/sibessc/modis?service=WMS&
request=GetMap&layers=myd10c2_snowcover&format=image/png&
srs=EPSG:900913&bbox=10018754,5009377,15028131,10018754&
width=256&height=256
```

WMS TIME Extension

The WMS TIME extension allows for the filtering of a multi-temporal dataset before the visualization is undertaken. This is based on a catalogue (e.g., through a database or vector file), which contains temporal information for each item in the multi-temporal dataset. For each “GetMap” request, the TIME parameter comprises either a single time period (e.g., year, month, or day) or a start and end date:

TIME=<start date>/<end date> Example: TIME=2018-01-01/2018-01-07
TIME=<date> Example: TIME=2018 (all scenes from 2018)

As the output is a single map, depending on the software, several strategies for the resulting multi-temporal images can be defined, such as mosaicking all resulting images or showing only the latest image within the given time range.

WMS Earth Observation Application Profile

The Earth Observation Application Profile of the WMS specification defines “conventions for the Earth Observation (EO) community to use OGC Web Services” (Lankester 2009), with the objective of providing an interoperable way to visualize EO data. That is, it defines how EO data, which contains dataset collections as well as temporal and band dimensions, can be provided using the OGC WMS specification. The temporal dimension is provided by the WMS TIME parameter. If multiple bands are available in the EO data,

they need to be provided by using an additional dimension (e.g., wavelength for optical data or polarization for radar data). An example request that includes the TIME parameter and the additional dimension for radar intensity (dim_sar) is shown below:

```
http://eoltd.co.uk/mapserver.cgi?version=1.3.0
      &request=GetMap&crs=CRS:84
      &bbox=78.105,24.913,94.794,36.358
      &width=560&HEIGHT=350
      &layers=ASA_IMP_1P_BANDS
      &format=image/png
      &time=2005-10-05T07:25:00
      &dim_sar=INTENSITY
```

Evaluation: Visualization of EO time-series data can be realized using the TIME parameter of the OGC WMS specification. In addition, the WMS EO application profile lists specifications regarding how to provide raster time-series layers. Although this is suitable for data provision, client applications need to support this extension.

3.2.5 Processing and analysis

The publishing of processing tasks as web service—close to the data or on dedicated processing hardware—plays a major role in dealing with the emerging opportunities, challenges, and needs for globally distributed data and increasing amounts of data. To better support data exploration, algorithms need to be provided as web services to transform data into information. Algorithms need to be directly linked with input data, which needs to be prepared for direct analysis ("analysis-ready data"). Therefore, EO data archives and analysis tools have to be linked. Web-based processing services enable the further processing of output data of algorithms, for example, converting geospatial output data to standardized web services for visualization and access. Furthermore, raster to vector conversion and ready-to-use maps can be added to allow easy exploration of the results of analysis without the need for further processing by the user. To ensure the deployment of algorithms without the need to consider the dependencies of the software and allow reproducible workflows, containerized solutions can be considered (Celesti et al. 2016; Beaulieu-Jones & Greene 2017). This allows the execution of an algorithm independently of server infrastructure.

Geoprocessing technologies are widely discussed in the research community. Hofer (2014) undertook a systematic literature analysis of the use of online geoprocessing and concluded that the field is still evolving and that further actions need to be focused on increasing the use of this technology (e.g., provide best practices and resources, reduce entry and access barriers, sharing of services, and the like). In 2009, a research agenda for geoprocessing services was published by Brauner et al. that identified three main

topics: 1) semantic descriptions of geoprocessing services, 2) orchestration of geoprocessing services, and 3) approaches for performance enhancements. A new paradigm has already been introduced and was later further developed to bring algorithms to the data (Friis-Christensen et al. 2007; Müller et al. 2010, 2013) to handle performance issues and increasing data volumes. This paradigm leads to a reduction in the amount of data transfer between different systems; however, it still needs to be further investigated. In addition, algorithms can be provided with interoperable standards. This can be achieved with geoprocessing services and is referred to as the "geo-enabled model web" (Geller & Turner 2007; Nativi et al. 2013; Dubois et al. 2013). Current research projects aim to standardize the connections to and between EO service providers by providing a standardized service interface to query and process EO data (Schramm et al. 2019) and "analysis-ready services" (Baumann 2019).

Further research has been conducted in several fields:

- Distributed processing (Friis-Christensen et al. 2007; Meng et al. 2010; Foerster et al. 2011; Schaeffer et al. 2012)
- Semantic processing (Farnaghi & Mansourian 2013; Wosniok et al. 2014; Vitolo et al. 2015; Sudmanns et al. 2018)
- Process orchestration (Nash et al. 2007; Meng et al. 2009; Eberle & Strobl 2012; De Jesus et al. 2012; Wu et al. 2014; Xiao et al. 2016; Hofer et al. 2017)
- Cloud-based processing (Sun 2013; Evangelidis et al. 2014; Veenendaal et al. 2016; Shelestov et al. 2017; Gorelick et al. 2017)
- Sharing geoprocessing logic (Müller et al. 2013)

OGC Web Processing Service

Service providers for web-based processing services need to describe their processes with input and output parameters, which can be individually set by the users. Process descriptions and execution methods can be standardized with the use of the OGC WPS specification (Schut 2007), which allows the publishing of processing tasks on the Web using an HTTP interface. A WPS-compliant service can contain several processes, which are connected to executable scripts on the server. Standardized methods (Table 3.6) allow a unique execution of processes and the handling of status updates. Available processes are listed in the "GetCapabilities" response. The process descriptions with information about available inputs and outputs and descriptions of the process itself can be retrieved using the "DescribeProcess" request. A process can be started using the "Execute" request, which includes the input values and properties for running the process, such as synchronous or asynchronous execution or whether the output values are stored on the

server. With version 2.0 of the WPS specification, there are further request methods to pause and cancel running processes. Using the WPS specification, processes can be started directly using data available on the local machine, the server infrastructure, or on the Web.

Table 3.6: Core methods of an OGC WPS (Schut 2007).

Request method	Description	Output format
GetCapabilities	Lists processes available in the requested WPS instance	XML
DescribeProcess	Describes process with inputs and outputs	XML
Execute	Executes a process	XML / Output file

Example of a WPS request (GetCapabilities):

```
http://artemis.geogr.uni-jena.de/cgi-bin/testbox.cgi?
service=WPS&request=GetCapabilities
```

Example of a WPS request (DescribeProcess):

```
http://artemis.geogr.uni-jena.de/cgi-bin/testbox.cgi?
service=WPS& version=1.0.0&request=DescribeProcess&
identifier=1013_single_ts_plot_point
```

Example of a WPS request (Execute):

```
http://artemis.geogr.uni-jena.de/cgi-bin/testbox.cgi?
service=WPS&version=1.0.0&request=Execute&identifier=1013_si
ngle_ts_plot_point&datainputs=datasetName=mod13q1_evi;pointX
=13.54;pointY=52.31
```

Evaluation: The OGC WPS specification allows a meeting of the needs of users though further standards related to input and output data formats, service execution properties, and the deployment of algorithms need to be defined (e.g., through WPS application profiles).

3.2.6 Data formats

As already described in Subsection 2.1.1 and shown in Table 2.2, multi-source EO data is available in different data formats (e.g., GeoTIFF, netCDF, HDF-EOS). The selection of user-friendly data formats needs to be observed from different perspectives: the technological perspective in order to provide efficient data access and visualization, and the user perspective, with commonly used data formats. Bordogna et al. (2016) recognized that there is still a gap to fill in order to enable the efficient access, retrieval, integration, visualization, analysis, and interpretation of geospatial time-series data, especially that with different data formats and from multiple sources. Furthermore, they identified a lack of complex functionalities that would allow stakeholders to easily perform queries on geospatial time-series data within current geoportals without the need to think about data formats and data structures. In addition, graphic diagrams showing time-series are mostly not available as query responses. Smith et al. (2016) describe similar needs, as data platforms "often do not demonstrate that data are readily available and easy to access and analyze [and] data insights are not realized without expertise in programming or other technical skills" (Smith et al. 2016, p. 98). Thus, user-friendly tools to explore aggregated time-series data and model outputs are necessary. In both of the above-referenced studies, data are structured as files with a predefined data format (e.g., GeoTIFF for raster time-series and text files for single time-series). In contrast to the GeoTIFF-based single layer data format and data structure (Astsatryan et al. 2015; Bordogna et al. 2016), the Hierarchical Data Format (HDF) is preferred by other researchers to allow a time-sequential representation of individual pixels for efficient raster time-series data access (Colditz et al. 2008; Van den Bergh et al. 2012). Gallaher and Grant (2012) propose the "data rods" time-series format for data management and data analysis. This allows raster time-series data to be converted into single time-series pixel objects that can be stored in object-based databases, which provide more efficient data access than accessing data from the file system. For all of the studies presented above, the main objective is overcome the current activities of data search, order, download, and transformation into a web service-based exploration and analysis of time-series data in an efficient and optimized way.

Evaluation: The choice of the "correct" data format is often not simple to answer as it depends on various subjects (e.g., user communities and diverse data structures). While the GeoTIFF format can only handle bands with the same raster dimensions in the same file, the HDF format can include different raster dimensions. However, depending on the user community, users may have more experience with one or the other data format.

3.3 Summary and Evaluation

The state of the art for web technologies and EO time-series data services and formats is summarized and evaluated in relation to user requirements in the following paragraphs.

Web technologies

Current web technologies are based on RESTful web services and make use of JSON for the exchange of structured data. Although SOAP-based web services can be useful in different domains and software architectures (e.g., orchestration engines), the RESTful approach needs to be supported for web services. The JSON data format is to be preferred in most cases as the data structure can be directly used as an object in various programming languages. The selection of software to provide web services depends on several aspects, such as the programming language and supported features (e.g., asynchronous processing) of the software. It often also depends on the use and the content: Processing services need to be distinguished from the provision of content management services. Cloud-based infrastructures and tools allow the hosting and provision of both data platforms and single services but usage costs need to be taken into account. EO-related web platforms can be based on several architectures, with different EO data, analysis tools, service interfaces, and infrastructures available to users. However, in most cases, they have been built for specific purposes and are therefore limited in terms of the available EO data, analysis tools, and web services.

EO time-series data services and formats

Although commonly used web service specifications for data discovery, access, visualization, and analysis exist, the focus has been mainly on machine-to-machine communication. In particular, the response formats of those web services either focus on web-specific formats (e.g., discovery results in XML or JSON) or are just not suitable for EO time-series data (e.g., OGC WCS for time-series data access), which in most cases is only usable by software developers. Other user personas, such as scientists and thematic experts, need additional output formats. Access to raster time-series data especially cannot be provided in a simple request using the existing services as they only provide access to individual files (multiple files are needed for time-series data). In addition, software developers who are not particularly familiar with the geospatial domain need to learn many specifications to handle data discovery, access, and analysis services. Thus, a uniform service specification and multiple output formats would foster the exploration of EO time-series data with support of multiple user personas.

Evaluation according to user requirements

Web technologies and EO time-series data services and formats have been evaluated in relation to the user requirements presented in Chapter 2.

The following conclusions can be drawn:

- ✓ Web services can be provided with support of different architectures (e.g., SOAP or REST), OGC-compliant specifications, asynchronous execution, and service chaining. However, no software supports all of these features.
- ✓ Multi-source EO data discovery and access and the harmonization of multiple service specifications from different data providers can be realized by means of a service-brokering approach.
- ✓ Several cloud-based infrastructures and processing platforms exist; however, pre-defined platforms always have limitations. As such, as proposed from Soille et al. (2018), the hosting of a self-developed platform offers the most flexible approach.

The following conclusions need to be further explored:

- × Specifications for data discovery and data access provide only different output schemata (e.g., JSON or XML) and do not support multiple output formats. Only the OGC WPS specification supports the provision of multiple outputs.
- × No uniform service specification for data discovery, access, and processing (analysis) is available. Although the OGC specifications follow same rules, many of the core methods and formats for requests and responses are different.
- × Formats for EO time-series data need to be harmonized for further exploration through the user in relation to existing geospatial applications and analysis tools.

Chapter 4: Review of EO Web Services, Tools, and Platforms

In this chapter, web services from existing EO data providers, geospatial tools to process EO time-series data, and cloud-based EO infrastructure platforms are described and evaluated focusing on their current state-of-the-art technologies and in relation to user requirements.

The following questions are reviewed:

- Where and how to search for data and what kind of web services for data discovery are provided? (Section 4.1)
- How to access EO data and what kind of web services are provided? (Section 4.2)
- How to process raster time-series data with geospatial tools? (Section 4.3)
- What kinds of EO data platforms exist and what do they offer? (Section 4.4)

In each of the sections, a conclusion is drawn. The review chapter concludes with a summary and recommendations for multi-source EO data discovery and access, time-series data processing, and cloud-based processing platforms.

4.1 Discovery of EO Time-series Data

Services for the discovery of EO time-series data for Sentinel, Landsat, and MODIS have been made available from space agencies (e.g., ESA and NASA), private companies (e.g., Google and Sinergise), and other organizations (e.g., USGS). As data discovery tools can be provided using different kinds of technologies and specifications, there is no common specification to fit all purposes of data discovery. The main specification for data cataloguing from the OGC, the CSW, has been made available for many satellite data providers through the brokering services offered by CEOS and GEOSS. Although this can be seen as a common approach for data discovery, many data providers either use additional standardized specifications (e.g., OpenSearch and OGC Web Feature Service) or self-developed web service specifications.

The most common services for the discovery of Sentinel, Landsat, and MODIS data archives today are listed in Table 4.1, including the specifications they provide and what kind of EO data is available. Most EO satellite data is not only discoverable by the operator of the satellite. Other data providers hosts copies of the EO data in different kind of formats and provide different kind of services for data discovery. Although standardized specifications for data discovery exist, many data providers, such as USGS, NASA, and Google, offer web services with other specifications that are optimized for their own data archives. In the following subsections, the services listed in Table 4.1 are described and evaluated.

Table 4.1: List of services for satellite data discovery for Landsat, Sentinel, and MODIS.

Data provider	Specifications provided	Data
ESA/Copernicus Open Access Hub https://scihub.copernicus.eu	OpenSearch	Sentinel
USGS Earth Explorer https://earthexplorer.usgs.gov/inventory	Self-developed service specification	Landsat, MODIS, Sentinel–2
NASA CMR https://cmr.earthdata.nasa.gov	Self-developed service specification	Landsat, MODIS
Google Earth Engine https://earthengine.google.com	Python library	Landsat, MODIS, Sentinel
Sinergise Sentinel-Hub http://www.sentinel-hub.com	OGC Web Feature Service	Landsat, Sentinel
GEOSS broker http://www.geodab.net	OGC Catalogue Service for Web OpenSearch, Others (e.g., REST)	Landsat–8, Sentinel
CEOS WGISS Integrated Catalogue http://ceos.org/cwic	OGC Catalogue Service for Web OpenSearch	Landsat–8, MODIS
ESA FedEO http://ceos.org/fedeo	OGC Catalogue Service for Web OpenSearch	Landsat, MODIS, Sentinel

4.1.1 Data provider solutions

Each of the EO data providers offers specific web services for data discovery. In the following paragraphs, services from major public data providers, such as ESA, USGS, and NASA, as well as commercial services from Google and Sinergise are described.

ESA/Copernicus Open Access Hub (Sentinel)—OpenSearch

ESA provides the REST-based OpenSearch specification for the discovery of EO data from Sentinel satellites.[14] Search parameters, such as dates, orbits, sensor modes, cloud cover, and a full-text search, can be used in addition to mission-specific parameters to filter the discovery request (Table A.1). An individual polygon can be used to intersect spatially with the scene of the EO data. Either XML or JSON can be requested as a data format for the response using the additional parameter "format." The response includes all available properties for each satellite scene, including the identifier, sensor mode, footprint, product type, size, dates, as well as mission-specific properties (e.g., polarization mode for radar data and cloud cover percentage for optical data). However, the quality and format of quick look images can be improved for some satellite missions (e.g., Sentinel–1). Furthermore, a user account is necessary and only two parallel service requests are allowed.

USGS Earth Explorer (Landsat, MODIS, Sentinel–2)—self-developed REST API

The inventory service of the USGS Earth Explorer allows searching for dataset collections and satellite scenes.[15] Spatial and temporal filters can be used for both search requests (Table A.2 and Table A.3). Cloud coverage and individual months can be set for satellite-scene searches. Furthermore, additional criteria can be used to filter the satellite-scene search using dataset-related parameters. As the USGS Earth Explorer consists of a wide range of EO data (e.g., Landsat, MODIS, and Sentinel–2), multi-source EO data can be searched. The search parameters are encoded as JSON and sent to an individual web service endpoint for login, collection search, or satellite-scene search. The response is encoded as JSON. Each of the resulting scenes includes the following information: start time, end time, polygon footprint, a bounding box, quick look image link, data ordering link, data download page link, scene identifier, and metadata URL. However, only a non-standardized web service interface is available and an interface key is required to conduct requests to the service. This key can be obtained using the login web service with a username and password. The full metadata information is only available when requesting an additional external metadata URL.

[14] https://scihub.copernicus.eu/dhus/search
[15] https://earthexplorer.usgs.gov/inventory

NASA CMR (Landsat, MODIS)—self-developed REST API

The Common Metadata Repository (CMR) of NASA contains metadata records for EO data.[16] Landsat and MODIS data are included in addition to others. A list of dataset collections can be searched using multiple filter parameters (Table A.4). Based on a collection identifier, the individual satellite scenes (granules) can be searched using further parameters (Table A.5). Different kinds of output formats can be requested for the responses (e.g., CSV, JSON, Keyhole Markup Language, and XML). With two web service endpoints, dataset collections and satellite scenes can be searched. Each of the resulting scenes includes the following information: start time, end time, polygon footprint, scene identifier as title, quick look image link, and data download link. Although filtering through cloud coverage can be effected within the request, this information is only available after requesting the full metadata record. The services can be used without a user login. However, only a fixed set of standardized filter parameters is available and no collection-specific parameters can be used to query the catalogue. The responses include only a limited set of parameters; a further request needs to be sent to retrieve the full metadata record.

Sinergise Sentinel-Hub (Sentinel, Landsat)—OGC WFS

The Sentinel-Hub infrastructure from Sinergise provides web services that are compliant with the OGC for data discovery.[17] Although the common standard for providing data discovery is the OGC CSW, Sinergise publishes the OGC Web Feature Service (WFS), which is used for vector data access, to provide satellite-scene discovery. The WFS includes a list of dataset collections, which is available with a “GetCapabilities” request. To retrieve the scenes available for a specific dataset collection, the parameters from Table A.6 can be used to filter them with a spatial bounding box within the “GetFeature” request. The response is available as XML or in GeoJSON formats, which are both standard formats for the OGC WFS specification. Each of the resulting scenes include only a few metadata: date, time, coordinate reference system (crs), a bounding box, cloud cover percentage, scene identifier (format dependent from satellite mission), polygon footprint, and path to local data. However, only a limited set of parameters can be used to discover scenes and only a limited set of metadata is provided in the results. Additional metadata need to be requested from external services. A commercial interface key is required to use the services.

[16] https://cmr.earthdata.nasa.gov
[17] https://www.sentinel-hub.com/develop/documentation/api/ogc_api/wfs-request

Google Earth Engine (Sentinel, Landsat, MODIS)—Python API

Google provides discovery, access, and analysis tools for a broad range of geospatial datasets, including EO data archives from USGS Landsat, ESA Sentinel, and several MODIS products (Gorelick et al. 2017). All tools are available using the web-based JavaScript editor (Playground) and the Python library '*earthengine-api*',[18] which is based on web services. A whitelisted service account is needed to run the Python Earth Engine API in automated workflows (e.g., within a web processing service). Within Google Earth Engine, satellite scenes can be filtered from dataset collections using several metadata properties (e.g., cloud cover, temporal and spatial boundaries, or any metadata item). In addition, individual properties can be calculated and used to filter the collection. No links to external quick look images or data download links are included in the metadata as both of these can be directly computed within and provided by Google Earth Engine. Listing A.1 shows the request of a Sentinel–1 collection filtered by point geometry and additional properties of the dataset (VV polarization and descending orbit direction). The response within the Python API is a list of objects, which can directly be further processed in Python. However, so far, only the Python library is available for use in self-developed applications. Therefore, these applications need to be developed in Python or a Python script needs to be available, which is executed through the command line.

4.1.2 Brokered web service solutions

Web service brokering describes the ability to search external services, which have been connected seamlessly to the broker. Brokering solutions offer the possibility of providing harmonized interfaces to connected data providers. In the following paragraphs, brokering services for data discovery from international organizations are described and evaluated.

CEOS WGISS Integrated Catalog (CWIC)—OGC CSW, OpenSearch

The CEOS Working Group on Information Systems and Services (WGISS) provides with CWIC[19] an integrated data catalogue for EO data providers based on a brokering approach (Shao et al. 2013). The broker includes metadata catalogues from various organizations, such as NASA (USA), USGS (USA), the Group for High Resolution Sea Surface Temperature, the European Organisation for the Exploitation of Meteorological Satellites, and further international space agencies from India, Brazil, China, and Canada. This includes data from Terra, Aqua (both carrying a MODIS sensor) and Landsat–8 satellites. Standardized interfaces, such as OGC CSW and OpenSearch, are provided through CWIC. The OpenSearch interface has been tested for Landsat–8 data. Filters,

[18] https://pypi.org/project/earthengine-api/
[19] http://ceos.org/ourwork/workinggroups/wgiss/access/cwic/

such as a bounding box, geometry, start time, and end time, can be applied to the discovery request. Each of the resulting Landsat–8 scenes is described with the following information: title (this relates to the unique identifier of each Landsat scene), CWIC identifier, date, collection identifier, data center, and polygon footprint. Additional external links are provided to access the data download page, quick look image, and full metadata record. Although there are a good number of brokered resources from international space agencies, there is currently no support for Sentinel satellites. Furthermore, it is not obvious that historical Landsat missions are available through CWIC as they do not appear on the list of dataset collections. However, Landsat data is registered in NASA's CMR, which is brokered by CWIC. The search functionalities are limited (e.g., no filtering for cloud cover). The use of this broker depends on the satellites users are interested in and whether they have been integrated in CWIC.

GEODAB (Sentinel, Landsat, MODIS)—OGC CSW, OpenSearch

The GEO Discovery and Access Broker (GEODAB; Nativi et al., 2014) acts as a brokering service to mediate between and harmonize metadata and catalogue standards. Currently, more than 150 data catalogues are registered with the broker. In addition to geospatial data catalogues, data catalogues from EO data providers have been made available through GEOSS, such as the Copernicus Open Access Hub from the ESA, the FedEO from CEOS, USGS Landsat–8, and CWIC. Several REST-based resource interfaces are available[20], such as the OGC CSW and OpenSearch. Several search filters can be applied to the discovery request, such as relative orbit, product type, product level, sensor operation mode, sensor swath, processing level, cloud cover percentage, and polarization mode. Additional parameters can be set, but the description shows only abbreviations, such as "illazan," "illzean," "sarPolCh," without any further description. The resulting metadata for a Sentinel–2 scene includes the following data: a bounding box, start/end date, platform name, instrument, instrument operation mode, product type, cloud cover percentage, relative orbit, polygon footprint, processing baseline and level, orbit direction, start orbit number, download link, and several OGC WMS layers added by GEOSS and linked to the Sinergise Sentinel-Hub services. Additional metadata for other satellites (e.g., Sentinel–1) can be included in the request to the service. The scene identifier from the original data provider is not provider, thus the subsequent retrieval of metadata from the original data provider is not possible. Although the GEODAB can be used to search for satellite data in different data catalogues, the resulting metadata per scene is limited. In most cases, further metadata needs to be requested using other services.

[20] http://production.geodab.eu/gi-cat-StP/ and https://www.geodab.net/apis

ESA FedEO (Landsat, Sentinel, MODIS)—OGC CSW, OpenSearch

The FedEO established by ESA provides a unique entry point for multi-source satellite missions.[21] FedEO was initially developed as a prototype for GEOSS and has been provided since 2012 by CEOS. It includes ESA EO data archives (e.g., Copernicus satellites, Landsat data at ESA, and historical ESA SAR missions) as well as EO missions from Canada, which is an associated ESA member state. In addition, NASA CMR has been integrated into FedEO, which provides access to further EO data, such as Landsat and MODIS data. OpenSearch and OGC CSW interfaces are provided through the FedEO web service infrastructure. Multiple response formats are available, such as Atom, RDF, JSON-LD (JSON for Linked Data), and GeoJSON, all of which are suitable formats for developers. A 'description document' lists some available dataset collections as "parentIdentifier," which needs to be used in the search for satellite scenes. Although the dataset collections from NASA CMR and from ESA/Copernicus Open Access Hub can be searched, the collections are not shown in this list. Thus, users need to know in advance how to search for collections from NASA CMR. Filters, such as cloud cover, orbit direction, swath identifier, orbit number, start date, end date, and geometry, can be applied when requesting the service. Each of the resulting Sentinel–2 scenes is described with the following information: start time, end time, instrument, sensor type, operational mode, orbit number, orbit direction, polygon footprint, quick-look image link, data download link, cloud cover percentage, scene identifier, product type, relative orbit number, and tile identifier. The resulting metadata includes a suitable amount of information, although this is dependent on the brokered service provider. In general, no user login is necessary to use this service. However, this may be necessary for some brokered services (e.g., Copernicus Open Access Hub).

4.1.3 Conclusions

The use of the FedEO brokering service from ESA seems to be the best brokering solution in terms of EO data availability and available metadata in request filtering and output responses. The ESA FedEO broker provides access to Landsat, Sentinel, and MODIS and thus provides a harmonized web service interface for multi-source EO data. However, a user login is necessary for some services and output formats are only suitable for users who are familiar with XML or JSON formats. Although the ESA FedEO broker seems to be quite unknown in science, its major focus is on providing a web service interface, which can be used by software developers.

[21] http://fedeo.esa.int/opensearch/readme.html

The use of the Google Earth Engine Python API allows the integration of on-demand processing steps while conducting the data discovery request (e.g., cloud cover percentage of a specific area of interest). Although Sentinel, Landsat, and many MODIS products are available, some specific drawbacks can be found:

- The service interface is currently only available in Python scripting language.
- The provision of quick-look images needs to be processed for each image when conducting the data discovery request.

Table 4.5a lists a comparison between user requirements and the discovery services described in this section. For most of the discovery services—except for the brokering services and NASA CMR—user logins are necessary, which require multiple user accounts to query multi-source datasets. The output format of the discovery services is mainly based on XML and JSON structures, which are optimized for developers but not for other users. Although asynchronous execution of these services is not available, all of them deliver the response immediately so that no asynchronous execution is necessary. Only a few provide OGC-compliant specifications. From the point of view of specific EO dataset collections, the following services can be recommended:

- MODIS data in the NASA CMR
- Landsat data (USGS archive) in the USGS Earth Explorer
- Landsat data (ESA archive) in ESA FedEO or Sentinel-Hub OGC service
- Sentinel data in the ESA/Copernicus Open Access Hub

Advantages as well as limitations for most of the services need to be considered:

- The USGS Earth Explorer does not deliver all the metadata in the initial discovery request. Thus, the full metadata records need to be requested afterwards.
- Quick-look images for Sentinel data can be retrieved from NASA Alaska Satellite Facility (Sentinel–1) and USGS Earth Explorer (Sentinel–2) without user login, in contrast to the ESA/Copernicus Open Access Hub.
- Sentinel–1 quick-look images from NASA Alaska Satellite Facility provide better quality and geo-referenced images than ESA/Copernicus.
- Sentinel-Hub OGC services provide only a too-small set of metadata; most metadata needs to be retrieved from other data providers.

4.2 Access to EO Time-series Data

Most EO data can be accessed through different data providers with various services and tools. In addition to simple scene downloads, further services exist to download preprocessed satellite scenes. As none of the data providers holds the archives for all EO data, several services need to be requested when providing access to multi-source EO datasets.

Table 4.2 lists selected data providers for Landsat, MODIS, and Sentinel data with their service specification provided for data access. While cloud infrastructure providers, such as Amazon Web Services and Google Cloud Storage, offer only scene downloads, this data can be accessed directly within the virtual environments without the need to download the data. Sinergise Sentinel-Hub OGC services provide standard-compliant access and visualization. When requesting data, further processing tasks can be undertaken on demand. Similarly, Google Earth Engine provides server-processing possibilities before data is downloaded. The resulting data can be accessed through the Google Earth Engine Python API. The USGS ESPA service provides an automated service for ordering Landsat and MODIS time-series data preprocessed on demand depending on the parameters selected by the user. As the request only submits an order to the service, data cannot be accessed directly, but only when the order has been processed and scenes ordered are ready for download.

Table 4.2: List of web services for Landsat, Sentinel, and MODIS satellite data access.

Data provider	Specifications provided	Data
ESA/Copernicus Open Access Hub https://scihub.copernicus.eu	Scene download via Open Data Protocol	Sentinel
USGS Earth Explorer https://earthexplorer.usgs.gov/inventory	Scene download via HTTP URL	MODIS
USGS ESPA https://espa.cr.usgs.gov	On order via REST-based API	Landsat, MODIS
NASA CMR https://cmr.earthdata.nasa.gov	Scene download via HTTP URL	MODIS
Google Earth Engine https://earthengine.google.com	Python API	Landsat, MODIS, Sentinel
Sinergise Sentinel-Hub http://www.sentinel-hub.com	OGC Web Coverage Service, Feature Information Service	Landsat, MODIS, Sentinel
Amazon Web Services https://registry.opendata.aws/landsat-8/	Scene download via HTTP URL	Landsat–8
Google Cloud Storage https://cloud.google.com/public-datasets/	Scene download via HTTP URL	Landsat, Sentinel–2

4.2.1 Data access services

In the following paragraphs, specific services from data providers are described and evaluated. The focus in this section is on web services—hence, data providers that only provide scene downloads via HTTP URLs are omitted.

ESA/Copernicus Open Access Hub (Sentinel)—Open Data Protocol

The Sentinel data from the EU Copernicus program are provided with the Open Data Protocol from the Copernicus Open Access Hub managed by ESA. The interface builds on HTTP and REST-based methods. In addition to data access, the Open Data Protocol allows for the querying of the data catalogue and the filtering of search results. Various response formats are supported, such as XML, JSON, and CSV. Datasets are described with metadata and provided as download files. In addition, quick look images can be retrieved using this service. The metadata of a specific product identifier can be queried with a defined URL. Additional links for data download and quick-look images are included in the response. User credentials are needed for each data download. Only two simultaneous data downloads are currently allowed. Since September 2018, not all of the data is kept online, but can be ordered from the Copernicus Long Term Archive.[22]

USGS ESPA (Landsat, MODIS)—self-developed REST API

Preprocessed Landsat and MODIS data can be ordered using the Science Processing Architecture (ESPA) interface of the USGS Earth Resources Observation and Science Center. In the course of the ordering process, functions to reproject, subset, resize, and convert to specific data formats can be optionally added (Table A.7). As an example, this allows to the clipping of the EO data to the area of interest. Prior to the request to USGS ESPA, the identifiers of satellite scenes to be ordered need to be sought elsewhere (e.g., through the USGS Earth Explorer, as described in the previous section). For each request, the username and password from USGS need to be sent. After the order has been placed, a web service request can be used to monitor its status and to download scenes once the processing has been completed. With USGS ESPA, the complete process, from ordering until download when data is ready, can be automated. However, users need to wait until the processing has been completed in the USGS ESPA processing queue.

Google Earth Engine (Sentinel, Landsat, MODIS)—Python API

In addition to the data discovery functions described in the discovery review (Subsection 4.1.1), data access is also available using the Google Earth Engine Python API. A list of scenes filtered by user-defined options can be exported, downloaded, or further processed using Python. Prior to export and download, data can be processed, for example, by

[22] https://scihub.copernicus.eu/userguide/LongTermArchive

extracting values for points or statistical summaries for polygons, calculating indices, or conducting further computations, such as clipping, masking, and mathematical operations. The data can either be exported to Google Drive (the preferred approach) or downloaded by generating a HTTP link (though there are size limitations). The individual values of both pixel and raster time-series can be exported as objects using the Python library. Using the Python API in an automated workflow or in web services requires a service account, which needs to be whitelisted by Google.

Sinergise Sentinel-Hub (Sentinel, Landsat)—OGC WCS, FIS

The Sentinel-Hub infrastructure from Sinergise provides services compliant with the OGC for data access. The OGC WCS is provided for raster data download for a given temporal period and spatial subset. Table A.8 lists the parameters for the WCS request for the Sentinel-Hub services. The resulting data is a raster dataset containing a mosaic of the selected dataset collection within the time ranges selected by the user (the TIME parameter of the WCS request). In the course of the request, further processing, such as classifications and indices calculation, can be integrated (see the EVALSCRIPT parameter in Table A.8). For access to raster time-series data, individual requests need to be conducted for each date. The Sentinel-Hub Feature Information Service performs statistical computations on the area of interest requested. For each image in the filtered data collection, statistical values are calculated, such as the mean, minimum, maximum, and standard deviation. Table A.9 lists the request parameters available for defining a dataset collection (e.g., layer and style), computation parameters (e.g., resolution and geometry), as well as filtering parameters (e.g., time, geometry, and maximum cloud cover).

4.2.2 Data download links

Today, many satellite scenes are no longer only available on the data archive of the satellite's operator (e.g., Sentinel data on the ESA/Copernicus Open Access Hub). Additional third-party data providers, such as Google, Amazon, as well as national EO data archiving centers, provide access to data using different technologies. For example, Google Cloud hosts the complete Landsat archive in a web-based folder structure, which allows direct access to individual scenes and bands in an automated manner without the need for users to login. As such, the use of third-party data providers can be useful for specific applications. In the following paragraphs, download links for complete satellite scenes for the German Copernicus Data and Exploitation Platform, Google Cloud Storage, and USGS Earth Explorer are described.

German Copernicus Data and Exploitation Platform

The German Copernicus Data and Exploitation Platform (CODE-DE)[23] provides discovery and access tools for worldwide Sentinel data. A user login is required for data download. As CODE-DE provides a rolling archive, not all the scenes are available (Reck et al. 2019). The length of the rolling archive period depends on the geographic region and the EO mission. The files are stored in the same format as on the ESA/Copernicus Open Access Hub, providing a zipped archive file. The scenes can be accessed with the URLs below and include variables for the year, month, and day of the scene acquisition date and the scene identifier (Reck et al. 2019):

```
https://code-de.org/Sentinel1/{year}/{month}/{day}/{id}.zip
https://code-de.org/Sentinel2/{year}/{month}/{day}/{id}.zip
```

Google Cloud Storage

The complete Landsat archive[24] as well as Sentinel–2 data,[25] with access to individual bands, are available on Google Cloud. Instead of a zipped archive file, all individual files can be accessed directly. This may be useful depending on the application. For example, if an NDVI needs to be calculated, only the red and near-infrared bands need to be downloaded. The URLs below can be used to access the data with the following variables: “path” as WRS path, “row” as WRS row, “utm” as UTM code, “lat” as latitude band, “grid” as grid square, and “id” as the scene identifier. Various EO data and specific products are available. No login is required to access the direct download links for EO data on the Google Cloud Storage.

```
https://storage.googleapis.com/gcp-public-data-landsat
       /LC08/01/{path}/{row}/{id}/*        → Landsat-8
       /LE07/01/{path}/{row}/{id}/*        → Landsat-7 ETM+
       /LT05/01/{path}/{row}/{id}/*        → Landsat 5 TM
       /LT04/01/{path}/{row}/{id}/*        → Landsat 4 TM
       /LM01/PRE/{path}/{row}/{id}/*       → Landsat 1 MSS
       /LM02/PRE/{path}/{row}/{id}/*       → Landsat 2 MSS
       /LM03/PRE/{path}/{row}/{id}/*       → Landsat 3 MSS
       /LM04/PRE/{path}/{row}/{id}/*       → Landsat 4 MSS
       /LM05/PRE/{path}/{row}/{id}/*       → Landsat 5 MSS

https://storage.googleapis.com/gcp-public-data-sentinel-2
       /L2/tiles/{utm}/{lat}/{grid}/{id}/* → Sentinel-2 Level 2
       /tiles/{utm}/{lat}/{grid}/{id}/*    → Sentinel-2 Level 1
```

[23] https://code-de.org
[24] https://cloud.google.com/storage/docs/public-datasets/landsat
[25] https://cloud.google.com/storage/docs/public-datasets/sentinel-2

USGS Earth Explorer

Landsat (LM0*, LT0*, LE07, LC08) and Sentinel–2 (S2) scenes are available on the USGS Earth Explorer. While the URLs below can be accessed without a user login, for the actual download, a user does need to be logged in to the USGS Earth Explorer. Based on the collection identifier (marked bold in the URLs) and the scene identifier (variable “id”), the download pages for individual scenes can be accessed. For Sentinel, the internal scene identifier used within Earth Explorer is needed (variable “usgs_id”).

```
LC08: https://earthexplorer.usgs.gov/download/12864/{id}/STANDARD/INVSVC
LE07: https://earthexplorer.usgs.gov/download/12267/{id}/STANDARD/INVSVC
LT0*: https://earthexplorer.usgs.gov/download/12266/{id}/STANDARD/INVSVC
LM0*: https://earthexplorer.usgs.gov/download/3120/{id}/STANDARD/INVSVC
S2  : https://earthexplorer.usgs.gov/download/10880/{usgs_id}/STANDARD/INVSVC
```

4.2.3 Data extraction services

Some applications require a direct extraction of time-series data from global EO data archives to be retrieved in a few seconds for direct visualization and further processing in mobile or web applications. Software developers can make use of data extraction services to build applications based on EO time-series data and processing tools. Only two of the previously mentioned data providers can directly extract values of EO time-series data in their catalogue without setting up own services or downloading full satellite scenes (Table 4.3): Sinergise Sentinel-Hub, with the Feature Information Service, and Google Earth Engine with the Python API, which can be integrated within a web service. Examples of the data extraction services for the Sentinel-Hub services and Google Earth Engine are provided in Section A.2 in the appendix.

Table 4.3: Data providers with services for the direct extraction of time-series data.

Data provider	Datasets	Features
Sinergise Sentinel-Hub Feature Information Service	Sentinel–1, Sentinel–2, Sentinel–3, Landsat–8, Landsat 5–7 (ESA archive)	Point extraction, area statistics (mean, max, min, standard deviation), area raster extraction
Google Earth Engine Python API	Sentinel–1, Sentinel–2, Landsat 4–8 (USGS archive), MODIS, etc.	Point extraction, area statistics (mean, max, min, standard deviation, percentiles), area raster extraction, individual calculations

4.2.4 Conclusions

EO data access is available in various ways, such as direct HTTP links, standardized web services, and ordering web services. Most of the scenes are available by means of scene downloads or by using cloud processing environments (Amazon and Google). Only services from the commercial provider Sinergise or with the use of the Rasdaman software allow direct EO data access based on standardized web services. Google Earth Engine provides a Python API to access and process data. Although the order and download approach of USGS ESPA does not provide immediate access to EO data, preprocessed datasets are available based on users inputs (e.g., clipping or reprojection). Direct extraction services are useful for many applications but are offered by only two data providers (Sinergise and Google Earth Engine). As there is no simple standard for those extraction services, the request and response formats are diverse. Table 4.5b lists a comparison between user requirements and the access services described in this section. Only a few services make use of standardized specifications, in particular there is only the data provider Sinergise, which offers OGC-compliant services for data access. Asynchronous access to EO data is only available from USGS ESPA and Google Earth Engine. No alternative data formats, such as summarized results, are available by default, but can be generated through Google Earth Engine. In addition, only a few services (e.g., Sentinel-Hub, USGS ESPA) provide the data within additional formats for geospatial data (e.g., GeoTIFF instead of JPEG2000).

MODIS data can be accessed using download links to direct HTTP URLs from its original data providers. Only the original data are provided—this needs to be processed further to meet the requirements of users. For Landsat data, the best option to download data to a local computer is either the use of the USGS ESPA web service, which allows the conducting of preprocessing steps before downloading the data, or access through cloud providers, such as Amazon, Google, and Copernicus DIAS. For Sentinel data, availability also depends on the data provider (e.g., USGS Earth Explorer provides access to Sentinel–2 data). While ESA provides a protocol with only scene download, the commercial Sinergise Sentinel-Hub Services provides access with the OGC WCS specification, which allows the subsetting of the data. In addition, further processing steps can be integrated into the request.

All the data relevant to this thesis can also be accessed using the Google Earth Engine Python API. This interface is extremely suitable for immediate time-series extraction of single pixels or small areas. However, limitations occur when downloading large amounts of time-series data (e.g., enough storage space on Google Drive or file-size limits for direct HTTP downloads).

4.3 Processing of EO Time-series Data

The processing of raster time-series data is an essential step in producing derived geospatial results from EO data. Work with spatial time-series data in particular needs to be explored, as users need knowledge of the processing of multi-dimensional data when dealing with large datasets with temporal and spatial extents. In the subsequent subsections, popular programming languages and geospatial tools that can analyze raster time-series data are reviewed with focus on their handling of raster time-series data.

4.3.1 Programming languages

Two commonly used programming languages, Python and R, are reviewed for their handling of EO (raster) time-series data. For both languages, the relevant libraries to read and analyze time-series data are described. A summary shows the requirements for time-series data structures for each of the languages.

Python

Several Python libraries exist for working with spatial and time-series data, such as *Pandas* (Python Data Analysis Library) and *GeoPandas* for single time-series, *NumPy* for raster matrix operations, and *xarray* for multi-dimensional raster time-series operations.

The Pandas library provides high-performance data structures and data analysis tools for the Python programming language (McKinney 2011). Relevant data formats can be read and written, such as CSV, Microsoft Excel, and databases. With *DataFrames* in Pandas, matrix data can be analyzed and manipulated. Using the GeoPandas extension, geospatial data is supported and spatial operations on geometric types can be conducted within the data structures. Listing 4.1 shows an example script for visualizing single time-series data extracted for a pixel of a raster time-series stack.

The Python library xarray provides access and analysis tools for multi-dimensional data (Hoyer & Hamman 2017). The project aims to provide a “pandas-like” toolkit for the analysis of multi-dimensional arrays, rather than tabular data, which is the focus for Pandas. Although the initial focus was on netCDF data structures, extensions have been integrated for working with GDAL-compatible data formats (e.g., GeoTIFF) with the introduction of the *open_rasterio* function. xarray makes internal use of the NumPy library for multi-dimensional data manipulation. Listing 4.2 shows an example Python code for loading a raster time-series dataset. For working with external data formats, such as GeoTIFF, a list of files (each file for a date, see Line 5) and a list of dates related to the files (Line 7) is required to combine the individual two-dimensional raster arrays to a three-dimensional time-series data frame (Line 8). In this example, the list of dates is extracted from the filenames.

The following requirements for raster time-series data structures within Python can be derived:

- Multi-band geospatial file (each band relating to a date) or multiple geospatial files (each file relating to a date) converted to a list of files.
- Variables with a list of dates in the same order as the bands in the geospatial file or the order in the multiple geospatial file list.

```
# extract pixel time-series from NumPy array
item = arr[:, 301:302, 301:302].reshape((arr.shape[0])).tolist()

# read dates from csv
df_dates = pandas.read_csv('/MOD13Q1.EVI.csv', sep=";")

# create Pandas dataframe using pixel time-series and dates column
df = pandas.DataFrame(item, index=list(df_dates['date']), columns=['evi'])
df.index = pandas.to_datetime(df.index, format='%Y-%m-%d')

# plot data
df.plot()
```

Listing 4.1: Python source code for working with Pandas after extracting a single time-series pixel (item) from a raster time-series stack (arr). With Pandas, CSV files or list of objects can be easily read, plotted, and further analyzed.

```
01 import glob
02 import pandas as pd
03 import xarray as xr
04
05 filenames = sorted(glob.glob('*.EVI.*.tif'))
06
07 time = xr.Variable('time', pd.DatetimeIndex([
           pd.Timestamp.strptime(f[9:16], '%Y%j') for f in filenames]))
08 da = xr.concat([xr.open_rasterio(f) for f in filenames], dim=time)
09 da
10
11 <xarray.DataArray (time: 389, band: 1, y: 694, x: 855)>
12 Coordinates:
13 * band     (band) int64 1
14 * y        (y) float64 5.743e+06 5.743e+06 5.743e+06 5.742e+06 ...
15 * x        (x) float64 6.89e+05 6.892e+05 6.895e+05 6.897e+05 6.899e+05 ...
16 * time     (time) datetime64[ns] 2000-02-18 2000-03-05 2000-03-21 ...
17 Attributes:
18  transform:   (688899.88262, 231.656086344, 0.0, 5743120.7846271 ...
19  crs:         +a=6371007.181 +b=6371007.181 +lon_0=0 +no_defs +proj=sinu
20  res:         (231.65608634466076, 231.49636861726358)
21  is_tiled:    0
22  nodatavals:  (-3000.0,)
```

Listing 4.2: Example work with the xarray library using raster time-series data. Dates are extracted from the filenames (Line 7).

R Project for Statistical Computing

Within the statistical language R, three packages can be considered when working with spatial time-series data: 1) the “raster” package, with the *RasterStack* object for dealing with multi-band geospatial raster data, and 2) the “stats” package, with the *ts* function for dealing with temporal information. Both are used with scientific time-series analysis tools, such as BFAST and Greenbrown. 3) The “rts” package, with the *Raster*TS* object for dealing with spatial raster time-series data.

A *RasterStack* combines raster objects with the same spatial extent and resolution. It can be automatically created using either a multi-band raster file (each band results in an individual raster object within the stack) or individual raster files. For multi-band files, a *RasterBrick* can be used instead. This leads to performance increases when doing calculations but limits the object to just a multi-band file instead of using multiple raster files at once. Within R, the *calc* function can be used to conduct an operation for each pixel time-series within a collection of raster objects (*RasterStack* or *RasterBrick*) and results in a single raster output (e.g., the temporal mean of each pixel time-series). Listing 4.3 shows the source code of an example that loads a multi-band raster file into a *RasterBrick* object.

R software comes with a large set of time-series tools. As such, the class *ts* from the “stats” package can be used for regularly spaced single time-series. The parameters start and frequency define the start and end of the time-series; start is the time of the first item of the time-series, and frequency is the number of items per year (e.g., 1 = annual time-series, 12 = monthly time-series, 365 = daily time-series). For irregular spaced time-series, the class *irts* in the “stats” package can be used with the vector objects ‘time’ and ‘value’ as parameters. An example of the *ts* class is shown in Listing 4.4.

A *Raster*TS* object from the “rts” package automatically combines a multi-layer raster object (*RasterStack* or *RasterBrick*) with temporal information. Both the multi-layer raster object and a vector with temporal information need to be passed to the *rts* function to create a *Raster*TS* object (Listing 4.5).

For both the *RasterStack* with the *ts* object and the *Raster*TS* object, the following requirements for data structures can be derived:

- A multi-band geospatial file, in which each band relates to a date, is converted to a *RasterStack* or *RasterBrick* object.
- A vector with dates, where the order of the dates needs to be the same as the bands or layers of the raster object, is needed for *Raster*TS* and the irregular *irts* function.
- For regular time-series, the *ts* object with the parameters start and frequency is needed.

```
1 data <- stack("SWOS_WQ_CDOM_FUB_Monthly_France_Berre.vrt")
2 data
3
4 class       : RasterStack
5 dimensions  : 125, 135, 16875, 118  (nrow, ncol, ncell, nlayers)
6 resolution  : 0.003593245, 0.003593245  (x, y)
7 extent      : 4.807761, 5.292849, 43.30219, 43.75135
8 coord. ref. : +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84
                +towgs84=0,0,0
9 names       : SWOS_WQ_C//ce_Berre.1, SWOS_WQ_C//ce_Berre.2, …
```

Listing 4.3: Example of working with multi-band raster data using *RasterBrick*.

```
1 data <- brick("SWOS_WQ_CDOM_FUB_Monthly_France_Berre.vrt")
2 data <- as.vector(extract(data, SpatialPoints(cbind(5.1, 43.5))))
3 data_ts <- ts(data, start=c(2002, 6), frequency = 12)
```

Listing 4.4: Example of working with single time-series data using the *TS* function.

```
01 data <- stack('SWOS_LSTT_AL_Oued-Sebaou_2000-2016.vrt')
02 data
03
04 class       : RasterStack
05 dimensions  : 59, 73, 4307, 781  (nrow, ncol, ncell, nlayers)
06 resolution  : 1000, 1000  (x, y)
07 extent      : -1050090, -977090.1, 4166999, 4225999
08 coord. ref. : +proj=utm +zone=34 +datum=WGS84 +units=m +no_defs
           +ellps=WGS84 +towgs84=0,0,0
09
10 dates
11 [1] "2000-03-05" "2000-03-13" "2000-03-21" "2000-03-29" "2000-04-06"
...

21 library(rts)
22 rasterTS <- rts(data, dates)
23 rasterTS
24
25 Raster Time Series with monthly periodicity from 2000-03-05 to 2017-02-26
26 class             : RasterStackTS
27 raster dimensions  : 59, 73, 4307, 781  (nrow, ncol, ncell, nlayers)
28 raster resolution  : 1000, 1000  (x, y)
29 raster extent      : -1050090, -977090.1, 4166999, 4225999
30 coord. ref. : +proj=utm +zone=34 +datum=WGS84 +units=m +no_defs
           +ellps=WGS84 +towgs84=0,0,0
```

Listing 4.5: Example of working with spatial time-series data using *Raster*TS* object

4.3.2 Geospatial tools

Geospatial tools allow the processing and analysis of EO time-series data. In the following paragraphs, a few geospatial tools and their handling of raster time-series data are described and the requirements for time-series data structures are summarized.

TIMESAT

Time-series data can be used within TIMESAT (Eklundh & Jönsson 2017) in different kinds of data formats. Whereas individual time-series data can be provided in a text data file (Figure 4.1, right), processing steps are required for raster time-series data. Raster time-series data needs to be processed to a headerless binary format (e.g., ENVI HDR). Each individual date needs to be represented by a raster file. An image file list in the form of a text file contains the number of images (first line) and the relative path to the files for each individual image in chronological order (Figure 4.1, left). As there is no provision of dates within TIMESAT, the resulting outputs refer only to the index as date. This needs to be converted to individual dates afterwards (Eklundh & Jönsson 2017).

```
391
MOD13Q1.A2000049.EVI.tif.bil
MOD13Q1.A2000065.EVI.tif.bil
MOD13Q1.A2000081.EVI.tif.bil
MOD13Q1.A2000097.EVI.tif.bil
MOD13Q1.A2000113.EVI.tif.bil
MOD13Q1.A2000129.EVI.tif.bil
MOD13Q1.A2000145.EVI.tif.bil
...
```

```
15 23 1
0.2362 0.2540 0.2665 0.2641 0.5325 0.5491 0.6073 0.5678 ...
```

Figure 4.1: Text file for spatial raster time-series data structure (left); text file for single time-series data structure (right).

The following requirements for data structures for TIMESAT to analyze raster time-series data can be derived:

- A single geospatial data file for each date and a text file containing the number of files and each filename on a separate line (Figure 4.1, left).

GRASS GIS

With open source GRASS GIS software, several processing tools for raster time-series data are available (Neteler et al. 2012). To enable users to use these tools, a space-time dataset needs to be created (Gebbert & Pebesma 2014). Listing 4.6 shows example processing steps: 1) Import each raster per date into GRASS GIS as a map, 2) create a new space-time dataset, and 3) register previously imported maps with dates in the new space-time dataset. For the last step, temporal information for each of the maps is needed—either in a separate text file or as increment value for continuous time-series. In Listing 4.6, a separate text file (.csv) has been used, which contains the name of the map and the date in CSV format.

```
# 1) import data
for file in SWOS_LSTT_ES_Fuente-de-Piedra_20*.tif; do
 r.in.gdal input=$file output=$file;
done

# 2) create new space-time dataset
t.create output=modis title="MODIS LST" description="MODIS LST"

# 3) register maps into space-time dataset
t.register input=modis file=SWOS_LSTT_ES_Fuente-de-Piedra_2000-2016.csv
```

Listing 4.6: Example of working in GRASS GIS with spatial time-series data

The following requirements for data structures to create a space-time dataset in GRASS GIS can be derived:

- A single geospatial data file for each date.
- A text file with relation between the GRASS GIS map name and date.

Open Data Cube

CEOS Open Data Cube software[26] allows the managing and analyzing of raster time-series data based on preprocessed raster data in netCDF format using the Python *xarray* library. To ingest data into the Open Data Cube software, individual datasets need to be registered and ingested. For each date, individual geospatial files (one per band and date) need to be registered. Additionally, a metadata file in text format (.yaml), which includes references to the files of each band, is needed for the data ingestion for each date. Listing 4.7 shows an example workflow using the *datacube* command-line executable: 1) A product type is added, 2) configuration YAML files are created for each satellite scene, 3) these are registered with the database, and 4) the data can finally be ingested into file structure of the Open Data Cube.

```
# 1) add product type to data cube
datacube product add "ls8_collections_sr_scene.yaml"

# 2) create YAML files for each dataset
python "usgs_ls_ard_prepare.py" "MODIS_*.tif"

# 3) add YAML files to data cube
datacube dataset add MODIS_*.yaml --auto-match

# 4) ingest data into data cube format based on YAML description file
datacube -v ingest -c "ls8_collections_sr_fuente_de_piedra_example.yaml"
```

Listing 4.7: Example workflow for ingesting raster time-series data into Open Data Cube.

[26] https://www.opendatacube.org

The following requirements for data structures to ingest raster time-series data into the Open Data Cube software can be derived:

- A YAML description file for each dataset collection.
- A single geospatial data file for each date (one per band), accompanied by a YAML text file (one per dataset including all bands) used for data ingestion.
- A YAML description file for the collection ingestion

Rasdaman

Rasdaman software (Baumann et al. 1998) allows the publishing of geospatial multi-dimensional data using the OGC WCPS. For integrating geospatial data, an import script is available that makes use of a "recipe" configuration file in JSON text format (Listing 4.8). A common way to ingest data is the use of the import script "WCSTImport," for which each geospatial file represents a date. In the recipe JSON text file, the relation between files and dates is given either by metadata tags (Lines 17-19) or by regular expressions based on the filenames. In addition, further processing information (e.g., about tiling and projection) can be configured within the recipe.

```
01 {
02    "config": {
03      "service_url": "http://localhost:8080/rasdaman/ows",
04      "tmp_directory": "/tmp/",
05      "default_crs": "http://localhost:8080/def/def/crs/OGC/0/Index2D",
06      "automated": false
07    },
08    "input": {
09      "coverage_id": "MyCoverage",
10      "paths": [
11        "/var/data/*.tif"
12      ]
13    },
14    "recipe": {
15      "name": "my_custom_recipe",
16      "options": {
17        "time_format": "auto",
18        "time_crs": "http://localhost:8080/def/crs/OGC/0/AnsiDate",
19        "time_tag": "MY_SPECIAL_TIME_TAG",
20      }
21    }
22 }
```

Listing 4.8: Example configuration file for ingestion into Rasdaman.

The following requirements for data structures for ingesting raster time-series data into Rasdaman software can be derived:

- A single geospatial data file for each date.
- A recipe JSON file, which is used for data ingestion.

4.3.3 Conclusions

The programming languages and geospatial tools investigated here mostly work with single-band rather than multi-band geospatial files for spatial time-series data. The use of separate information regarding the date of each geospatial file applies to all of them—this is either provided as a regular expression extracted from the filename or as list of dates in the same order as the list of files. For some of the tools, either further processing or ingestion steps are necessary for time-series data. This can either be provided to the users as tutorial or directly adjusted in the analysis tools. Thus, a common data structure and format for all EO time-series data, to which the user is accustomed, needs to be considered when designing and setting up data platforms. Automated conversion tools for converting from EO data formats into this common data format are necessary to simplify the work with that data.

4.4 Cloud-based EO Time-series Data Platforms

In these times of large EO data archives, it seems obvious to bring geoprocessing and analysis tools to the data instead of downloading the original EO data. Many cloud-based infrastructures and platforms provide different kinds of EO data and are available with different types of services (Table 4.4). In general, one can distinguish between cloud infrastructures (e.g., Amazon, Google, and Copernicus DIAS) and processing and service platforms (e.g., Google Earth Engine, Sentinel-Hub, and Open Data Cube). Cloud infrastructures mainly provide virtual environments (Subsection 4.4.1), which were initially focused on providing virtual operating systems, but today have improved to incorporate serverless infrastructures and the execution of containerized application without the need to setup and install virtual machines (see Subsection 3.1.5). On top of cloud-based infrastructures or local server environments, processing platforms have been developed to simplify work with EO data and the execution of algorithms provided on web-based platforms (Subsection 4.4.2). Some web-based platforms also include an option to upload or develop algorithms on the platform. In addition, service platforms exist that provide only web services for data discovery, visualization, access, and analysis (Subsection 4.4.3), which can be integrated in self-developed applications.

Table 4.4: EO time-series data infrastructure and web platforms (* Usage costs).

Provider	**EO data**	**Type**
Cloud infrastructures		
Amazon Web Services*	Sentinel–2, Landsat–8 https://aws.amazon.com/de/public-datasets/	Virtual Machines—any software that can run on Linux/Windows. Serverless.
Google Cloud (Compute Engine)*	Sentinel–2, Landsat 1–8 (USGS archive) https://cloud.google.com/storage/docs/public-datasets/	Virtual Machines—any software that can run on Linux/Windows. Serverless.
Copernicus DIAS*	Sentinel, Landsat, others *(dependent on platform)*	Virtual Machines, Web portal, Services
Processing and service platforms		
Google Earth Engine	Sentinel–1, Sentinel–2, Landsat 4–8 (USGS archive), MODIS, etc.	JavaScript-Playground/Python API (functions can be used individually)
Sinergise Sentinel-Hub services*	Sentinel, MODIS, Landsat–8, Landsat 5–7 (ESA)	OGC-compliant web services (WMS, WFS, WCS)
Open Data Cube	Landsat, Sentinel, MODIS, others *(dependent on platform)*	Web portal, Services
NASA Giovanni	MODIS	Web portal
ORNL DAAC	MODIS	Web portal, Services
Rasdaman	*Dependent on platform*	OGC WCS, WCPS, WMS

4.4.1 Virtual environments

Virtual environments offer infrastructure with the option to develop and use algorithms and processing tools. Three currently available virtual environments are described and evaluated in the following paragraphs: Amazon Web Services, Google Cloud Platform, and Copernicus DIAS.

Amazon Web Services and Google Cloud Platform

Both Amazon Web Services and Google Cloud Platform provide solutions for scalable infrastructure combined with access to large amounts of EO data. Fees are generally calculated depending on the usage, hardware, operating system, and networking features chosen by the user. Access to virtual machines is provided by command line, programming libraries, or service interfaces. As described in Subsection 3.1.5, modern technologies are provided by Amazon and Google (e.g., serverless web services). As all data is provided unzipped, it can be used directly within self-developed applications. In contrast to the various access services from data providers, the data can be directly processed or analyzed.

Amazon Web Services provides on-demand computing platforms based on the infrastructure of Amazon. USGS Landsat–8 data can be accessed via HTTP requests from external applications. Other EO data is only available within the infrastructure or by using libraries to access data externally. Several open datasets are publicly available,[27] which can be used without storage costs, such as:

- USGS Landsat–8
- ESA Sentinel–1 Ground Range Detected (one-year rolling archive)
- ESA Sentinel–2
- NASA MODIS (some products)

The Google Cloud Platform provides a suite of cloud computing services that runs on the Google infrastructure. All data is available via HTTP requests from external applications and can further be accessed using libraries. Several open datasets are publicly available,[28] and can be used without storage costs, such as:

- USGS Landsat 1–8 (complete USGS archive)
- ESA Sentinel–2

[27] https://aws.amazon.com/de/public-datasets/
[28] https://cloud.google.com/storage/docs/public-datasets/

Copernicus Data and Information Access Services

The Copernicus DIAS program aims to provide discovery, access, and processing capabilities for all Copernicus data and information. To this end, the European Commission set up five platforms from different European consortiums in 2018 (European Commission 2018b): Mundi Web Services, Sobloo, CREODIAS, ONDA-DIAS, and WekEO. The European Association of Remote Sensing Companies has published a first comparison of the DIAS platforms.[29] Differences between the platforms are based on data availability and the availability of processing tools (e.g., Jupyter Notebooks or virtual desktops). All the platforms provide free discovery and access services, which allow for searching and downloading of Copernicus data. In addition, virtual machines and other services are provided on a pay-per-use basis, with direct access to EO data archives. The data available on the different platforms differs: At least Sentinel data is available either directly or on demand. Many DIAS platforms also provide access to Landsat data and data from Copernicus Services. As these platforms are still new (all of them published in 2018), additional features can be expected in the future.

4.4.2 Processing platforms

Processing platforms offer web-based applications with the option to develop and launch algorithms and processing tools. Selected processing platforms for vegetation monitoring are described and evaluated in the following paragraphs: NASA Giovanni, Oak Ridge National Laboratory Distributed Active Archive Center (ORNL DAAC) MODIS Tools, Google Earth Engine, and the Open Data Cube interfaces.

NASA Giovanni

The NASA Giovanni—Interactive Visualization and Analysis system offers a wide range of MODIS Level–3 products as well as various time-series extraction and analysis tools (Acker & Leptoukh 2007). Although it provides access to all input data for a given bounding box and time range, these functions are not available as web services, which would enable their integration into self-developed applications. As such, only the web-based system provided by NASA Giovanni can be interactively used by users. After data integration and data analysis has been conducted, input data and intermediate processing data can be accessed. Results are shown in interactive charts and can be downloaded in various data formats. In addition, all inputs and outputs from the processing steps can be downloaded and reproduced. Interactive charts, netCDF, and CSV formats are available as output formats (Figure 4.2).

[29] http://earsc.org/news/dias-comparison

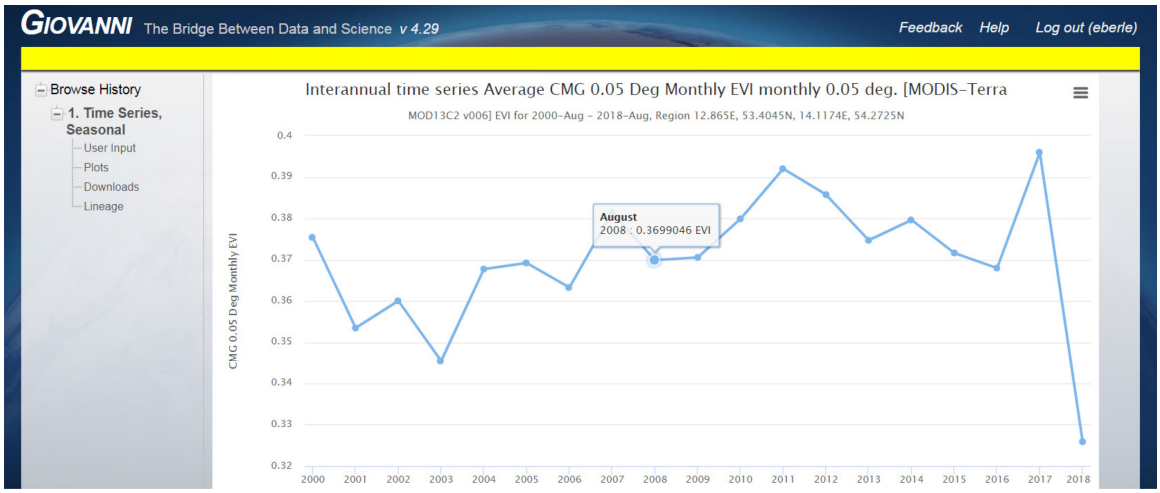


Figure 4.2: Results of the web-based NASA Giovanni tool analysis of MODIS time-series data.

ORNL DAAC MODIS Land Subsets

The MODIS Land Subsets[30] tool from the ORNL DAAC in the United States provides time-series extraction for a bounding box around a pixel location. In addition to time-series data access, land-cover types and a phenology time-series plot for vegetation data are also available. Both, SOAP and REST-based web services are available for querying the data catalogue, extracting time-series, and submitting an order for data access and analysis graphs. Although the tool can be accessed as a web service, further analysis can only be conducted after an email with the resulting data has been received by the user. Results are available on the web portal (Figure 4.3). As status responses from the ordered process are not available, a solution to monitor the status of processing needs to be developed.

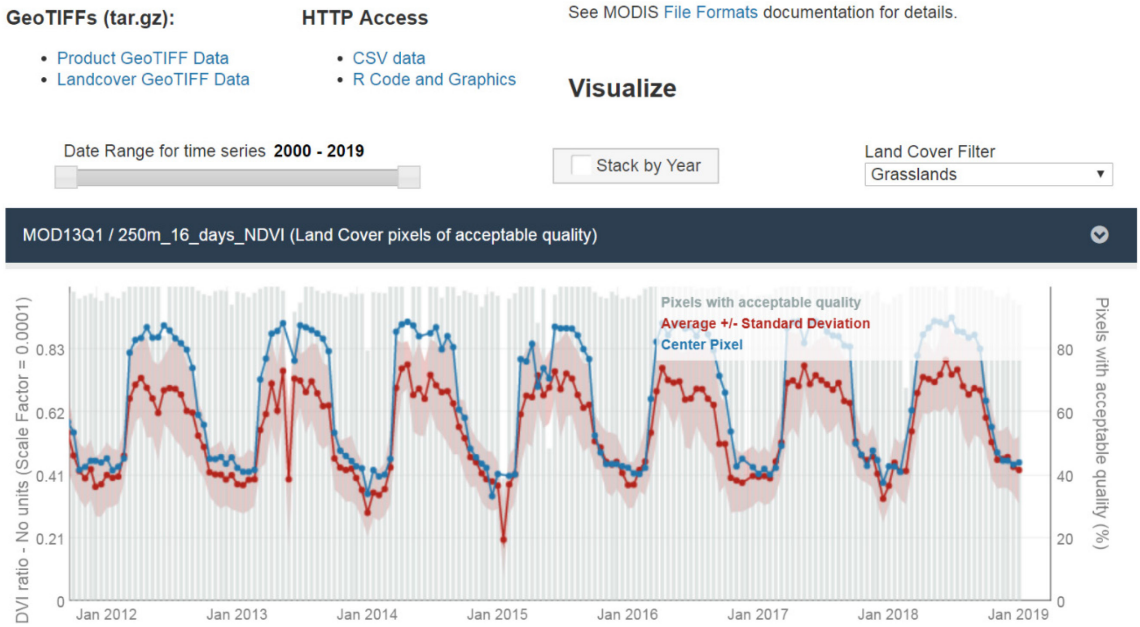


Figure 4.3: Results of the web-based MODIS Global Subsets tool from ORNL DAAC.

[30] https://modis.ornl.gov/data/modis_webservice.html

Google Earth Engine

Google provides a web-based Google Earth Engine Playground for interactively discovering and analyzing EO data in a browser (Gorelick et al. 2017). The user interface (Figure 4.4) consists of a code editor component, an interactive map showing the resulting data and features, a console output, a script explorer, documentation, and an assets window. Any code used in the editor can also be used by the Python API and integrated in self-developed applications. Several geospatial tools have been provided by Google, such as (un)supervised classifications, masking, mathematical operations, edge detection, spectral transformations, operators to reduce dimensions, object-based methods, array operations, and the like. Users can upload their own geospatial data and use it in conjunction with the other datasets available. Any resulting data can be downloaded either using HTTP requests or exported to Google Drive. Earth Engine allows the sharing of applications with the public. Commercial use of Earth Engine needs to be approved by Google.

Web-based applications can be published based on JavaScript source code integrated into the Playground editor. With layouts, panels, and widgets, user interface components can be integrated into applications. Without web development technologies, such as HTML and Cascading Style Sheets, applications can be developed and provided based on the Google Cloud Platform. The application can make use of the data, tools, and processing capabilities available in Earth Engine. Figure 4.5 shows an application in which users can select a point for the extraction of Sentinel–1 time-series data.

Open Data Cube interfaces

The user interface of the CEOS Open Data Cube visualizes the available EO data and provides access to the analysis tools registered in the backend and their resulting data after execution (Figure 4.6). The backend stores areas of interest and analysis tools linked to these areas. Individual raster data and raster time-series data can be ingested into the Open Data Cube. A fixed set of analysis tools is provided with the default installation. Various output formats, such as GeoTIFF, netCDF, and PNG, for the results of the analysis are available in the user interface. The interface makes use of a variety of technologies, such as Celery for task scheduling and parallel executions. Extensions are available to provide OGC-compliant visualization (WMS), download (WCS), and processing (WPS) services. A Jupyter Notebook server can be connected to the Open Data Cube on the same infrastructure. This allows the running of own analyses using Python in the browser. Own analysis tools need to be implemented as Django applications in consideration of the technologies used in the Open Data Cube user interface (e.g., Python Celery). The user interface provides RESTful web services.

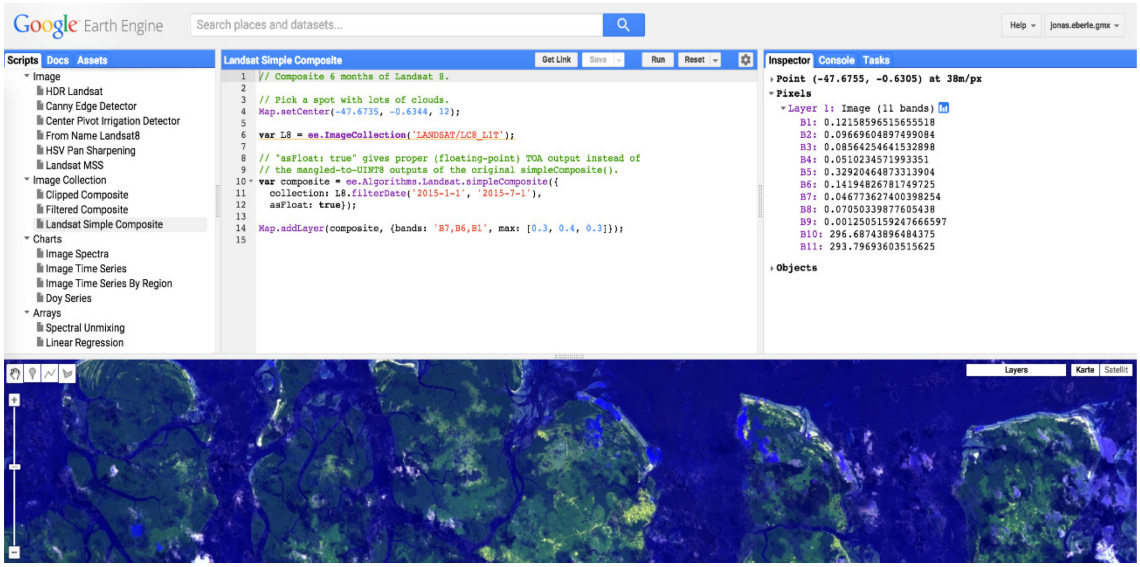


Figure 4.4: Google Earth Engine Playground web application: JavaScript and Earth Engine functions are used to develop analysis algorithms, with the results presented in the map.

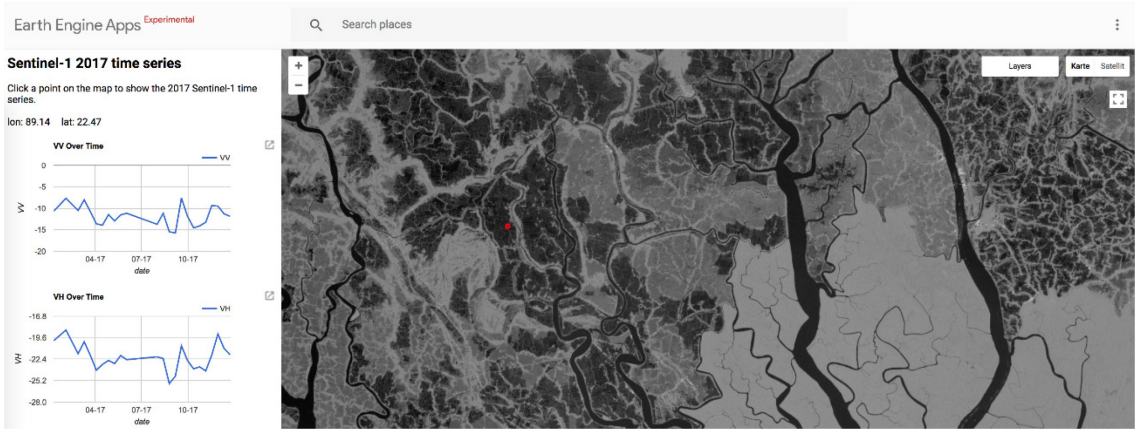


Figure 4.5: Screenshot of a Google Earth Engine App (Clauss 2018).

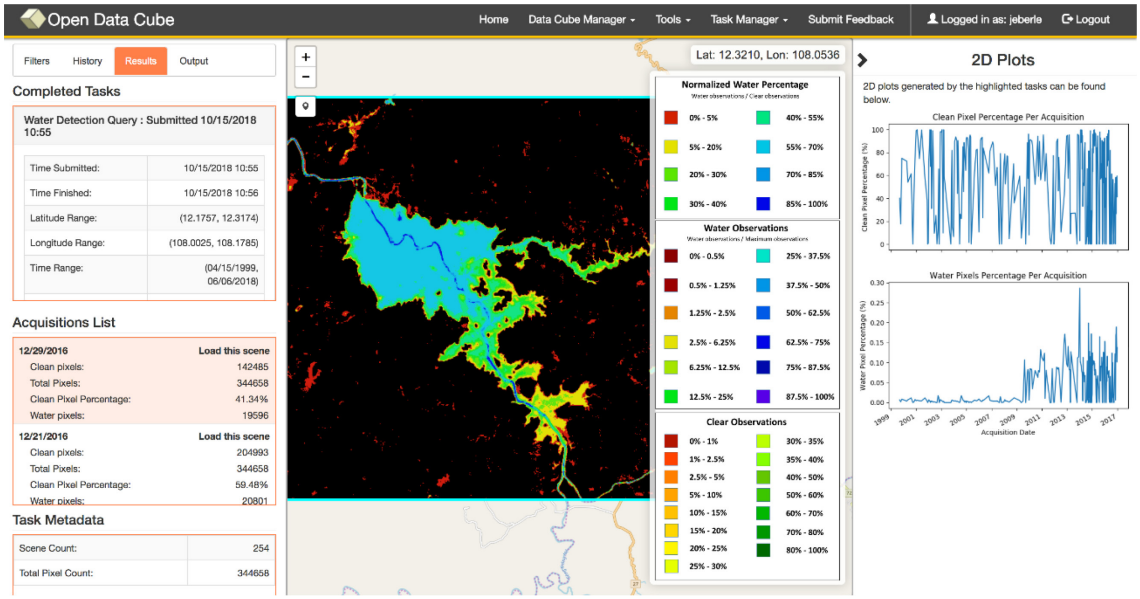


Figure 4.6: User interface of the CEOS Open Data Cube showing results of the water-detection algorithm in the map and as time-series plots on the right.

4.4.3 Service platforms

Service platforms only provide web services, which can be used in other applications. Focusing on standard-compliant web services (e.g., according to the OGC specifications), the services can be used directly in many geospatial applications. The commercial web services from Sentinel-Hub[31] provided by Sinergise are an example of service platforms. With OGC WMS, WFS, and WCS, they provide interoperable web services for EO data. In addition, nonstandard-compliant web services are also provided to the user, such as the Sentinel-Hub Feature Information Service,[32] which provides statistical summaries for a given area of interest over time for EO time-series data. Besides data discovery and data access, data can be processed with pixel-based scripting possibilities[33] when requesting the data.

A self-developed service can be created using Rasdaman software[34]. Rasdaman provides web services compliant with the OGC WCS, WCPS, and WMS. EO data ingested into Rasdaman can be analyzed using the WCPS specification for individual service requests. Output formats of the WCPS includes GeoTIFF, PNG, and CSV.

4.4.4 Conclusions

While the potential of cloud data providers and their EO data archives in combination with the execution of geoprocessing tools are immense, various limitations exist, such as usage costs, limited functions, or data availability dependent on specific providers. Popular cloud providers, such as Amazon or Google, offer a variety of tools to simplify the setup of processing in the cloud (e.g., serverless infrastructure). Copernicus DIAS platforms provide OGC-compliant data discovery, visualization, and access services in addition to virtual processing environments, which are similar to those of Amazon and Google. The use of cloud providers within self-developed platforms can have a major impact but must be considered meaningful. Due to the distributed behavior of service-based infrastructures, cloud providers can be used in various ways, for example, only for the provision of web services for specific tasks or specific EO data.

Processing platforms are diverse in their technological implementation. While with Google Earth Engine, only specific operations and functions can be used (e.g., own analysis tools cannot be uploaded), other platforms can be extended by own tools or used directly within programming languages (e.g., Jupyter Notebooks). The Open Data Cube provides simple access to analysis tools, but for the implementation of own algorithms in the user interface,

[31] https://www.sentinel-hub.com/develop/documentation/api/ogc_api
[32] https://www.sentinel-hub.com/develop/documentation/api/fis-request
[33] https://sentinel-hub.com/develop/documentation/api/custom-evaluation-script
[34] http://rasdaman.org/

several technologies (e.g., Django, Celery) need to be known. Although there are platforms that provide many of the platform-specific user requirements, there are often important requirements that are still missing (e.g., the provision of own analysis tools, own user management, and specific visualization outputs for the results of analysis). Service platforms are still new but offer enormous opportunities for the direct processing and visualization of EO data while conducting the service request. Although the analysis of EO data can be integrated upon request, time-series analyses are not possible; only single pixels of a single scene or a temporal mosaicked scene can be classified.

Table 4.6 presents a comparison of the user requirements and the various processing and service platforms for EO data access and analysis described in this section. While EO data access and visualization tools are available on most of the platforms, there are differences pertaining data analysis, user management, and service interfaces. Especially information about data lineage in relation to reproducible research (e.g., how the data has been processed), is only provided in detail by the NASA Giovanni system. In addition, most of the systems cannot be hosted on own platforms with specific EO data and analysis tools. Thus, a self-developed standardized and flexible middleware system operating between users and data providers fosters the exploration of EO time-series data.

Table 4.5: Comparison of user requirements between different web services providers for discovery and access of EO data.

a) Data discovery services

	ESA/Copernicus Open Access Hub	USGS Earth Explorer	NASA CMR	Google Earth Engine	Sentinel-Hub	GEOSS Broker	CEOS WGISS	ESA FedEO
Service architecture	REST	REST	REST	N/A	REST	REST	REST	REST
Asynchronous	☒	☒	☒	N/A	☒	☑	☒	☒
Standardized services	OpenSearch	☒	☒	☒	OGC WFS	OGC CSW, OpenSearch	OGC CSW, OpenSearch	OGC CSW, OpenSearch
OGC specification	☒	☒	☒	☒	☑	☑	☑	☑
Output formats	XML, JSON	JSON	XML, JSON, CSV	N/A	XML, GeoJSON	XML, JSON	XML	XML, GeoJSON
Authentication needed	Yes	Yes	No	Yes	Yes	No	No	*

b) Data access services

	ESA/Copernicus Open Access Hub	USGS Earth Explorer	NASA CMR	Google Earth Engine	Sentinel-Hub	USGS ESPA	Amazon Web Services	Google Cloud
Service architecture	REST	REST	REST	N/A	REST	REST	REST	REST
Asynchronous	☒	☒	☒	☑	☒	☑	☒	☒
Standardized services	Open Data Protocol	☒	☒	☒	OGC WCS	☒	HTTP URL	HTTP URL
OGC specification	☒	☒	☒	☒	☑	☒	☒	☒
Output formats	XML, JSON	JSON	XML, Atom, JSON	N/A	XML	JSON	N/A	N/A
Data formats	ZIP	GeoTIFF, ZIP	*	Python object, GeoTIFF	GeoTIFF, PNG, JPEG	GeoTIFF, HDF-EOS, ENVI	GeoTIFF	GeoTIFF
Authentication needed	Yes	Yes	*	Yes	Yes	Yes	No	No
Summarized results	☒	☒	☒	☒	☒	☒	☒	☒

* dependent on data provider

Table 4.6: Comparison of user requirements between different processing and service platforms for access and analysis of EO data.

	Google Earth Engine	Sentinel-Hub	NASA Giovanni	ORNL DAAC	Open Data Cube	Rasdaman
Multi-source EO and geospatial data*	Landsat, MODIS, Sentinel	Landsat, MODIS, Sentinel	MODIS	MODIS	Landsat, MODIS, Sentinel**	Landsat, MODIS, Sentinel**
EO data visualization	☑	☑	☒	☑	☒	☑
Time-series data download	Export to Google Drive or download links	Individual requests to web service	Individual data requests	Zipped archive file after order submission	☒	Summarized statistics, individual requests
Analysis tools	Various programming functions	Various programming functions	Fixed set of tools	Phenology, Land cover	Fixed set of tools**	Through WCPS query
Analysis output formats	Interactive maps and charts, GeoTIFF	Through OGC WMS, WCS web service	Interactive charts, netCDF, CSV	GeoTIFF, Graphics, R, CSV, Interactive charts	Default**: PNG, GeoTIFF, netCDF	GeoTIFF, CSV, PNG
Data analysis with various programming languages	☒	☒	☒	☒	Python through connected Jupyter Notebook	☒
EO data discovery	☑	☑	☒	☒	☒	☒
Own user management	☑	☒	☒	☒	☑	☒
Self-hosting	☒	☒	☒	☒	☑	☑
Interfaces	Python API, UI	REST, UI	UI	REST, SOAP, UI	Python, REST, UI	REST
OGC web services	TMS	WMS, WCS, WFS	☒	☒	WMS, WCS**	WCPS
Lineage information	☒	☒	☑	☒	☒	☒

* Only Landsat, Sentinel, and MODIS are considered here

** Dependent on each instance

UI = User interface

4.5 Recommendations

A key issue in discovering and accessing EO time-series data is the need to learn how to use the different tools and services offered by each data provider. Focusing on the needs of a user, a unique standardized and easy-to-use interface will foster the use of EO time-series data. Solutions, such as brokering services, have been established to bring EO data from different data providers together and to provide a standardized and harmonized interface. However, often the results of a discovery request are still too complex. Data providers often provide different web services for the discovery and accessing of EO time-series data. In addition, data from the same EO mission is often available from several data providers. As such, data discovery and access tools from multiple services can be used.

The further processing and analysis of EO time-series data requires specific data structures and data formats, which are often different for individual geospatial tools. However, some common specifications can be derived from the review: Individual geospatial files, which can be combined into a multi-band time-series dataset for each observation, are necessary for each date and observation (e.g., bands and indices). Dates need to be either extracted from the file name or included in a separate text file.

Various cloud-based virtual environments, processing platforms, and service platforms exist, with different infrastructures, EO data, and analysis tools available. Which infrastructure or platform is to be used is often a difficult decision but this depends on the EO data and functions that need to be provided. In addition, open source software exists that can be installed and hosted on own platforms with own EO data and analysis tools.

The following recommendations based on the previous review can be made:

- Provide multiple data formats to meet the requirements of different users.
- Establish service brokering to harmonize data discovery and access.
- Standardize data structures and formats for multi-source EO data.
- Provide a combined implementation of a self-developed data platform integrating web services hosted by cloud-based infrastructures (e.g., services from virtual environments or serverless web services and service platforms).
- Offer either the uploading of own analysis tools or a direct linking of data access services with analysis services within a processing platform.

Chapter 5: Concepts and Methods

Based on the previous chapters, concepts and methods have been defined and grouped into three components to bridge the gap between EO data archives and user applications (Figure 5.1):

1) Service-based EO time-series data middleware as an overall concept for an EO data platform focusing on general methods for user-aligned web services and user-aligned output formats,

2) Service brokering for multi-source EO time-series data discovery and access, specifically focusing on user-aligned approaches for discovery and access, and

3) Uniform EO time-series data structure, processing, and analysis, specifically focusing on the further provision, analysis, and processing of time-series data after the data has been downloaded.

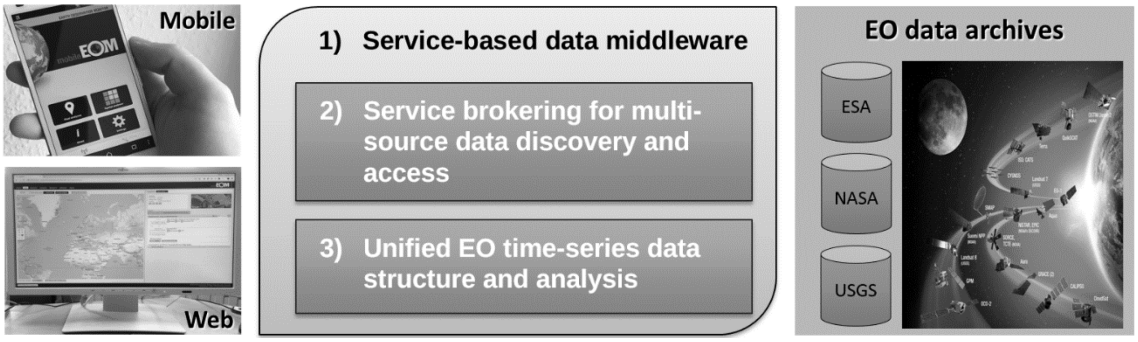


Figure 5.1: Main components of this thesis for bridging the gap between EO data archives and user applications (Image on the right: Courtesy NASA/JPL-Caltech).

5.1 Service-based EO Time-series Data Middleware

The current state of web technologies offers tremendous potential to increase the accessibility of EO time-series data and analysis tools. Today, global EO data archives can be analyzed in virtual environments and processing platforms without data downloads in scalable infrastructures on Amazon Web Services, Google Cloud, or Google Earth Engine. However, not all EO data is available on clouds and, thus, traditional methods for data discovery and download still need to be undertaken manually. As these methods have in most cases been made available through web services, a centralized server infrastructure can provide automated access to multi-source EO time-series data, which enables users to access and analyze them.

In this thesis, a service-based infrastructure and data platform have been developed with a focus on a user-driven design. As most of freely available EO data is available through web services, platforms with user-aligned web services can easily be developed to simplify the discovery, access, and analysis of EO time-series data. Desktop and mobile applications can benefit from these user-aligned web services as the complexity of geospatial time-series data processing is hidden.

In general, such a data platform needs to fulfill the following criteria to enable the provision of user-aligned web services and user-aligned output formats:

- **Automated access to user-requested data linked to scientific analysis tools:** Both data access and analysis need to be reproducible and to focus on user-specific requirements.
- **Uniform data processing:** This includes data formats, processing interfaces, and metadata descriptions for EO-based time-series data as key components to provide automated and on-demand tools for data access and analysis. Various output formats for each geospatial tool need to be considered.
- **Interoperable web services for spatial time-series data, which can be used within different applications:** This includes web services for the visualization of data and the results of analysis, discovery of available datasets, time-series data access and extraction, and the execution of time-series analysis tools.

The overall concept of the regional data middleware system is described in the next subsection (5.1.1), including the system and software architecture of the middleware. The research towards user-aligned web services (Subsection 5.1.2) and user-aligned output formats (Subsection 5.1.3) are explained thereafter. Finally, the software and services implemented are presented (Subsection 5.1.4).

5.1.1 Concept of a regional data middleware system

The overall concept of the regional data middleware system focuses on providing easy and standardized access to EO time-series data and analysis tools. It combines the advantages of web service-based geoprocessing tools and user-aligned interfaces. Many users spend a tremendous amount of time on data discovery, download, and harmonization (Zhao et al. 2012). The middleware approach bridges existing limitations between users and data providers with automated workflows for EO time-series data.

The data middleware system allows users to focus on their main interests:

- Software developers focus on the development of applications based on the web services of the middleware.
- Scientific users focus on the development and use of algorithms without the need to handle data preprocessing issues.
- Thematic experts focus on the execution of analysis tools and the interpretation of the results of analysis in their area of interest.

The middleware approach supports the exploration of EO time-series data for regional areas of interest. Although access to global EO data is available, the system is not aimed at conducting global analyses, as data downloads of global datasets are not feasible. The middleware aims to connect to major EO data providers, such as NASA, USGS, ESA, Google, and Amazon. The main functions include time-series data discovery and access for user-specific areas of interest, data analysis, and the provision of standard-compliant web services. The results of analysis are automatically prepared for online visualization. Different user personas can interact with the middleware: Scientists to retrieve data and execute algorithms; software developers to use the services in their applications; and thematic experts to explore the algorithms and their results.

5.1.1.1 Middleware components

The methodological concept is shown in Figure 5.2. A fundamental requirement for the components of the middleware is direct access to global EO data. Thus, data discovery and data integration need to be connected to external data providers. A centralized management of time-series data (“analysis-ready data”) is necessary to combine data access with additional processing and analysis steps (“data processing and analysis”). Time-series data can be made available for external applications, such as Open Data Cube or Rasdaman (“application-ready data”). The provision of the middleware with standardized web services and metadata descriptions fosters the development of client applications. All these components (Figure 5.2) are described in the following paragraphs.

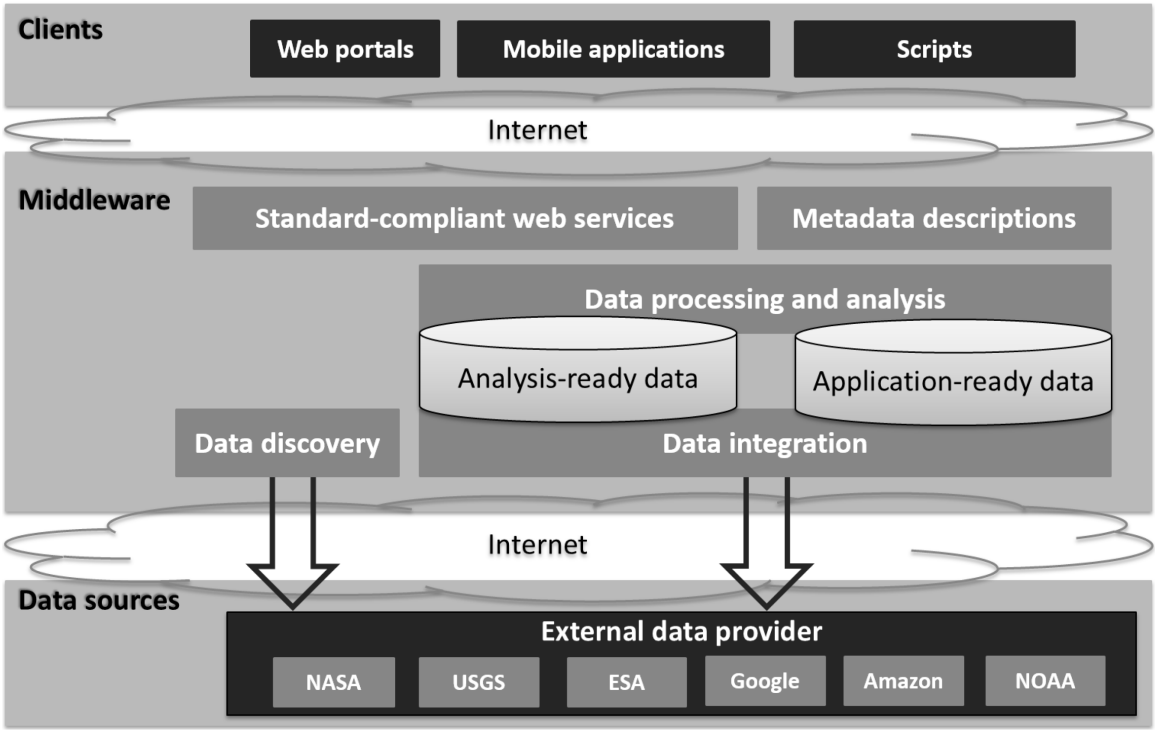


Figure 5.2: Concept of the methodological development, including the middleware approach and user-friendly client applications (e.g., web portals, mobile apps, scripts).

Data discovery

Questions regarding the availability of specific EO data (e.g., Landsat, Sentinel, and MODIS) can be answered by the data discovery component within the middleware. Users need to be able to search for datasets in their own area of interest by filtering through sensors, cloud coverage, and other relevant parameters. A multi-source data discovery approach involves different data providers. Thus, a brokering system harmonizes requests to and responses from metadata catalogues and is connected to the web services of the individual data providers. Specific user-aligned output formats need to be considered to ensure use by different user personas. In many cases, data discovery is necessary to conduct data access and data integration.

Data integration (data access)

Access to EO time-series data can be established by connections to external data providers. As many of the data providers offer different types of web services for data access, requests to the individual data providers need to be harmonized for each of the services. Both data extraction and data integration need to be investigated:

1) Data extraction involves time-series data for a single pixel or statistical summary of an area in order to provide a quick overview of the time-series values.

2) Data integration into the middleware includes data preprocessing steps and data format conversions in order to enable further analysis.

Analysis- and application-ready data

The term “analysis-ready data” in this thesis describes the ability of handling time-series data with a uniform data structure and format, which allows it to be easily used in programming languages and by geoprocessing and analysis tools. Therefore, a uniform data structure is necessary, and the data downloaded needs to be converted into this structure and format. To foster the use of time-series data by external analysis tools, it is also relevant to support data structures and formats used in other software, such as GRASS GIS, Open Data Cube, Rasdaman, and OGC web services. In this thesis, methods have been defined to automatically convert EO time-series data into the data format required by other applications (“application-ready data”). With this accomplished, users can directly use EO time-series data from the middleware in applications that are supported by the middleware.

Data processing and analysis

The combination of data access and data processing allows users to analyze time-series data using specific time-series analysis tools. A flexible approach has been considered in order to enable user-specific data processing and analysis, which allows users to conduct their own processing of the data. This includes the use of interactive data exploration and analysis tools, such as the web-based Jupyter Notebooks, the algorithms in the Open Data Cube, or others registered as applications in the middleware.

Standard-compliant web services

The methodological concept includes the provision of standard-compliant web services to allow users and applications to interact with the middleware. Web services allow the integration of the middleware into different types of applications, such as web and mobile applications and programming scripts. In addition, these web services can be compiled into processing chains, for example, to combine data access and data analysis tools. In order to provide uniform and user-aligned web services, the use of standard-compliant web services has been investigated.

Metadata descriptions

Descriptions of geospatial data are important to provide information about the kind of data that is available in the middleware. Client applications can use a metadata catalogue, which contains the metadata descriptions, provided by the middleware to search for available data. In addition, further information about the geospatial data can be explored, such as information about the data processing conducted.

5.1.1.2 System architecture

The architecture of the middleware system comprises three main components accompanied with their respective web services (Figure 5.3):

1. Administration with RESTful services,
2. Geoprocessing tools with geospatial web services, and
3. Exploration tools with application services.

Based on a service-oriented infrastructure, the individual services can be provided by different servers. Only those services that need to access the EO data within the geospatial database need to be managed centrally. Figure 5.3 shows the architecture of the system based on the administration services, geospatial web services for data discovery, integration and analysis, geospatial database, and application services for the external exploration tools. The three components and the geospatial database are described in the following paragraphs.

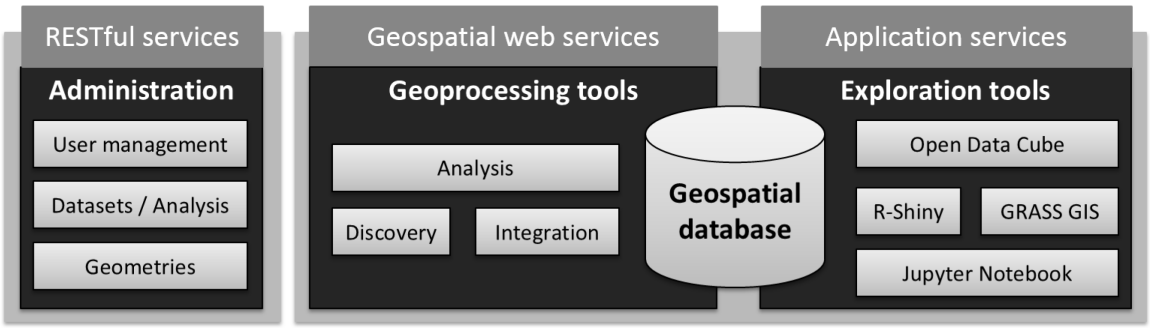


Figure 5.3: System architecture of the regional data middleware system divided into administration, geoprocessing tools, and external exploration tools.

Administration

The administration component manages the registration and authentication of user accounts, the areas of interest registered in the middleware, the data integrated, and analysis tools used. The provision of these functions by means of RESTful services allows access to these data using different client applications. The web services are part of a web content management system (CMS), which can be accessed through an additional administration web interface. The CMS provides user management tools, such as user registration, user removal, password resetting, login, and logout. In addition to content management, the execution of processes can be controlled with process scheduling to limit overloading of the server (e.g., only a single data integration process at the same time). As the main interaction of client applications with the middleware is based on RESTful services, the software behind the CMS can be changed.

Geospatial database
The geospatial database consists of EO time-series data and the results of analysis that have been generated through the geoprocessing tools. A fixed folder structure and filename scheme allows centralized data management and links to external exploration tools. Standardized OGC-compliant web services allow simple use and integration into GIS software (e.g., QGIS and ArcGIS) and web-mapping libraries (e.g., OpenLayers).

Geoprocessing tools
The geoprocessing tools manage the processing for data discovery, access, integration, and analysis. Scripting languages, such as Python and R, are widely used for geospatial data processing. For both languages, many packages exist to handle and process all kinds of EO time-series data, including tools for data discovery, access, processing, and analysis. Although both languages can be considered for the geospatial data discovery and processing component, there are some external libraries, such as the Google Earth Engine API or the Open Data Cube API, which only exist for Python. In addition, support for command-line tools is necessary for some processing and analysis tools. Communication with client applications is based on standard-compliant geospatial web services, such as OGC WMS (visualization), WCS and WFS (access), WPS (processing), and CSW (discovery).

Exploration tools
Web-based exploration tools allow users to work with data without having to download and preprocess it and install software. As it is connected to the middleware system, data in the geospatial database can be directly accessed by these applications. As a prerequisite, the exploration tools need to have access to the geospatial database. Depending on the specific tool, EO time-series data needs to be converted into the format that the application requires (e.g., the specific data format for the Open Data Cube software). For example, Jupyter Notebooks can be used to further process EO time-series data by providing a web-based interactive development environment. Thus, after data integration, users are able to open the dataset within a Jupyter Notebook application. Other applications (e.g., Open Data Cube) can be linked to data ingested in the middleware to allow users to work with algorithms that are available in the application. As a centralized CMS, the middleware can manage links between the data ingested and exploration tools available.

5.1.1.3 Spatial data infrastructure
The middleware concept is based on methods of SDIs. A relevant part of an SDI is the provision of interoperable services. As such, standard-compliant web services for data visualization, access, processing, and metadata cataloguing following the specifications

of the OGC are available. Figure 5.4 shows services from the SDI, which can be used by several client applications connected through the Internet. In this thesis, the emphasis is on the automated processing of EO time-series data. Processing tasks, such as data downloading, processing, and analysis are provided using the OGC WPS specification. Within the process algorithm (shown in Figure 5.4), external software can be made available, for example, by executing software on the command line or through libraries used in programming languages. This allows any kind of software to be integrated into a service environment and provided as a web service.

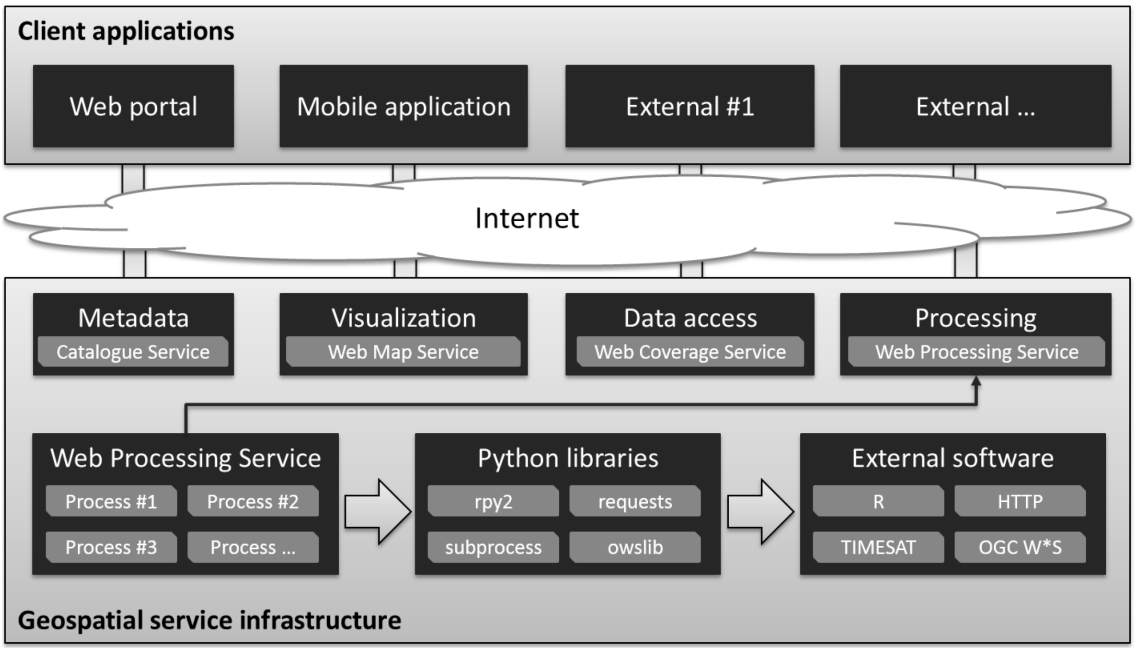


Figure 5.4: Connections of client applications to the geospatial service infrastructure, including processing based on OGC WPS in Python.

5.1.2 User-aligned web services

The use of web services facilitates the exploration of data based on web technologies. Data and processing tools can be made available through web services to allow the decentralized use of these tools in multiple applications, such as mobile applications, scripting programs, and web portals. Hosted on a web server, services can be managed in a central environment and used in various applications. In general, the use of web services can be adjusted by means of the different input parameters defined by each web service. A service-based infrastructure ensures the integration of EO time-series data and processing tools in any kind of application. Linking EO data with web services provides a range of opportunities for user-specific exploration tools. Thus, the approach of providing EO time-series data with standard-compliant web services needs to align with the specific needs of users. Web services allow the concealing of complex data processing behind web-based resources. Simple requests to a web-based resource can initiate complex

processing on a server—the user retrieves only the resulting data. However, the resulting data must be adapted to the respective user personas. Scientific users need to understand the processing within a web service, while thematic experts are interested mainly in the results. Therefore, multiple output formats, logging, documentation, and reproducible services are key issues when providing an infrastructure based on web services. In addition, several services can be linked by service chaining opportunities.

Research has been conducted to standardize web services for the provision of geospatial data discovery and access (Bai & Di 2011; Baumann et al. 2016b). However, more focus needs to be placed on user-aligned web-based interfaces for the exploration of big data (Tsinaraki & Schade 2016). Although standardized specifications of web services for data discovery, access, and analysis exist, the focus has been mainly on “machine-to-machine” communication. With regard to the specific needs of users, this “machine-to-machine” communication needs to be transferred to a more “human–machine” communication interface.

5.1.2.1 The human–machine interface

As concluded in the state-of-the-art chapter, existing web services for EO time-series data discovery, access, and analysis need to be enhanced to meet the requirements of different user personas. Although response formats from commonly used web service specifications (e.g., XML and JSON) can be used for individual user personas (e.g., developers), others prefer additionally processed output formats. Within the concept of a “human–machine interface” specification, data discovery, access, and analysis need to be available with uniform and standardized web service specifications. Output formats for the resulting data and technical and functional aspects of web services—both suitable for the user personas relevant to this thesis—must be reconsidered to enable user-aligned services and applications. Thus, two main criteria are investigated:

1) The provision of multiple output formats to foster the use of services by different user personas.

2) A uniform web service specification for data discovery, access, and analysis to simplify the learning curve when using these services.

Figure 5.5 shows the concept of both the standard solution, with individual requests and arbitrary output formats (left), and the uniform web service with multiple user-aligned output formats (right). To fulfill both these criteria, the service specification for data discovery, access, and analysis needs to be harmonized and the output formats need to be adapted to the requirements of users.

Therefore, traditional service specifications, which only provide a fixed set of output formats, such as OGC CSW, OpenSearch, and OGC WCS, need to be replaced by a more generic specification. In this thesis, the OGC WPS specification has been evaluated for the uniform web service specification as it allows for multiple input and output parameters, diverse processing within the execution, and further requirements set by the user (e.g., support for long-running processes). The WPS specification allows a differentiation between services (e.g., individual processes for data discovery and access), but the execution and handling of the services remains the same. For each of the processes, a flexible set of input parameters and output formats can be defined to meet the individual requirements of the different user personas. In addition, WPS-compliant web services can be used by existing geospatial software that is compliant with OGC standards.

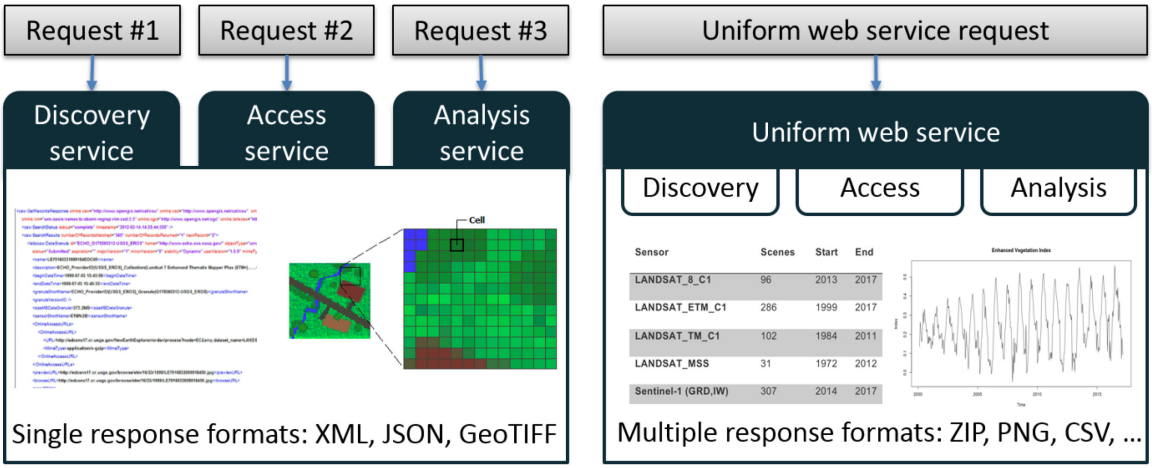


Figure 5.5: Traditional discovery, access, and analysis request/response (left) compared to a uniform web service request that provides multiple user-aligned output formats and a uniform web service interface (right).

5.1.2.2 Service chaining

The chaining of services allows the consecutive execution of various steps in the course of the exploration of geospatial data, such as data discovery, integration, and analysis. An example of service chaining is presented in Figure 5.6, based on two services: 1) time-series data access, and 2) time-series data analysis. To enable service chaining in the centralized data middleware without transferring large amounts of data, specific rules for the services are necessary.

The approach considered in this thesis and shown in Figure 5.6 is based on a centralized output directory for each of the services in the middleware, which can also be retrieved by other services on the same server (“local processing directory”). The results of a data access service are stored in this processing directory on the server, which is accessible by a unique identifier. Based on this unique identifier, the analysis service can access the input data on the server (“Input dataset”). The analysis service needs to understand the

data structure and data management produced by the access services to allow direct use of the previously integrated dataset. The local processing directory described by the unique identifier allows for several subdirectories: The “data” directory stores the outputs of the data access service. Further directories are specific to the analysis tools used (e.g., “greenbrown_*”, “bfast_*”, and “timesat_*”). A timestamp for each of the analysis directories allows multiple executions of the same analysis, for example, when using different parameters of the algorithm.

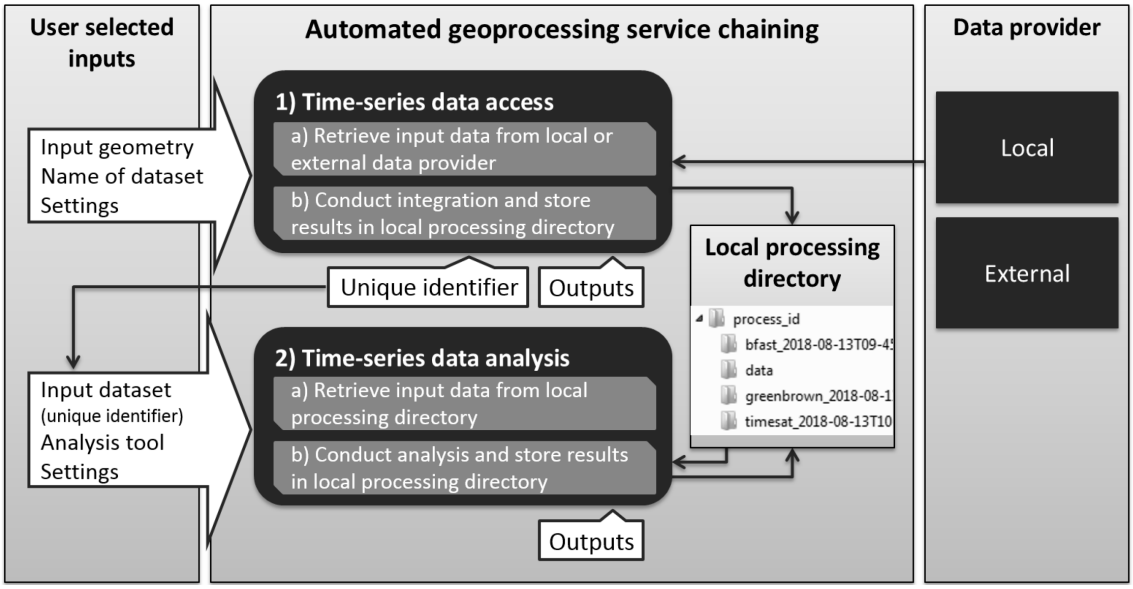


Figure 5.6: Automated geoprocessing service chaining using data access and data analysis services: The results of the data access service are stored in a local processing directory, which is accessed using a unique identifier from the subsequent data analysis service to retrieve the “Input dataset.” The directory listing of the webserver for the folder belonging to the unique identifier includes output results from different services (data access and data analysis).

5.1.3 User-aligned output formats

With focus on the user requirements, one of the aims is to simplify working with and analyzing EO time-series data. Data output formats that are specifically provided to individual user personas can facilitate and support further analysis of time-series data. For example, users do not need to use other software to view the results of the analysis with web-based on-the-fly visualization. In the following subsections, three user-aligned output formats that have been explored in this thesis are described.

5.1.3.1 OGC web services

OGC web services allow the standardized integration of data into GIS and web-mapping libraries. In light of this, it is important to publish geospatial data with OGC-compliant web services. Raster data of spatial time-series data can be made available for visualization using the OGC WMS and for data download using the OGC WCS. In general, software for the provision of raster data often offers services that comply with both specifications. The

temporal dimension based on WMS-TIME and WCS-TIME needs to be considered for raster time-series data. Although the visualization of raster data can be achieved automatically using a black-white color profile, this is not suitable for most use cases. Thus, individual stylings using the OGC Styled Layer Descriptor specification need to be provided and used in the software for each data type (e.g., specific color styles for vegetation index, snow cover, and natural color images). Both the EO data and the results from the analysis tools need to be prepared for visualization, which is relevant when providing analysis tools as web services.

5.1.3.2 Figures and charts

The extraction of EO time-series for single pixels allows the immediate analysis of the pixels over time. While data access in general only provides time-series values, users need to use further tools to visualize the time-series. Providing ready-to-use time-series plots helps users to understand the data without further data processing having to be conducted. Besides a general plot of the data (Figure 5.7, left), further analyses, such as a decomposition analysis plot of a time-series (Figure 5.7, right), can be conducted automatically and provided to the user.

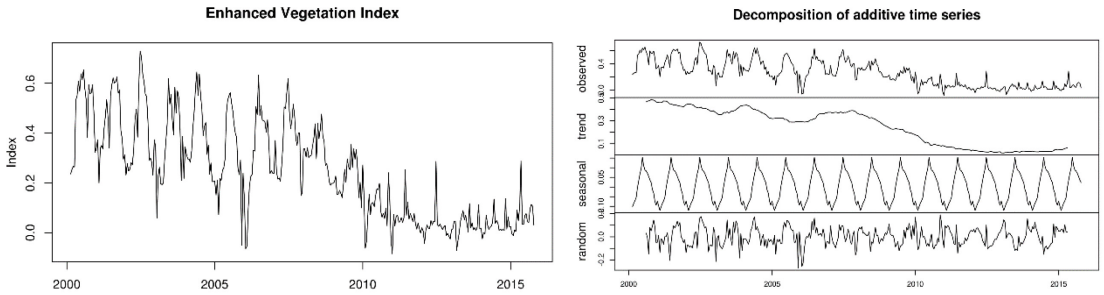


Figure 5.7: Time-series plots provided as outputs while accessing the data (left: general time-series plot; right: decomposition plot).

5.1.3.3 Statistical summaries

Statistical summaries of an area of interest are often used to make first assumptions regarding the indications of the data within this area (e.g., to analyze trends or changes). Without having to analyze each individual pixel within the area of interest, a CSV spreadsheet file can foster this analysis. Figure 5.8 shows an example from an NDVI extraction for an area of interest selected by the user: Statistical summaries are provided as the mean, minimum, maximum, and standard deviation (stdev) values. Using spreadsheet software, users can plot the data (e.g., maximum values versus mean values for each date). In this example, a clear change in the maximum NDVI values is visible in the time-series. This statistical summary is automatically calculated in addition to providing access to raster time-series data.

	A	B	C	D	E
1	date	min	max	mean	stdev
2	18.02.00	0,00	0,96	0,14	0,13
3	05.03.00	-0,18	0,89	0,26	0,18
4	21.03.00	0,06	0,63	0,24	0,11
5	06.04.00	0,07	0,51	0,18	0,09
6	22.04.00	0,08	0,34	0,16	0,05
7	08.05.00	0,15	0,33	0,20	0,04
8	24.05.00	0,21	0,45	0,27	0,05
9	09.06.00	0,23	0,54	0,32	0,06
10	25.06.00	0,20	0,45	0,26	0,05
11	11.07.00	0,13	0,92	0,33	0,08
12	27.07.00	0,24	0,51	0,31	0,04
13	12.08.00	0,17	0,42	0,27	0,04
14	28.08.00	0,19	0,42	0,26	0,04
15	13.09.00	0,15	0,38	0,21	0,04
16	29.09.00	0,18	0,41	0,23	0,04

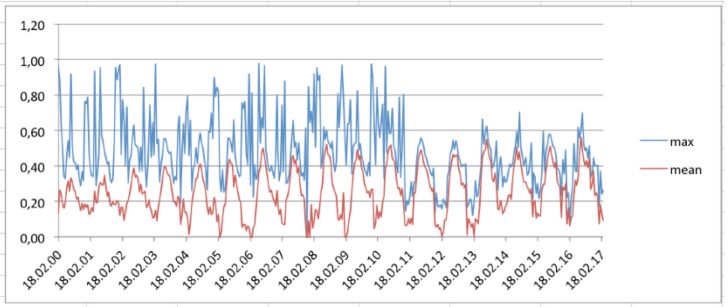


Figure 5.8: Spreadsheet file output showing statistical summaries (min, max, mean, and standard deviation) for an area of interest (polygon).

5.1.4 Implementation: Middleware software architecture and web services

The implementation of the software architecture and web services, which frames the overall middleware, are described in the following subsections. In addition to a description of the administration (Subsection 5.1.4.1) and the geospatial web services (Subsection 5.1.4.2), a definition of processing web services for data discovery, integration, and analysis is provided (Subsection 5.1.4.3).

5.1.4.1 Content Management System

The CMS provides the overall software to manage user authentication, areas of interest, registered data, and the analyses conducted. RESTful web services are provided by the CMS for the registration of areas of interest, data ingestion, analysis execution, and user registration and authentication. In addition, a list of available EO data collections and analysis tools linked to the data collections are managed through the CMS. Although in the latest middleware application the *Django Web Framework* is used as a CMS, older instances of the middleware system are based on *Drupal CMS* (see Chapter 6, “Example use cases”). As the client applications make use of the RESTful services, the CMS software is exchangeable. The data in the CMS is stored in a *PostgreSQL* database with geospatial support provided by the *PostGIS* extension. This enables the storage of geometries from the areas of interest and provides support for geospatial operations.

5.1.4.2 Geospatial web services

Visualization and access services are provided using services compliant with OGC specifications for geospatial data, which is available as the output of the data integration and data analysis processes. Each of the processes contains its own service instance using an individual configuration file for the open source software *MapServer*. This allows for the simple management of the OGC services as a removal of the directory with the analysis or integration results automatically removes the OGC service for the process undertaken. For the data visualization, a set of visualization styles have been prepared for each data type (e.g., vegetation index product, snow cover product, and land surface

temperature product) to convert geospatial raster data into RGB images. The provision of raster time-series data is based on the TIME extension of OGC WMS and WCS.

The open source software *pycsw* is used to publish metadata catalogues with the OGC CSW specification. Hence, each user registered in the middleware can have his or her own instance of the metadata catalogue. The open source software *PyWPS* provides services for data discovery, access, processing, and analysis based on the OGC WPS specification. The CMS backend software acts as a proxy for long-running processing services that are executed asynchronously.

5.1.4.3 Processing web services

Data discovery, integration, and analysis are made available using web services compliant with the OGC WPS specification. Each process managed by the middleware is registered in the CMS to retain the references to the inputs and outputs of the process. The monitoring of asynchronous process executions allows for notifying users with an email when the process is complete. In the following paragraphs, the processing services for data discovery, data integration, and data analysis are specified, described, and example service requests are presented.

Data discovery service

In general, geospatial data discovery is provided by the specifications of OpenSearch or OGC CSW. As neither of these meet the requirements of the user personas described in this thesis, the OGC WPS specification has been used for the specific use case of EO time-series data discovery. As a processing service, several user-aligned processing tasks can be integrated into the discovery service, such as the provision of multiple output formats, summary statistics, and additional calculations for the area of interest. In contrast to the general discovery specifications mentioned above, the WPS interface allows for flexible inputs and outputs for each process, which are described in detail in Table 5.1 for a data discovery process.

The processing service for EO data discovery designed in this thesis consists of an interface for searching for EO data based on a location defined as an input parameter. Both point and polygon geometries can be used as location input described using the well-known-text (WKT) format. A list of EO data collections supported by the discovery process needs to be defined by the processing service, as connections to external databases need to be made for the purpose of data discovery. Further queries of the metadata can be set up as a single input string of the WPS process. The additional input parameter “MinOverlap” (minimum overlap percentage) can be used to filter the resulting scenes based on the spatial overlap between the scene geometry and the given area of interest

(location input). Several outputs are provided to the user: The resulting scenes discovered based on the input parameters in several formats (e.g., CSV, JSON, or Shapefile) as well as a figure (“SummaryChart” output), which can be directly used for further interpretation of the results (see Subsection 5.2.3).

The discovery process can be executed with a single WPS execute request (Figure 5.9). Depending on the request, either all of the outputs or a single output can be returned to the user. Figure 5.9 shows an example WPS request and the CSV output file for a given polygon and Sentinel–1 GRD data.

Table 5.1: Inputs and outputs for a user-specific EO time-series data discovery service.

	Name	Type	Description
Input	Location*	WKT geometry	Geometry in well-known-text format (e.g., Point or Polygon)
Input	Datasets*	String	Multiple selection of EO data collections connected within this service
Input	MinOverlap	Integer	Minimum overlap percentage between scene geometry and location input
Input	Query	String	Filter query for discovery
Output	CSV	File URL	Results in CSV format
Output	JSON	File URL	Results in JSON format
Output	Shapefile	File URL	Results in Shapefile format
Output	SummaryChart	File URL	Summarized results in PNG chart
Output	SummaryCSV	File URL	Summarized results in CSV format

* Mandatory

```
http://artemis.geogr.uni-jena.de/cgi-bin/testbox.cgi?service=WPS&version=
1.0.0&request=Execute&identifier=s1_datahub_test&datainputs=[wkt=POLYGON((
13.59 55.79, 14.72 55.84, 14.10 58.48, 13.00 58.43, 13.59 55.79));
product=GRD;minoverlap=70]&rawdataoutput=output
```

```
Title,                          Overlap,  Link,        Geometry
S1A_EW_GRDM_1SDH_20150427T…,    0.94,     Download,   POLYGON ((...))
S1A_EW_GRDM_1SDH_20150427T…,    1.00,     Download,   POLYGON ((...))
S1A_EW_GRDM_1SDH_20150422T…,    0.87,     Download,   POLYGON ((...))
S1A_EW_GRDM_1SDH_20150415T…,    0.94,     Download,   POLYGON ((...))
S1A_EW_GRDM_1SDH_20150415T…,    1.00,     Download,   POLYGON ((...))
S1A_EW_GRDM_1SDH_20150410T…,    0.87,     Download,   POLYGON ((...))
...
```

Figure 5.9: WPS discovery request (top) and an example of CSV output (bottom).

Data integration service

User-aligned services for EO time-series data access need to consider many geospatial processing steps, such as clipping and merging to the area of interest, implementing quality masks, and scaling factors, as well as handling missing values. None of these can

be easily provided in the existing access specifications for raster data, such as OGC WCS or OGC WCPS. Although OGC WCPS can conduct data processing while accessing, only processed data is returned—no additional output formats can be provided to support the output formats for different user personas. Thus, a processing service for data access has been investigated in this thesis (Table 5.2). Specifically, data integration into the middleware has been considered in addition to general data access. Therefore, data access not only passes the data to the user but also stores and manages the data requested by the user in the middleware.

For any data access request, the location and the name of the EO data collection are mandatory to start the data access process. The location input supports the WKT format using points or polygons. The dataset input provides a predefined list of datasets, such as Landsat–8, Sentinel–2, Sentinel–1, and so forth. Only data from EO data collections that are connected to the middleware are supported by the process and can thus be accessed and integrated. Quality masks and scaling factors are not available for all the collections and depend on the collection selected. This is reflected in and implemented by the data access process. Further queries used to filter the resulting scenes (e.g., maximum cloud coverage, time ranges) can be added optionally to the data access request.

Different outputs are considered for single pixel and polygon-based raster extraction (Table 5.2). As mentioned above, all the data accessed is stored in the middleware—in contrast to the OGC WCS/WCPS specification. While the extraction of a single pixel extraction is less complex and only a few additional files are required (e.g., time-series plot, decomposition plot, and data CSV file), data access for areas of interest results in individual geospatial data files for each date. For the latter, statistical summaries, such as time-series values for the mean, minimum, maximum, and standard deviation, are calculated and provided as output. Both single pixel extractions and area-based extractions are provided in addition with OGC-compliant services, using either the OGC SOS for pixel extractions or the OGC WMS for visualization and the OGC WCS for data downloads for area-based extractions.

The concept of the unique identifier output (UUID output) allows users to reference the accessed and integrated data in other services, such as the analysis service or the download service (see Subsection 5.1.2.2 for service chaining). Users do not need to use the OGC WCS to download the data to their local computer; instead, the complete processing directory can be downloaded as a zipped archive file using an additional download service in the middleware. This zipped archive file contains the complete folder

of the unique identifier, which also includes results from analysis services conducted after the data integration.

Figure 5.10 shows an example WPS request as well as resulting outputs (excerpt). The data integration request is conducted for the Enhanced Vegetation Index layer of the MODIS Vegetation Index dataset (MOD13Q1) and a given point of interest. The directory listing of the UUID (Figure 5.10, bottom-left) can also be accessed using a web browser.

Table 5.2: Inputs and outputs for a user-specific EO time-series data access service.

	Name	Type	Description
Input	Location*	WKT geometry	Geometry in well-known-text format (e.g., Point or Polygon)
Input	DatasetName*	String	Single selection of an EO data collection
Input	Query	String	Filter query for discovery used for data access (e.g., only specific type)
Output	Properties	File URL	Text file with dataset properties (e.g., no data value, begin, end, scale factor)
Output	TimeseriesCSV	File URL	Result in CSV for point geometry
Output	TimeseriesChart	File URL	Time-series chart for point geometry
Output	TimeseriesFiles	File URL	CSV file that connects date and geospatial raster file
Output	SummaryChart	File URL	Statistically summarized results in PNG chart for polygon geometry
Output	SummaryCSV	File URL	Statistically summarized results in CSV format for polygon geometry
Output	UUID	String	UUID to re-use the dataset as input for analysis processes

* Mandatory

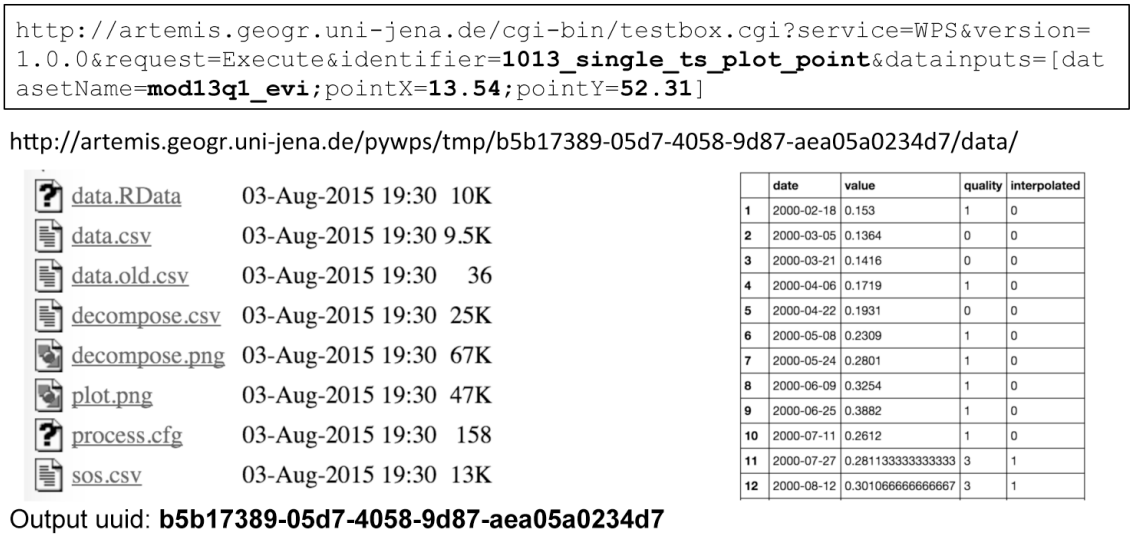


Figure 5.10: WPS access/integration request (top), file directory output (bottom left), and CSV output (bottom right). The UUID output is a unique identifier for re-using the data integrated into the system in analysis processes.

Data analysis service

EO time-series analysis processes can only be conducted based on existing input data. Thus, either data needs to be uploaded to the server in the correct data format and data structure, or it first needs to be integrated with the middleware service for data access and integration (see the previous paragraphs). Using a UUID for the previously integrated dataset as specified by the data integration service allows referencing of the input data within the analysis process.

Table 5.3 describes the inputs and outputs of an analysis process in general. Each analysis process needs to have different input parameters as they provide specific parameters on their own. A mandatory input is the reference to the input data, for example, using the UUID described above. Outputs of an analysis process need to be carefully thought for each of the algorithms provided as a web service and included in the middleware:

- What is relevant for users to understand the analysis result?
- What kind of figure simplifies the resulting output of the analysis?

Both questions need to be answered separately for any algorithm and a good compromise needs to be found to serve different user personas. In addition, any geospatial data should be made available with an OGC-compliant web service for interactive visualization (OGC WMS) and access (OGC WCS for raster data, OGC WFS for vector data). Styling information is required for each of the geospatial data outputs that are available as a visualization service. For each of the analysis processes conducted, an independent OGC-compliant service containing all geospatial data of the process is provided by the middleware using MapServer software.

Figure 5.11 shows two example WPS requests for breakpoint detection and trend calculations, both with the resulting output figure. The UUID for the data access and integration service allows the chaining of access and analysis services. Therefore, both need to be available: the data access process needs to provide the UUID and the analysis process needs to integrate the data structure based on the UUID. In addition, the algorithms need to be designed to fulfill the requirements of users by providing documentation steps, such as log files, as well as suitable output formats (e.g., PNG figures).

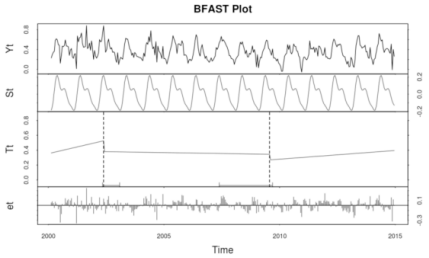
Table 5.3: Inputs and outputs for a user-specific EO time-series data analysis service.

	Name	Type	Description
Input	UUID*	String	Relation to the previously accessed/integrated dataset (UUID output of data access process)
Input	Multiple input parameters	String, Number, Complex	Dependent on algorithm/tool
Output	Multiple output parameters	String, Number, Complex	Dependent on algorithm/tool
Output	Figure	File URL	Main figure of the output as graphic file
Output	ResultsOGC	URL	OGC-compliant service to serve OGC WMS, WCS, WFS for geospatial outputs

* Mandatory

Breakpoints for vegetation time-series

```
http://artemis.geogr.uni-jena.de/cgi-bin/
testbox.cgi?
service=WPS&version=1.0.0&request=Execute&i
dentifier=2010_single_ts_bfast_point&datain
puts=[uuid=b5b17389-05d7-4058-9d87-
aea05a0234d7]
```



Trends for vegetation time-series data

```
http://artemis.geogr.uni-jena.de/cgi-bin/
testbox.cgi?
service=WPS&version=1.0.0&request=Execute&
identifier=3010_single_ts_greenbrown_point
&datainputs=[uuid=b5b17389-05d7-4058-9d87-
aea05a0234d7]
```

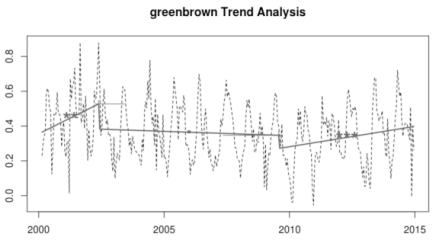


Figure 5.11: WPS analysis requests for breakpoint (top) and trend calculations (bottom), with a figure output on the right.

5.2 Service Brokering for Multi-source Data Discovery and Access

Discovery of and access to EO data has improved continuously in recent years. Standardized web service interfaces exist that are used by various data providers, such as ESA, NASA, and USGS. However, data providers often offer additional interfaces to provide functions and data formats that are more suitable—in contrast to the standardized interfaces. This leads to various web service specifications, which need to be known by users who want to discover and access multi-source EO data. Thus, a key issue is to find and obtain access to EO time-series data without the need to learn how to use the various tools and services that are provided by each data provider. A unique standardized and easy-to-use interface needs to be developed that covers not only data discovery and access, but also focuses on the needs of the user. Although international organizations, such as GEO and CEOS, have established centralized brokering approaches that include all kind of EO data, the resulting data is still too complex for responses to simple questions, such as "*What kind of data is available for specific years and months in my area of interest?*"

Finding suitable scenes of EO time-series data for a user-defined area of interest is an essential task. Searching for useable data can be a difficult task as data needs to be of good quality and may have to meet several conditions. For example, when working with optical data, cloud coverage of satellite scenes is an important factor when searching for data. However, if the area of interest is smaller than the satellite scene and clouds do not cover this area even if the whole scene is mostly covered, then the cloud coverage information of the satellite scene is useless for the real area of interest. Thus, on-the-fly data processing for the specific area of interest while searching for data can be considered to solve such issues and enhance the discovery of freely available EO data. In addition, automated approaches for multi-source EO data access need to be established to simplify the downloading of EO data. In most cases, only links to satellite scenes are provided in data discovery; further processing needs to be undertaken by the user. For some applications, on-the-fly extraction of EO time-series data for a user-defined area is necessary. The downloading and processing of whole satellite scenes is not an option—a web service can solve this.

In this section, a brokering approach to centralize EO data discovery and access based on multiple data providers is described. In addition to the service-brokering concept (Subsection 5.2.1), the metadata model and additional extensions are described (Subsection 5.2.2). User-aligned output formats are further investigated to provide easy-to-understand overviews and summaries (Subsection 5.2.3). Finally, the implementation of the service brokering as a Python package is described (Subsection 5.2.4).

5.2.1 The concept of web service brokering

Brokering services forward requests to connected external web services. Therefore, a translation of the initial request needs to be adapted for each of the connected data providers. As each data provider offers different kinds of web services, a separate translation of the user's request to the specifications used by the data provider is necessary. As shown in the review chapter, existing data catalogues provide different web service interfaces, filter capabilities, and metadata for individual EO data. The following main limitations of currently available data discovery services need to be addressed when designing the brokering service:

- Different web service specifications need to be known by the user.
- Different data providers need to be requested individually by the user.
- The quality of metadata needs to be enhanced (e.g., quick-look images and sun angles).
- Additional filter capabilities based on the area of interest are required.
- Simple overview charts are missing (e.g., what kind of data is available when).

Although brokering services already exist, such as FedEO, which supports many of the EO missions used in this thesis, a new brokering concept is necessary to meet the requirements of multiple user personas. Therefore, this concept focuses on "human–machine" communication that provides multiple output formats to meet the requirements of the different user personas. To ensure good quality of the metadata, the most relevant metadata items are extracted from multiple data providers. The metadata from the connected data providers is translated into a common abstract metadata model along with the full metadata records. Extensions of the metadata model allow the integration of on-the-fly computed properties and interactive visualization services (see Subsection 5.2.2).

The following subsections describe the methods for service brokering and the quality enhancements of metadata in the brokering concept.

5.2.1.1 Brokering methods

The brokering approach provides a single entry point to search the external metadata catalogues registered in the broker. Brokering for data discovery and access can be separated into four main methods:

1. List of available collections (various EO missions)
2. Search for available satellite scenes
3. Downloading of satellite scenes
4. Direct extraction of time-series data (if applicable)

The brokering framework needs to know how to retrieve the list of available collections, search for satellite scenes, obtain access to data from each data source, and extract data for a given location. Depending on the available functionalities of each data provider, all these functions need to be implemented within the brokering software. Figure 5.12 shows the general concept of the brokering software for multi-source EO data that provides access to different data providers based on uniform methods (get_datasets, search_granules, get_data, and extract_data). These methods are described in the following paragraphs.

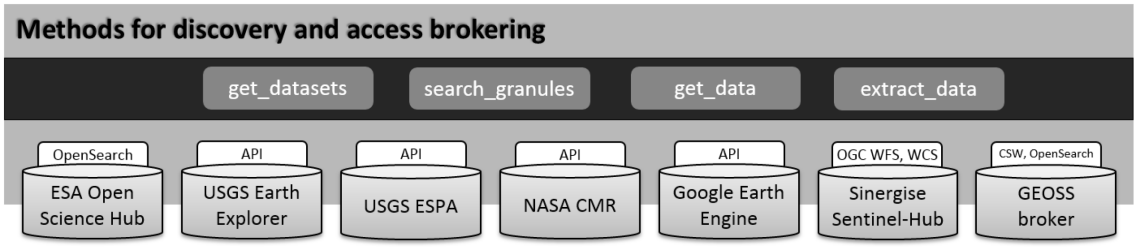


Figure 5.12: Concept of the multi-source EO data discovery and access broker.

List available collections (get_datasets)

Metadata catalogues provide access to one or multiple collections (EO missions or several products). With this method, a list of collections available within the catalogue is provided, which can then be used to search for individual scenes. This method is not always available within the specification used by the data provider (e.g., OpenSearch). In general, OGC CSW does not support such queries either. However, extensions provided within the ESA FedEO broker support this. Within the brokering framework developed in this thesis, a list of collections available is generated for metadata catalogues that do not provide this functionality.

Search for satellite scenes (granules) in a selected collection (search_granules)

Individual scenes for a given location and time range can be searched for within the selected collection. In hierarchical metadata catalogues, the collection acts as a parent metadata record. A list of satellite scenes available for the query with the given filter parameters is provided, with multiple properties for each scene (e.g., UUID, cloud cover, footprint, and begin and end date) depending on the results from each metadata catalogue (see Section 4.1). For some metadata catalogues, it is necessary to request an additional web service (e.g., NASA CMR) to obtain the full metadata record.

Data download (get_data)

Each of the satellite scene's metadata responses provides a download link to access the data. Depending on the data provider, either direct links to the complete scene (e.g., ESA/Copernicus Open Access Hub) or links to web pages (e.g., USGS Earth Explorer)

are available. In many cases, login credentials for each individual data provider are necessary to download the data. A download can also be handled by a third-party data provider using the search results from another metadata catalogue (e.g., using Amazon Web Services for a data download with search results from USGS Earth Explorer).

Data extraction (extract_data)

A few data providers (e.g., Google Earth Engine and Sinergise Sentinel-Hub) support the direct extraction of EO time-series data. In comparison to the data download method, this allows the accessing of data without downloading complete scenes. In this thesis, only statistical summaries of the area of interest over time for areas or time-series values for single pixels are provided with this method.

5.2.1.2 Metadata quality enhancement

Collections available from different metadata catalogues or data providers may contain different sets of metadata. Although most of the metadata relevant to users can be extracted from the original metadata resource, data providers may provide different thumbnails and quick look images as well as links for the downloading of data. To be able to extract the best metadata and the files referenced to it, it may be necessary for individual satellite scenes to obtain metadata from different data providers. For example, quick-look images from Sentinel–1 demonstrate that the NASA Alaska Satellite Facility provides better images as the original ESA/Copernicus Open Access Hub (see Figure 5.13). Thus, providing Sentinel–1 quick-look images from NASA is preferable to using the original ones from ESA. Another example shows that quick-look images from ESA/Copernicus Open Access Hub can only be accessed with a user login; the same quick-look images from the USGS Earth Explorer for Sentinel–2 can be accessed without login credentials, which makes it easier to integrate them in own applications.

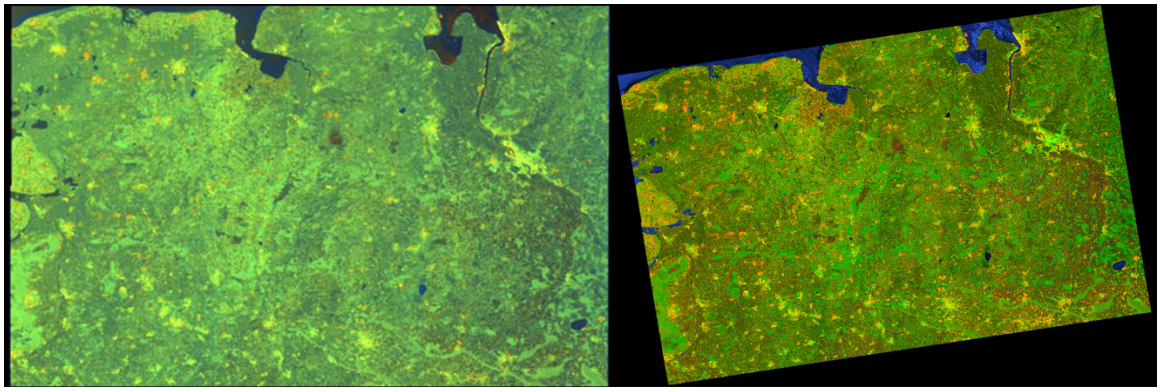


Figure 5.13: Comparison of quick-look images of a Sentinel–1 scene.[35] On the left, the image from the ESA/Copernicus Open Access Hub; on the right, the image provided by NASA. The latter includes enhanced colors and is spatially oriented within the bounding box.

[35] Granule-ID: S1A_IW_GRDH_1SDV_20180810T171655_20180810T171720_023187_0284ED_959E

5.2.2 Metadata model

A unique metadata model for the discovery of satellite scenes brings together the results of each external discovery service. As each data catalogue provides different fields of attributes, a set of three mandatory items has been drafted:

- Unique identifier (id)
- Geometry of the scene
- Start date and time of the scene (date of acquisition).

Depending on the data provider, these can be extended with additional data. Table 5.4 shows the metadata items supported by each data provider. A set of common metadata items is defined to ensure the good quality of the data discovery:

Unique identifier (id)	Title	Geometry
Time start	Time end	Updated date time
Download URL	Browse/Thumb URL	Metadata URL
Filename	Size	Cloud cover percentage

Further information about collection-specific metadata can be added depending on the type of data (e.g., optical or radar), if this information is provided by the discovery service. As each of the data providers may name the attributes differently, a mapping of the metadata to the common metadata model needs to be integrated into the brokering framework, as shown in Figure 5.14. Items from the individual query responses for each metadata catalogue are mapped to the common metadata model.

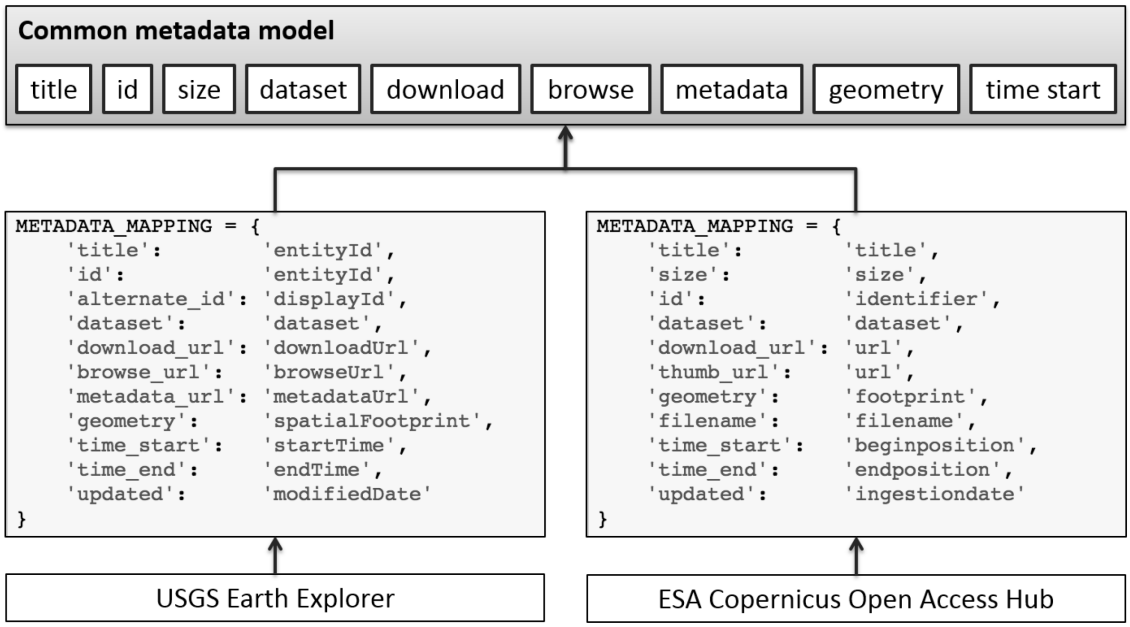


Figure 5.14: Metadata mapping between multiple data providers to establish a common metadata model.

Table 5.4: Metadata items from the common metadata model and their appearance in external metadata catalogues registered in the broker (* can be generated on request).

Metadata item	GEOSS broker	ESA FedEO	CEOS CWIC	ESA Sentinel	USGS EE	NASA CMR	Google Earth Engine	Sentinel-Hub
Identifier		☑	☑	☑	☑	☑	☑	
Geometry	☑	☑	☑	☑	☑	☑	☑	☑
Start date	☑	☑	☑	☑	☑	☑	☑	☑
End date	☑	☑		☑	☑	☑	☑	
Filename				☑				☑
File size				☑				
Download URL	☑	☑		☑	☑	☑	*	
Browse URL		☑		☑	☑	☑	*	
Metadata URL		☑		☑	☑			
Cloud cover	☑	☑					☑	☑

5.2.2.1 Metadata extensions

An extended metadata model enhances the discovery of satellite scenes with filter capabilities and additional metadata elements to better meet the requirements of the user personas. Figure 5.15 shows the connections between these methods in a flowchart, starting with the user's discovery request and extending to the resulting list of scenes. Three different methods are described in the following subsections: on-the-fly computed properties, additional download links, and interactive satellite scene browsing.

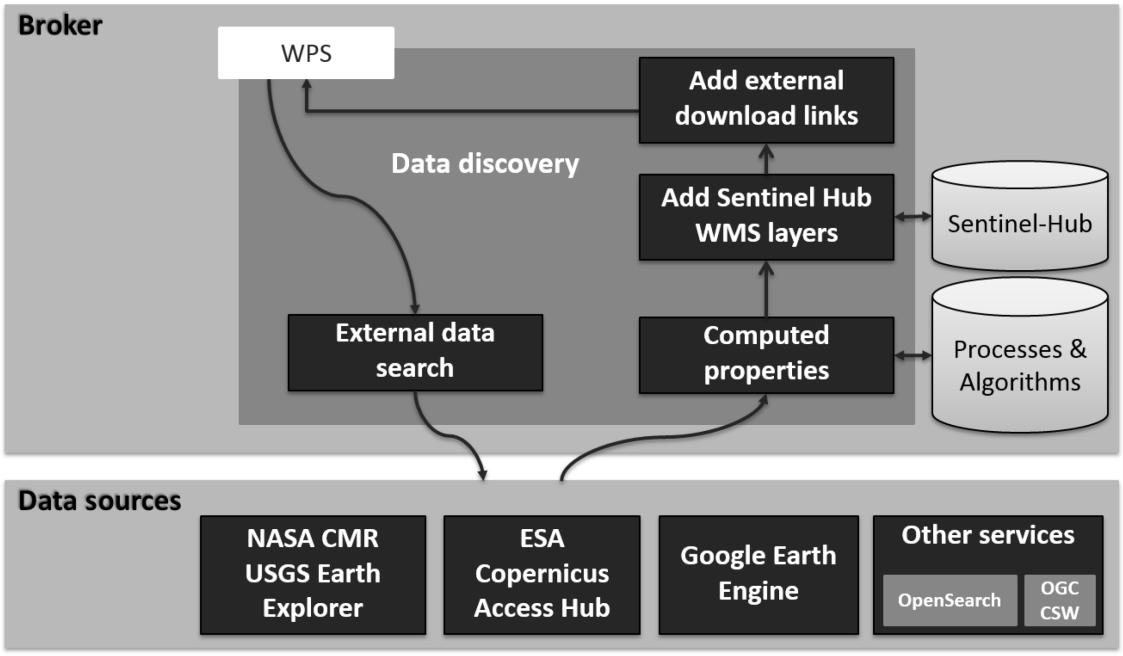


Figure 5.15: The data discovery broker integrates post-processing steps after the external data search has been conducted and before the results are sent back to the user as a WPS response. This includes optional on-the-fly computed properties (e.g., overlap calculation, cloud cover based on the area of interest), added visualization layers (e.g., from Sentinel-Hub), and added external download links (e.g., Google and USGS).

5.2.2.2 On-the-fly computed properties

Metadata properties of satellite scenes always relate to the full geospatial extent of the scene. However, in many cases, users search for data for a specific area of interest that either covers several scenes or is smaller than the satellite scene. In both cases, it is relevant to calculate properties, such as cloud cover percentage or the size of the overlapping geometry, on-the-fly based on the area of interest provided as input by the user. This allows users to filter satellite scenes based on the real area of interest, in contrast to the geometry of the satellite scene. Two on-the-fly computed properties are described in the following paragraphs.

Calculation of overlap between scene geometry and the area of interest

In some cases, the satellite scene only covers a minimal percentage of the area in which the user is interested. Therefore, filtering by overlapping area is introduced to minimize the satellite scenes not relevant to the area of interest. For each of the resulting scenes, a spatial intersection is effected and the size of the intersected area is set in relation to the geometric size of the scene. This parameter is added to each metadata item as an additional on-the-fly computed metadata property.

Calculation of cloud cover for the area of interest

Specific scene properties need to be calculated based on the area of interest in order to be suitable for use as filter parameters (e.g., cloud coverage). To this end, data analysis needs to be undertaken—either after the data download or using an online processing tool available in a web service. Google Earth Engine was used to compute the cloud coverage of Landsat scenes in the area of interest defined by users in their requests. Figure 5.16 shows the Google Earth Engine Playground, including the script editor, scene visualization, and the area of interest (red area). In the script, the selected satellite scene is loaded and clipped to the area of interest; afterwards the “simpleCloudScore” algorithm from Earth Engine is executed (Google 2017, 2018a). This algorithm calculates “simple” cloud coverage for the area of interest of the Landsat scene using a combination of brightness, temperature, and the Normalized Difference Snow Index. Although Google states that “it is not a robust cloud detector” (Google 2018a), it is used here only as an example. Any other algorithm can be used for this kind of on-the-fly computed data property. The result of this algorithm is a raster image with cloud scores of between 0 and 100 percent for each pixel, which can be summarized as mean value. Using the Google Earth Engine API, this analysis can be conducted and retrieved as a JSON object (Landsat scene identifier with cloud cover percentage for the area of interest), which can be further processed in the brokering framework. The example used in Figure 5.16 shows that the cloud cover of the scene (20%) is much larger than in the area of interest (4%). As such,

filtering satellite scenes according to the cloud cover for the complete scene at a level of less than 20% would have removed this scene from the resulting list even though the cloud cover over the area of interest is much less.

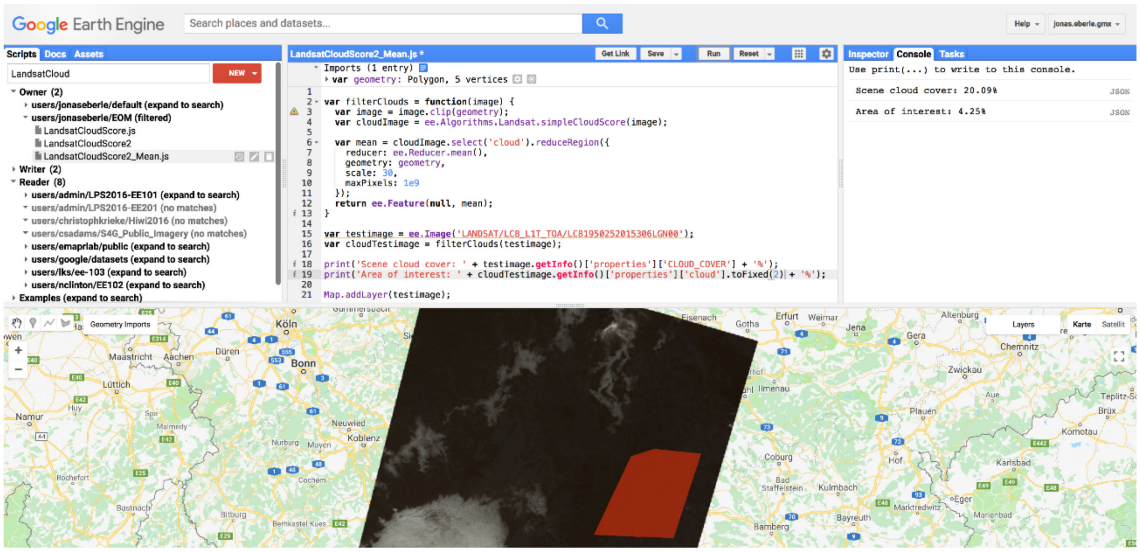


Figure 5.16: Cloud cover calculation for a Landsat-8 scene in an area of interest (red area) in Google Earth Engine. While the cloud cover of the scene is 20 percent, it is only 4 percent in the area of interest.

5.2.2.3 Additional download links

Today, EO data is available from a variety of data providers. When searching for data in multiple metadata catalogues, download links from multiple data providers are available (e.g., USGS Earth Explorer or ESA/Copernicus Open Access Hub). In addition, links can be generated from various data providers based on their metadata, such as direct download links for Amazon Web Services or CODE-DE (see Subsection 4.2.2). The advantages of showing additional download links are diverse. For example, for download links from Google Cloud, users need not have a user login and can access individual bands of scenes—in contrast to the original services from USGS Earth Explorer. In addition, the download speed or service performance may be different for each of the external platforms (e.g., an increased data download speed from the CODE-DE platform within German research networks).

5.2.2.4 Interactive satellite scene browsing

The visualization of individual satellite scenes is useful to obtain a first impression of the data. Using the Sentinel-Hub web service provided by Sinergise, the OGC WMS-TIME services can be used to visualize Landsat and Sentinel scenes based on data archives from ESA and Amazon Web Services. If access to the Sentinel-Hub WMS is available (a commercial interface key is required), a WMS link with multiple layers (e.g., used for natural color, false color, and NDVI) can be added to the metadata of each of the satellite

scenes. This WMS can then be used in own applications. For example, Figure 5.17 shows the integration of the visualization services in the “Satellite data explorer” on the GEO-Wetlands Community Portal. Although it is not possible to filter the WMS down to an individual scene, scenes from a selected day are automatically mosaicked and provided by the Sentinel-Hub services.

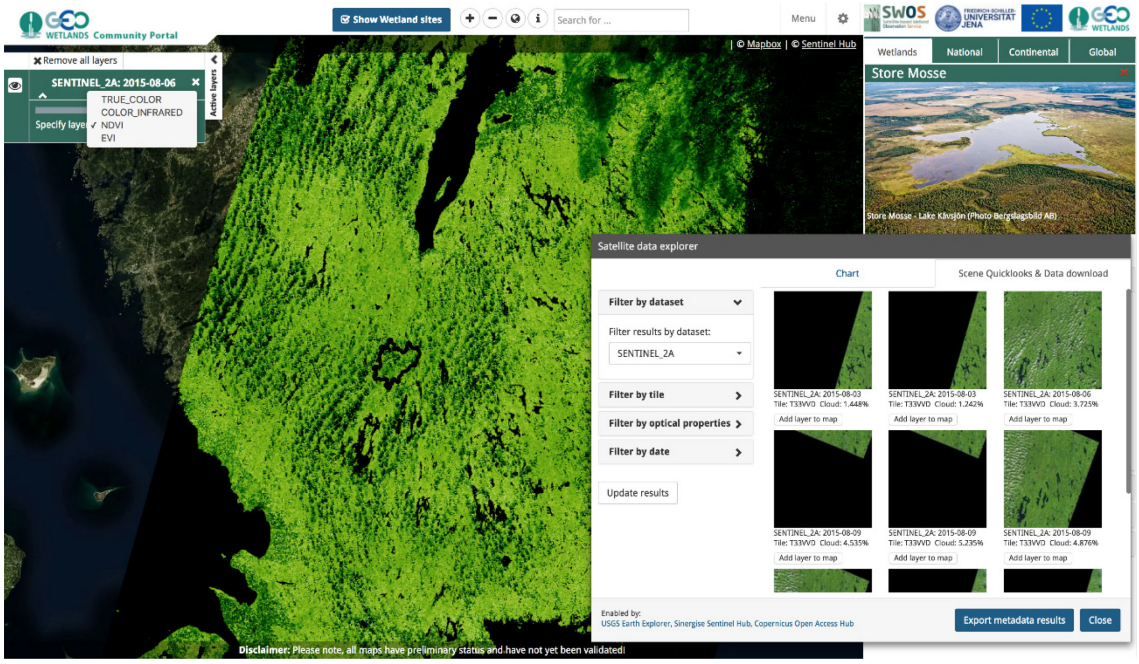


Figure 5.17: On-the-fly visualization of a satellite scene using the NDVI layer.

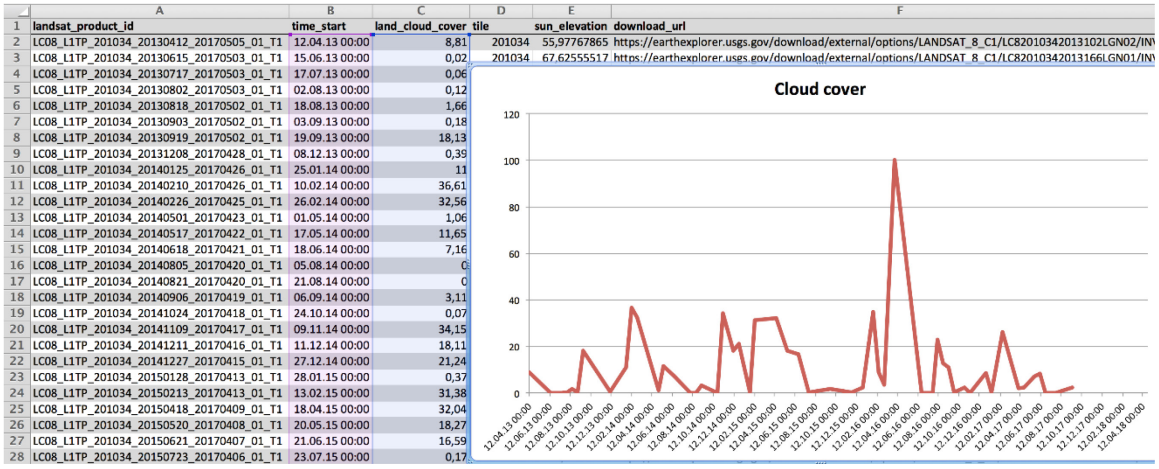
5.2.3 Discovery output formats

Various output formats of data discovery results are needed to fulfill different needs of applications and meet the requirements of user personas. To ensure ease of use of the data discovery results, summarized outputs can be provided in simple formats (e.g., PNG or CSV). To enable developers to build applications based on the discovery results, geospatial output files need to be available in additional formats, such as GeoJSON or SQLite database. Other output formats, such as CSV spreadsheet files and summarized outputs, are described in this subsection.

5.2.3.1 CSV spreadsheet files

In order to enhance the analysis of results from EO data discovery, the data resulting from a discovery request is converted to a CSV spreadsheet file so that it can used by any spreadsheet software or programming language. In contrast to the usual formats, XML and JSON, most users are familiar with the CSV file format. Both of these formats are especially suitable for developers but can often not be handled by other user personas. As an example of the CSV output, Figure 5.18 shows Landsat–8 scenes for a given area of interest (only parts of the columns available are shown). Using the CSV file, users can

directly work within spreadsheet software to analyze the results (e.g., plotting the amount of cloud cover of each scene over time, as shown in Figure 5.18).



	landsat_product_id	time_start	land_cloud_cover	tile	sun_elevation	download_url
2	LC08_L1TP_201034_20130412_20170505_01_T1	12.04.13 00:00	8,81	201034	55,97767865	https://earthexplorer.usgs.gov/download/external/options/LANDSAT_8_C1/LC82010342013102LGN02/IN
3	LC08_L1TP_201034_20130615_20170503_01_T1	15.06.13 00:00	0,02	201034	67,62555517	https://earthexplorer.usgs.gov/download/external/options/LANDSAT_8_C1/LC82010342013166LGN01/IN
4	LC08_L1TP_201034_20130717_20170503_01_T1	17.07.13 00:00	0,06			
5	LC08_L1TP_201034_20130802_20170503_01_T1	02.08.13 00:00	0,12			
6	LC08_L1TP_201034_20130818_20170502_01_T1	18.08.13 00:00	1,66			
7	LC08_L1TP_201034_20130903_20170502_01_T1	03.09.13 00:00	0,18			
8	LC08_L1TP_201034_20130919_20170502_01_T1	19.09.13 00:00	18,13			
9	LC08_L1TP_201034_20131208_20170428_01_T1	08.12.13 00:00	0,39			
10	LC08_L1TP_201034_20140125_20170426_01_T1	25.01.14 00:00	11			
11	LC08_L1TP_201034_20140210_20170426_01_T1	10.02.14 00:00	36,61			
12	LC08_L1TP_201034_20140226_20170425_01_T1	26.02.14 00:00	32,56			
13	LC08_L1TP_201034_20140501_20170423_01_T1	01.05.14 00:00	1,06			
14	LC08_L1TP_201034_20140517_20170422_01_T1	17.05.14 00:00	11,65			
15	LC08_L1TP_201034_20140618_20170421_01_T1	18.06.14 00:00	7,16			
16	LC08_L1TP_201034_20140805_20170420_01_T1	05.08.14 00:00	0			
17	LC08_L1TP_201034_20140821_20170420_01_T1	21.08.14 00:00	0			
18	LC08_L1TP_201034_20140906_20170419_01_T1	06.09.14 00:00	3,11			
19	LC08_L1TP_201034_20141024_20170418_01_T1	24.10.14 00:00	0,07			
20	LC08_L1TP_201034_20141109_20170417_01_T1	09.11.14 00:00	34,15			
21	LC08_L1TP_201034_20141211_20170416_01_T1	11.12.14 00:00	18,11			
22	LC08_L1TP_201034_20141227_20170415_01_T1	27.12.14 00:00	21,24			
23	LC08_L1TP_201034_20150128_20170413_01_T1	28.01.15 00:00	0,37			
24	LC08_L1TP_201034_20150213_20170413_01_T1	13.02.15 00:00	31,38			
25	LC08_L1TP_201034_20150418_20170409_01_T1	18.04.15 00:00	32,04			
26	LC08_L1TP_201034_20150520_20170408_01_T1	20.05.15 00:00	18,27			
27	LC08_L1TP_201034_20150621_20170407_01_T1	21.06.15 00:00	16,59			
28	LC08_L1TP_201034_20150723_20170406_01_T1	23.07.15 00:00	0,17			

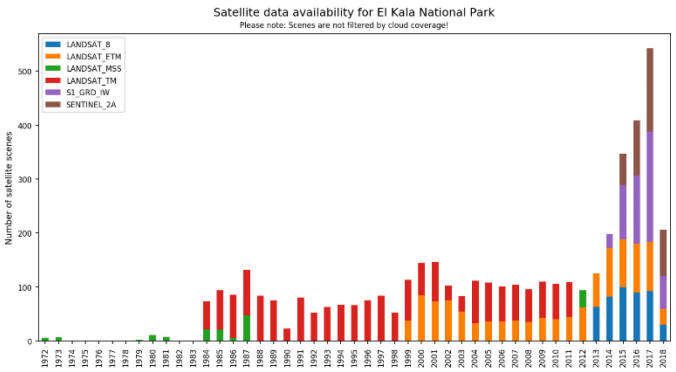
Figure 5.18: Spreadsheet file output showing scenes from a data discovery request, including scene metadata and download links.

5.2.3.2 Summary output by year and by satellite

An additional summary output format has been designed and is made available to provide information about what kind of EO data is available from which sensor when. Therefore, two additional outputs are calculated within the data discovery:

1) Graphic output shows the number of scenes per satellite sensor per year in a PNG file (Figure 5.19, left). Especially for historical years, it is easy to identify when data is available from which kind of satellite sensor.

2) Tabulated output shows the number of scenes per sensor in total, in addition to the temporal range that includes the year of first and last scene from the sensor requested (Figure 5.19, right). This output format is stored as a JSON file and as a spreadsheet CSV file. It allows quick insight into how many scenes are available from each sensor in the complete temporal dimension.



Total amount by sensor

Sensor	Scenes	Start	End
LANDSAT_MSS	161	1972	2012
LANDSAT_TM	1817	1984	2011
LANDSAT_ETM	1149	1999	2018
LANDSAT_8	457	2013	2018
SENTINEL-1_GRD_IW	517	2014	2018
SENTINEL_2A	400	2015	2018

Figure 5.19: Summary output generated from the discovery result list: graphical output per year and per satellite sensor (left); tabulated output per satellite sensor with the total number of scenes, including the year of first and last scene (right).

5.2.4 Implementation: EO time-series data discovery and access brokering

An EO time-series data discovery and access broker has been developed to harmonize discovery and access between multiple data providers. Different data providers, such as USGS Earth Explorer, ESA/Copernicus Open Access Hub, NASA CMR, and Sentinel-Hub from Sinergise, have been integrated and are available through the brokering software, provided as a pyEOM Python library, which is developed within this thesis. Access to Landsat and Sentinel satellite data as well as the MODIS sensor has also been integrated in the broker.

Figure 5.20 shows the system architecture of the broker and integrated services: The USGS archive of Landsat is requested from the USGS Earth Explorer; Sentinel–2 is requested from both the USGS Earth Explorer and the ESA/Copernicus Open Access Hub; Sentinel-Hub OGC services from Sinergise are used to query the ESA archive of Landsat and to add WMS visualization services to the satellite scenes discovered. The harmonized data discovery broker connects to the different specifications of the data providers and provides a uniform metadata response. In addition to the data discovery, automated data access has been integrated into the Python library.

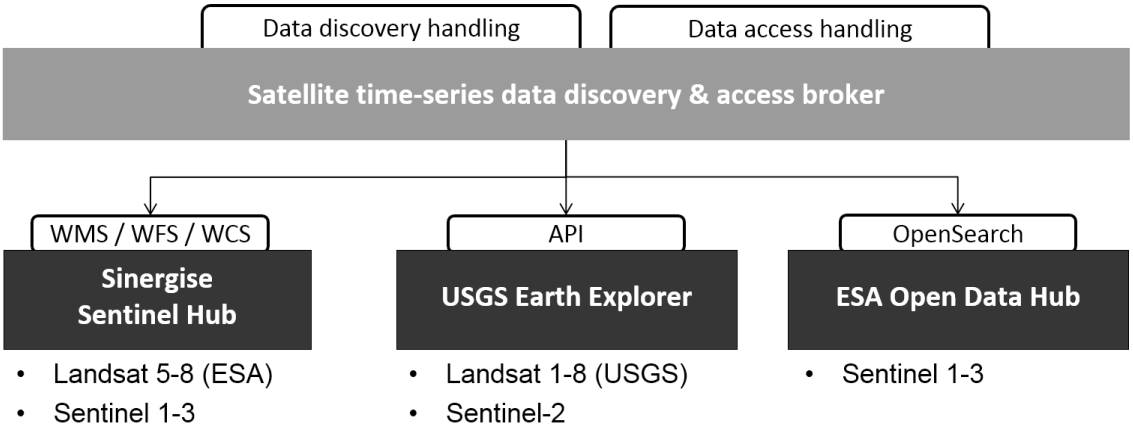


Figure 5.20: System architecture of the satellite time-series data discovery and access broker linked with three external data providers (Sinergise, USGS, and ESA), providing data discovery and data access methods.

The discovery component of the brokering software described in Subsection 5.2.4.1 provides a unique interface for the user to discover data; the access component described in Subsection 5.2.4.2 provides a unique interface for data access. Discovery and access to the data providers mentioned above have been integrated based on the descriptions in the previous sections. Users of this pyEOM Python library need to have user credentials from each data provider to be able to discover and access EO data.

5.2.4.1 Discovery brokering software

Within the discovery brokering software, USGS and ESA archives for both Landsat and Sentinel data can be searched for a given area of interest with a few lines of Python code (Listing 5.1). Based on the common and extended metadata models, uniform records of metadata are extracted and provided in different output formats. The main functions of the discovery brokering (discovery search, post-processing, and metadata export) are described in the following paragraphs. The pyEOM Python library makes use of external libraries, such as *Shapely*, *Fiona*, *Pandas*, *OWSlib*, and *GDAL*.

```
01 from pyEOM import SatelliteBroker
02
03 # Register brokering services (ESA Science Hub, USGS Earth Explorer,
Sinergise Sentinel-Hub)
04 broker_all = SatelliteBroker(
05     esa_scihub=('*username*', '*password*', ['S1*_IW_GRD*']),
06     usgs=('*username*', '*password*', ['LANDSAT_MSS', 'LANDSAT_TM_C1',
                  'LANDSAT_ETM_C1', 'LANDSAT_8_C1', 'SENTINEL_2A']),
07     sentinelhub=('http://example.org/v1/wfs/*apikey*', ['L5.TILE'])
08 )
09
10 # Search for all scenes based on given geometry (geom_wkt)
11 scenes = broker_all.search(geom_wkt)
12
13 # Optional post processing of initial search results
14 scenes_meta = broker_all.post_process_results(scenes, min_overlap=0.1)
15 scenes_meta = broker_all.retrieve_metadata(scenes_meta)
16 scenes_meta = broker_all.get_external_quicklooks(scenes_meta)
17 scenes_meta = broker_all.add_sentinelhub_wms_url(scenes_meta)
18
19 # Export data to GeoJSON file
20 broker_all.export_results(scenes_meta, 'results.json')
```

Listing 5.1: Python-based source code for undertaking data discovery.

Discovery search

With the initialization of the *SatelliteBroker* class, each individual data provider needs to be registered. Lines 4–8 in Listing 5.1 show the registration of the ESA/Copernicus Access Hub (esa_scihub), the USGS Earth Explorer interface (usgs), and the Sinergise Sentinel-Hub services (sentinelhub). For any of these services, login credentials or an interface key are necessary. As described in the concept of the web service brokering, any of the data providers' methods for login, scene search, and metadata mapping are implemented. Supported by this brokering software, data providers have been integrated with these methods in individual Python files. The discovery search of all data providers registered with their respective EO data collections is conducted using the "search" method of the *SatelliteBroker* class. This method needs to have an area of interest in the WKT format as the first argument (Line 11 in Listing 5.1). In this step, the different methods of login, scene search, and metadata mapping for each registered data provider are executed. Finally, a list of resulting scenes is returned after duplicate scenes from multiple data providers are automatically removed.

Postprocessing of discovery results

Postprocessing steps can be conducted optionally to enhance the discovery result. An additional filter for the initially scene results can be applied to remove scenes with a minimal overlap with the area of interest (see Subsection 5.2.2.2). A minimum overlap percentage can be defined between zero and one in order to reduce the number of scenes. For example, if the overlap between scene geometry and area of interest geometry is less than 10 percent, scenes are removed from the resulting list when the parameter is set to 0.1. This is effected with the “post_process_results” method with the parameter “min_overlap” (Line 14 in Listing 5.1).

Additional metadata can be retrieved for each satellite scene that provides a metadata URL and the method “read_metadata” is implemented for the data provider. As some of the catalogues do not provide all the metadata for the scenes in the discovery search, further metadata needs to be extracted from external URLs. This has been implemented for the USGS Earth Explorer interface and the Sinergise Sentinel-Hub services, as neither provide the full metadata record in the discovery results. This can be effected using the “retrieve_metadata” method of the *SatelliteBroker* class (Line 15 in Listing 5.1).

When satellite scenes are discovered in data catalogues from several data providers, some of the metadata information can be enhanced. For example, the quick-look images for Sentinel–1 scenes at the NASA Alaska Satellite Facility are geo-referenced with an enhanced color stretching (see Subsection 5.2.1.2). Although the data discovery search was conducted in the ESA/Copernicus Open Access Hub, the link to the quick-look images can be replaced using the “get_external_quicklooks” method for the Sentinel–1 results (Line 16 in Listing 5.1).

For the visualization of satellite scenes, additional WMS links, layer names, and temporal ranges (the acquisition date of the scene) can be added to the resulting metadata of each scene. The OGC WMS service published by Sinergise provides visualizations for Landsat and Sentinel time-series data. Using the “add_sentinelhub_wms_url” method, three metadata items (“senhub_wms_url,” “senhub_wms_layers,” “senhub_wms_time”) are added to each scene in the returned list (Line 17 in Listing 5.1).

Metadata export

The *Pandas* Python library is used to calculate a summary of the resulting scenes. Depending on the data format given as an argument (“format”) for the “export_results” method (Line 20 in Listing 5.1), different steps are conducted: Using the CSV format alone, the built-in *Pandas* export to CSV is accomplished. When using either the default format GeoJSON or other geospatial data formats additional files are stored:

- Lists of tiles that have appeared, collections, start and end year, as well as the total number of scenes per collection are calculated and exported in JSON format.
- The full list of scenes and their metadata is exported to either GeoJSON or Shapefile format depending on the “format” argument.
- An SQLite database can be exported based on the previously generated GeoJSON file. The *ogr2ogr* command-line tool is used to convert GeoJSON to SQLite.

Both graphic and tabular output, which summarize the resulting scenes, can be generated with the “generate_table_chart” method of the *SatelliteBroker* class. The *Pandas* library is used to generate summaries for each collection (e.g., Landsat MSS, Landsat TM, Landsat ETM, Landsat 8, Sentinel–1, Sentinel–2). Two output formats are provided, as previously shown in Figure 5.19 (Subsection 5.2.3.2):

1) A list of datasets with the total number of scenes, year of first scene, and year of last scene stored into a JSON and a CSV file.
2) A graphic summary of available satellite scenes grouped by collection and year stored in a PNG file.

5.2.4.2 Access brokering software

In this part of the brokering software, access to multi-source EO time-series data is provided. Two different methods are implemented:

1) A scene-based data download to the computer based on a previously conducted data discovery search.

2) Statistical extraction of EO time-series based on an area of interest (point or polygon) for a selected collection.

For each of the data providers registered to support data access, login credentials or specific user rights need to be available, such as access to NASA LPDAAC for downloading MODIS data, credentials for USGS or ESA for their data access interfaces, or a whitelisted service account to properly use the Google Earth Engine Python library.

Scene data download

Scene data downloads can only be conducted following the execution of a discovery search. In general, the links in the “Download URL” metadata item for each satellite scene are used to access each scene within the “download” method of the broker instance (Line 14 in Listing 5.2). For some of the data providers (e.g., USGS ESPA), data access is not provided with a direct file download but rather using an ordering process. In this case, the

order is submitted and the process waits until data is available for download. In addition to data discovery, access to download URLs has also been integrated for each of the data providers. Thus, the “get_data” method has been integrated in the implementation of the data provider’s configuration file.

```
01 from pyEOM import SatelliteBroker
02
03 # Register brokering services (ESA Science Hub, USGS Earth Explorer,
Sinergise Sentinel-Hub)
04 broker_all = SatelliteBroker(
05     esa_scihub=('*username*', '*password*', ['S1*_IW_GRD*']),
06     usgs=('*username*', '*password*', ['LANDSAT_MSS', 'LANDSAT_TM_C1',
                'LANDSAT_ETM_C1', 'LANDSAT_8_C1', 'SENTINEL_2A']),
07     sentinelhub=('http://example.org/v1/wfs/*apikey*', ['L5.TILE'])
08 )
09
10 # Search for all scenes based on given geometry (geom_wkt)
11 scenes = broker_all.search(geom_wkt)
12
13 # Download all scenes from data discovery
14 broker_all.download(scenes)
```

Listing 5.2: Python code to download all scenes found during the discovery search.

Data extraction

Google Earth Engine and Sinergise Sentinel-Hub services allow for on-the-fly time-series data extraction. The “extract” method of the brokering instance is integrated to standardize extraction for the registered data providers. For each of the data providers, the “extract_data” method is implemented in the provider’s configuration file, otherwise no data extraction is possible. To use the extraction method from the broker, several inputs are required: A geometry in WKT format containing the area of interest for statistical computations, a temporal range, the data provider selected, and the name of the collection supported by the selected data provider (Line 9 in Listing 5.3). The resulting object contains a time-series “DataFrame” object from the *Pandas* library, which can be used for further processing.

```
1 from pyEOM import SatelliteBroker
2
3 # Register brokering services (e.g., Sinergise Sentinel-Hub)
4 broker = SatelliteBroker(
5     esa_scihub=('http://services.eocloud.sentinel-hub.com/v1/fis/*apikey*')
6 )
7
8 # Extract statistical information based on given geometry (geom_wkt), temporal
range, data provider and dataset
9 data = broker.extract(geom_wkt, time='2015-08-01/2017-10-01',
                         provider='sentinelhub', dataset='EVI')
```

Listing 5.3: Python-based source code for conducting data extraction for Sentinel.

5.3 Unified EO Time-series Data Structure and Analysis

The exploration of geospatial time-series data is the basis for generating information and knowledge. Geospatial software and analysis tools can be used to extract information from time-series data. As such, users need to be able to work with time-series data in a programming language and geospatial tool of their choice. As much EO data is provided in different data formats (e.g., HDF, GeoTIFF, netCDF, and JPEG2000) and structures (see Figure 2.5), a conversion of these is often necessary to enable users to analyze this data. In addition, geospatial software and analysis tools handle raster time-series data differently. Thus, time-series data needs to be standardized and harmonized in format and structure as well as adjusted for the use in different processing and analysis tools.

Several questions are discussed in this section in order to standardize and harmonize the spatial and temporal handling of spatial time-series data:

1. How to simplify working with spatial time-series data for different users?
2. How to standardize the analysis of time-series data in automated workflows?
3. What kinds of time-series data output formats are useful?

Based on the review of raster time-series processing tools (Section 4.3), a specification has been drafted to standardize the management and handling of EO time-series data (Subsection 5.3.1). This uniform data structure is linked to the commonly used data processing tools described in Subsection 5.3.2. Further aspects of standardizing analysis tools are investigated (Subsection 5.3.3). Finally, the standardized and automated data processing and data analysis software are described (Subsection 5.3.4).

5.3.1 Common EO time-series data structure

Common data structure and data formats are necessary to provide data that can be directly used for analysis (“analysis-ready data”). Various file formats exist for use as a common data format however, in most cases, these are different in many user communities. In addition, new file formats, such as GeoPackage (Yutzler 2017), which focuses on providing a single format for both raster and vector data, should first further be explored in order to be used as a common format. In general, what is more important is the kinds of file formats users are familiar with, for example, using GeoTIFF for raster data and Shapefile for vector data. Although the definition of a common file format is important, in general most of available file formats can be read by the *GDAL* library, which is integrated in much geospatial software. More relevant—especially for time-series data—is the structure of the data due to the addition of the temporal dimension. In addition to file storage, database storages (e.g., PostGIS or SciDB) can be useful, though all users need to have knowledge of the specific database, which is often not the case. The structure of

time-series data needs to fulfill requirements from both point of views—the users' (easy to handle) and the services' (optimized for both publishing and analysis).

Although many applications, such as Open Data Cube and Rasdaman, have other internal data structures and formats, it is relevant to bridge the gap between accessing data from the data archives and being able to analyze data within software applications and programming languages. As such, it is important to define a uniform data structure, which can easily be used or exported to other data structures with minimal effort.

5.3.1.1 Spatial time-series data management

Observations extracted for an area of interest consist of multiple observations per date and can either be statistically reduced to values, such as the mean, minimum, maximum, and standard deviation, or retained as geo-referenced raster objects. For data management, both raster files and raster databases can be used. As explored in Section 4.3, the common approach with geospatial tools is to use raster files in combination with date extraction from their filenames or from a separate text file. Although the use of raster databases, such as SciDB, Rasdaman, or PostGIS Raster, can be considered useful, as different processing steps can be conducted while the data is being accessed, access to this database needs to be implemented for each geospatial tool. Rather than using raster databases, individual files, consisting of an individual raster file for each acquisition date, are proposed for storing data. The filename itself consists of a unique identifier for the dataset combined with the date.

Figure 5.21 shows an example listing for a vegetation index dataset with final GeoTIFF files. Rather than using one file per date, processing and analysis tools often work with a single raster file that includes one band per date. This can be achieved by using the virtual raster table (VRT) format from *GDAL*: VRT[36] is a lightweight XML-based format for composing geospatial data, which can be read by the *GDAL* library and allows integrating geospatial calculations to be applied on data access. A VRT file acts in the same way as a geospatial data file, such as GeoTIFF, HDF, ENVI, and the like. The VRT format can be used to restructure existing geospatial data without the need to process and store these additional files. For the common raster time-series data format, a VRT-based text file refers to each GeoTIFF file as individual band in the new geospatial dataset (see the example file "timeseries.vrt" in Listing 5.4). This file, which is based on raster time-series data, can be loaded into R as *RasterBrick* or *RasterStack* (Listing 5.5). Many analysis algorithms, such as Greenbrown and BFAST for vegetation analyses, that have been developed in R make use of this file format. Although there is no temporal information

[36] https://www.gdal.org/gdal_vrttut.html

stored in the VRT time-series data file, the link between the band number and the date needs to be managed externally. Therefore, a CSV file is provided as part of the common data structure, linking date, filename, and band number. This can be used in conjunction with the VRT file.

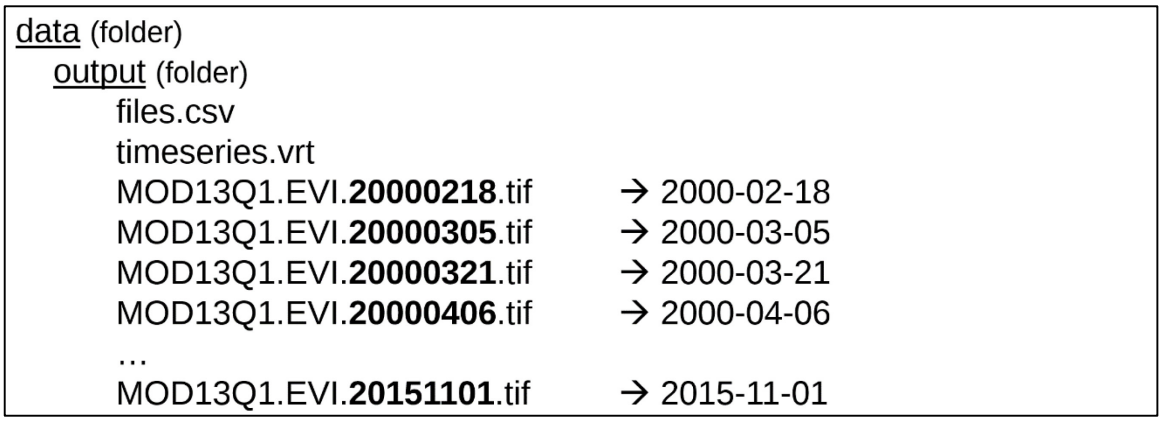


Figure 5.21: Time-series observations extracted for an area of interest in the final GeoTIFF raster file format (.tif from the output folder). Dates can be extracted either from the filenames, as shown in this figure, or from the text CSV file (files.csv).

```
01 <VRTDataset rasterXSize="29" rasterYSize="22">
02    <SRS>...</SRS>
03    <GeoTransform>...</GeoTransform>
04    <VRTRasterBand dataType="Int32" band="1">
05      <NoDataValue>-3000</NoDataValue>
06      <ComplexSource>
07        <SourceFilename>MOD13Q1.EVI.20000218.tif</SourceFilename>
08        <SourceBand>1</SourceBand>
09        <SourceProperties RasterXSize="29" RasterYSize="22" DataType="Int32"
             BlockXSize="29" BlockYSize="22" />
10        <SrcRect xOff="0" yOff="0" xSize="29" ySize="22" />
11        <DstRect xOff="0" yOff="0" xSize="29" ySize="22" />
12        <NODATA>-3000</NODATA>
13      </ComplexSource>
14    </VRTRasterBand>
15    <VRTRasterBand dataType="Int32" band="2">
16      <NoDataValue>-3000</NoDataValue>
17      <ComplexSource>
18        <SourceFilename>MOD13Q1.EVI.20000305.tif</SourceFilename>
19        <SourceBand>1</SourceBand>
20        <SourceProperties RasterXSize="29" RasterYSize="22" DataType="Int32"
             BlockXSize="29" BlockYSize="22" />
21        <SrcRect xOff="0" yOff="0" xSize="29" ySize="22" />
22        <DstRect xOff="0" yOff="0" xSize="29" ySize="22" />
23        <NODATA>-3000</NODATA>
24      </ComplexSource>
25    </VRTRasterBand>
26 </VRTDataset>
```

Listing 5.4: Example VRT files for two bands referencing external GeoTIFF files.

```
1 class       : RasterBrick
2 dimensions  : 22, 29, 638, 393  (nrow, ncol, ncell, nlayers)
3 resolution  : 231.6564, 231.6564  (x, y)
4 extent      : 905776.4, 912494.4, 5982062, 5987159  (xmin, xmax, ymin, ymax)
5 coord. ref. : +proj=sinu +lon_0=0 +x_0=0 +y_0=0 +a=6371007.181
                    +b=6371007.181 +units=m +no_defs
6 data source : /data3/pywps/5bc4e83e2f95e/data/output/timeseries.vrt
```

Listing 5.5: RasterBrick output in R using the VRT dataset containing each date as a band.

5.3.1.2 Data processing information

Additional information about the raster dataset is often necessary for further data processing. This includes information about temporal coverage and internal data settings (e.g., no data value and scale factor). The common data structure stores this information in a separate text file. This allows immediate data processing and analysis of the data without having to calculate this information again. For example, with the *ts* function within R, several parameters, such as start year, start offset, and temporal frequency, are needed to conduct analysis (e.g., as used within Greenbrown and BFAST). The following information has been extracted automatically from the data ingestion process and is stored within a text file as part of the common data structure (example values from a 16-day MODIS NDVI dataset are given in brackets):

- **Temporal information**
  - Start year and start offset as number of items [2000, 4]
  - Frequency—the number of items per year for regular time-series [23]
  - Number of years [14]
  - Number of dates [319]

- **Data information**
  - Projection [SR-ORG:6842]
  - No data value [-3000]
  - Scale factor [0.0001]
  - Number of pixels in X dimension [13]
  - Number of pixels in Y dimension [17]

5.3.2 Specifications for data processing tools

In addition to the common EO time-series data structure, data processing tools often use other data structures and formats. To simplify work with EO time-series data in various programming languages (Subsection 5.3.2.1) and geospatial tools (Subsection 5.3.2.2), additional data formats are provided as part of the common EO data structure. These can be used to simplify the ingestion of EO data into GIS software, databases, or data cubes. This subsection aims at bridging the gap between the data provided within this data structure and linked processing applications (“application-ready data”).

5.3.2.1 Programming languages

To ease the handling of the common data structure and format in programming languages, specific data formats can be stored for R and Python. In addition, web-based development

environments (e.g., Jupyter Notebooks) can be provided automatically. These are described in the following paragraphs.

R workspace

Users working with the statistical language R can be provided with a ready-to-use workspace that already includes data as time-series objects (for a single pixel) or raster objects based on a single multi-band file or on multiple files (for multiple pixels). Both can be created during an active session in R and saved as a workspace file. This file can then be loaded in a new session and the data directly used in the analysis. Using the *rpy2*[37] Python library, it is possible to open an R session in Python. This library is used to load the data, as described in Subsection 5.3.1, and save the content of the session to an R workspace.

Python xarray

The recommended way to store xarray data structures is by using the netCDF format.[38] Although xarray data structures based on the common data structure can be created with few lines of Python code (see Subsection 4.3.1), the raster time-series data can additionally be exported to the netCDF format. This automatically integrates the temporal dimension into the netCDF format and stores all data previously managed in multiple GeoTIFF files in one netCDF file. Without any further processing steps, this file format can be loaded with the xarray Python library.

Jupyter Notebooks (Python, R)

Interactive Jupyter Notebooks provides a web-based development environment for several programming languages. For Python and R, exporting the common data structure allows users to work directly with the data automatically preloaded in the Jupyter Notebook provided to the user. Using the *Jupyter Notebook Format* (nbformat)[39] Python library, preconfigured notebooks, in which access to raster time-series data has already been implemented, can be created automatically and provided as a Jupyter Notebook file. An example screenshot of such a Jupyter Notebook is shown in Figure 5.22.

[37] https://rpy2.readthedocs.io/
[38] http://xarray.pydata.org/en/stable/io.html#netcdf
[39] https://github.com/jupyter/nbformat

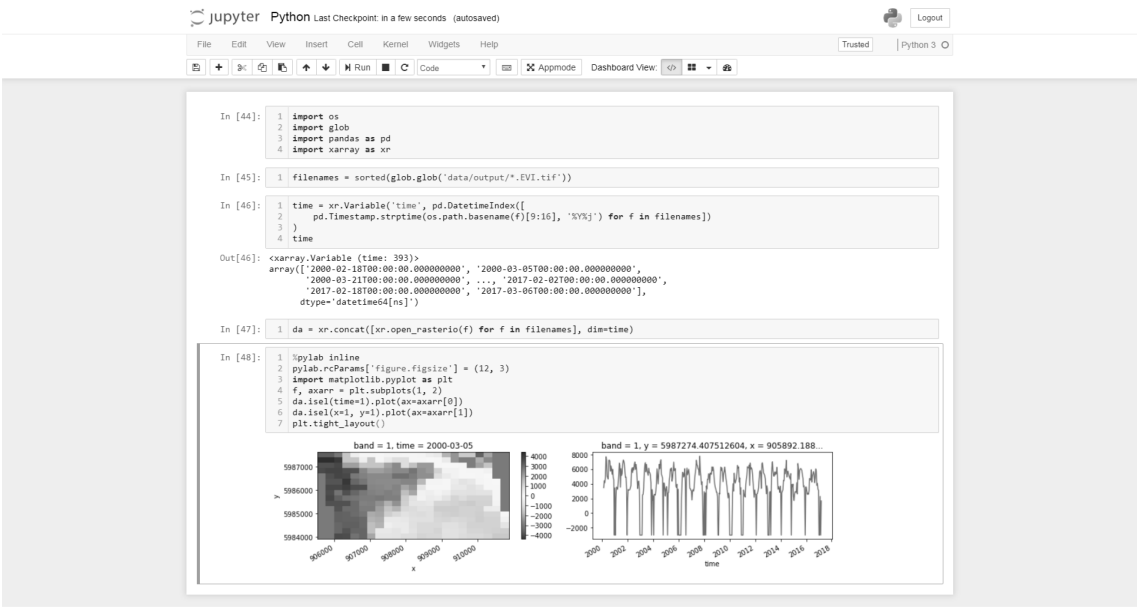


Figure 5.22: Screenshot of a Jupyter Notebook with preloaded time-series data in Python.

5.3.2.2 Geospatial tools

In addition to programming languages, geospatial tools can be used to further process and analyze EO time-series data. A connection to such applications allows a direct transfer of data to other tools and subsequent processing by the users. Geospatial processing tools for time-series data processing used and investigated in this thesis are described in the following paragraphs.

GRASS GIS location and mapset

A new GRASS GIS location and mapset can be created automatically on the server already containing the individual geospatial files from the raster time-series dataset and already registered in a space-time dataset. Using the GRASS Python API, the location and mapset can be created and datasets can be inserted without starting GRASS as application. As GRASS GIS stores its locations and mapsets to a file-based structure in the path of the location, this can be also provided as a download on a web server. As an alternative, users can use Jupyter Notebooks to access this mapset and work with the GRASS GIS Python API on the server.

Open Data Cube

Registration and ingestion of raster time-series data into the Open Data Cube software are based upon a set of supported dataset types (e.g., MODIS EVI, MODIS NDVI, and Landsat). Specific configuration files describing the dataset type and their content and preprocessing scripts for creating metadata files need to be available. To publish datasets into an instance of the Open Data Cube, access to the PostgreSQL database and the data folder is required. If those requirements are met, the following steps can be conducted automatically (see also Listing 4.8):

1. Registration of dataset type (if not already done).
2. Creation of metadata files for each of the geospatial data files (one per date).
3. Registration of metadata files.
4. Creation of a metadata file for the specific collection to be ingested.
5. Ingestion of the dataset based on the previously created collection metadata file.

Following successful ingestion into the data cube instance, the dataset can be loaded using the Python API from the *datacube* library (Listing 5.6).

```
# load library and create datacube connection
import datacube
dc = datacube.Datacube(app = 'my_app', config = '/datacube/.datacube.conf')

# load dataset from datacube
landsat_dataset = dc.load(latitude = (37.04, 37.20),
                          longitude = (-4.89, -4.63),
                          platform = "LANDSAT_8",
                          time = ('2017-01-01', '2017-03-01'),
                          product = "ls8_lasrc_fuente_de_piedra_example",
                          measurements = ['red', 'green', 'blue', 'nir',
                                          'swir1', 'swir2', 'pixel_qa'])

# run water detection algorithm for the first scene in the time-series
first_scene = landsat_dataset.isel(time=0)
water_classification = wofs_classify(first_scene, mosaic = True)

# plot water detection result
water_classification.wofs.plot()
```

Listing 5.6: The ingested data can be directly loaded using the *datacube* Python library.

Rasdaman

Inserting data into Rasdaman software can be achieved using the recipe JSON configuration file and the *wcts_import* Python script. To include the temporal information of the time-series, a regular expression needs to be provided to extract the date and time from the filename of each geospatial data file. As an export, the JSON configuration file can be written automatically when the URL for the web service endpoint of the Rasdaman instance is provided. Users can download the dataset directory and the Rasdaman recipe JSON file and ingest data into their local Rasdaman server.

5.3.3 Specifications for data analysis

Analysis tools can be used to derive further information, such as breakpoints, trends, and phenological information, from vegetation time-series data. In general, all analysis tools follow the same principles: They are based on data inputs, process data, and provide resulting outputs. To link analysis tools to EO time-series data, each of these steps need to be investigated for each analysis tool. In the following subsections, the input data formats (Subsection 5.3.3.1) and the output postprocessing (Subsection 5.3.3.3) are described for the time-series analysis tools BFAST, Greenbrown, and TIMESAT, which are used for vegetation time-series analysis (see Subsection 2.1.2). Subsection 5.3.3.2

describes additional aspects of the execution of algorithms, such as parallel computing, logging, and documentation.

5.3.3.1 Preprocessing and inputs data formats

The input data of a process often needs to be adjusted because the data retrieval component stores data in a different format. Especially for time-series data, it is often necessary to preprocess the original data to match the format and structure specified by the analysis tool. A raster time-series object is provided in R, which uses a multi-band file. Therefore, algorithms developed in R should make use of this internal raster time-series format, as occurs with BFAST and Greenbrown. Other tools, such as the TIMESAT command-line executable, need preprocessed data in specific file formats. In such cases, a connector is necessary to establish the link between the common EO time-series data structure and the input data. In the best case scenario, the algorithm can directly use the resulting outputs from the common data structure or a derived format, as described in the previous sections.

Table 5.5 lists data inputs for the algorithms BFAST, Greenbrown, and TIMESAT. Whereas BFAST and Greenbrown, both developed in R, use the standardized time-series methods of R, a special file format is necessary for TIMESAT. Therefore, preprocessing of the data input needs to occur only for TIMESAT. A Python class is written as a wrapper for TIMESAT to overcome this issue.

Table 5.5: Data inputs for the algorithms BFAST, Greenbrown, and TIMESAT divided into execution for single and spatial time-series data.

Analysis tool	Description of data input
BFAST	Single: univariate time-series (vector time-series, start, frequency) Spatial: calc-function with multi-layer raster object using *ts* function
Greenbrown	Single: univariate time-series (vector time-series, start, frequency) Spatial: multi-layer raster object of class brick, start, frequency
TIMESAT	Single: text data file Spatial: headerless binary format and text reference file

5.3.3.2 Algorithm execution

The execution of an algorithm can be accompanied by parallel computing, process logging, and documentation. This allows optimization of the execution of the analysis tool and provides advantages to users, such as performance increase and reproducible analysis. Users need to know what has been done with the input data, which enables the reproducing of the results of the analysis. Thus, all processing steps need to be documented, including information about the software used for processing, the individual commands with parameters, the scripts executed for data access or data analysis, and the parameters used for the analysis tools.

Parallel computing

Some algorithms, such as BFAST and TIMESAT, support high-performance and parallel computing. With the BFAST algorithm, the high-performance option can be set to “foreach” for the breakpoint detection method. The *foreach* package in R provides a looping construct for executing code that repeatedly supports parallel computing (Calaway 2017). This allows the running of operations on multiple processors and cores or on multiple nodes of a cluster. As the BFAST algorithm is a pixel-based approach, another option is the parallel execution of individual pixels. This approach is used by the TIMESAT software. If multiple processors are activated, the software splits the spatial input data into the number of multiple processors defined by the user. Each of the parts will be executed in parallel; thereafter the results are merged in the final output.

Process logging

Returning responses to the user while the processing task runs is an important feature of an algorithm that enable users to follow up on the progress of the execution of the process. Although responses from the algorithm itself depends to a great extent on the structural workflow of the algorithm, a few recommendations can be considered:

1) Process the fragmentation into steps, which can include the current step number and the full number of steps within the response.

2) Loops within the process can be integrated into the response by including the current index of the loop (e.g., for pixel-based analysis within a raster).

3) Responses of command-line tools should be directed to the standard output. This enables access to response messages, which can be redirected to the user.

Logging packages exist in many programming languages, including Python and R. Such packages allow the printing of log messages of different levels (e.g., warning, info, and debug) into single or multiple output channels (e.g., files and standard output).

Process documentation

In contrast to process logging, from a technical point of view process documentation contains the individual processing steps that have been conducted. The documentation enables users to reproduce the steps undertaken in the processing workflow. When command-line tools are used (e.g., to pre- or postprocess geospatial data), the exact command, including the parameters can be saved in a documentation text file. Within programming languages, the major functions should be included or otherwise exported as script file (e.g., bfast.R or timesat.py) and stored alongside the documentation file. This allows reproduction and debugging of the processing workflow.

Listing 5.7 shows an example (some parts have been truncated) for the processing of a single date containing several MODIS tiles and the extraction of the NDVI dataset for a user-defined area of interest. After the data download (Lines 4–10), the NDVI band is extracted (Lines 14–15), and thereafter the individual tiles are merged (Line 18), projected (Line 18), clipped to the area of interest (Line 24), and finally compressed (Line 27). Any of these steps can be performed locally to reproduce or debug the processing workflow.

```
01 #processing MOD13A3 for study area: Forest Dragon
02
03 #download files
04 wget -c ftp://…/MODIS_Composites/MOLT/MOD13A3.005/2005.01.01/*.h28v04*.hdf*
05 wget -c ftp://…/MODIS_Composites/MOLT/MOD13A3.005/2005.01.01/*.h25v03*.hdf*
[... truncated ...]
11
12 #dataset extraction
13 #processing MOD_Grid_monthly_1km_VI:1 km monthly NDVI
14 gdal_translate -a_nodata -3000 -of GTiff -a_srs "+proj=sinu +lon_0=0 +x_0=0
       +y_0=0 +a=6371007.181 +b=6371007.181 +units=m +no_defs"
       HDF4_EOS:EOS_GRID:"MOD13A3.A2005001.h26v04.005.2007355120626.hdf":
       "MOD_Grid_monthly_1km_VI:1 km monthly NDVI"
       MOD13A3.A2005001.h26v04.005.2007355120626.NDVI.tif
15 [... gdal_translate is being executed for each tile ...]
16
17 #merge for selected study area
18 gdal_merge.py -o merge.tif *.h28v04.*.NDVI.tif *.h25v03.*.NDVI.tif
19
20 #set projection
21 gdalwarp -t_srs EPSG:4326 merge.tif MOD13A3.A2005001.NDVI.6.tif
22
23 #clip data to selected study area
24 gdalwarp -of GTiff -cutline PG:"dbname=sibessc" -csql "select
       ST_SetSRID(geom, 4326) from userdata.study_areas WHERE uid='6'" -cblend
       0 -crop_to_cutline MOD13A3.A2005001.NDVI.6.tif
       MOD13A3.A2005001.NDVI.6.clipped.tif
25
26 #compress data
27  gdal_translate  -co  COMPRESS=PACKBITS  MOD13A3.A2005001.NDVI.6.clipped.tif
       MOD13A3.A2005001.NDVI.6.clipped.compressed.tif
```

Listing 5.7: Processing log file with download and processing steps.

5.3.3.3 Postprocessing and output data formats

The eventual datasets can be in a variety of data formats, as defined by the algorithm. They can contain figures, charts, text files, or geospatial data. In most cases, it is necessary to process the outputs to achieve better readability and easier evaluation of the results. The BFAST and Greenbrown libraries do not provide any output files—these need to be exported from the programming language R. Thus, exports of figures and geospatial raster data as well as an optimized vector dataset need to be integrated and provided to users. Other algorithms, such as TIMESAT, deliver functions or command-line tools to export the results to geospatial data. Both of these need to be integrated in the postprocessing to generate user-aligned output formats (e.g., GeoTIFF files or prepared figures). In addition, the postprocessing of the results—including the generation of a vector pixel Shapefile and the provision of OGC WMS and WFS web services—allows direct

exploration of the results by the user. For each analysis tool used within this thesis, Table 5.6 compares the resulting regular output files with the postprocessed output files generated by the middleware and provided to the user. These additional output formats are described in the following paragraphs.

Results CSV

The most important numbers of the results of the analysis (e.g., fitted time-series, trend lines) are provided in a dedicated CSV spreadsheet file, which can be used directly either in any spreadsheet software or with web-based interactive charting tools.

OGC web services

To allow direct visualization of the results in an interactive map, geospatial output data needs to be made available using the OGC WMS specification for raster data and the OGC WFS specification for vector data. Therefore, visualization styles need to be prepared during the postprocessing of the execution of the algorithm for any geospatial output of either algorithm. To provide these OGC web services, an OGC-compliant web server needs to provide a service instance for each execution of the analysis.

Figures and charts

The provision of figures and charts allows immediate visualization of the results of the analysis. Depending on the algorithm and the input time-series data (e.g., pixel or raster based), different figures can be generated. Spatial map outputs can be shown for raster-based analyses (Figure 5.23) and individual charts for single pixel analyses, which can be used to provide a quick look at the results of the analysis. In addition to the figure generated, the values for the results of an individual time-series analysis can be stored in a CSV spreadsheet file (e.g., the “Results CSV” file for BFAST and Greenbrown or the “Data CSV” file for Greenbrown).

Vector Shapefile

Raster output data from time-series analysis contains per-pixel information. The exploration of individual raster pixels in web-based systems is best accomplished using vector outputs, as individual pixels—transformed to rectangle vector features—can be interactively selected in web-mapping libraries. Thus, geospatial raster outputs can be converted to a vectorized data format, which can be made available with standard-compliant web services (e.g., OGC Web Feature Service). In this vector file, each of the pixels is represented by an individual polygon. A column with the value for each band is integrated in the geospatial output data (Figure 5.24).

Table 5.6: Scientific time-series analysis tools (BFAST, Greenbrown, and TIMESAT) and the resulting regular output files, as defined by the algorithm, and the postprocessed output files, as specified within this thesis.

Analysis tool	Resulting (regular) output files	Postprocessed output files
BFAST—Single (R)	• "BFAST" object—List with various components (e.g., fitted trend and seasonal, deseasonalized, noise or remainder, breakpoints), including prepared plot figures	• PNG figure • Results CSV (time-series with original data, fitted season, and trend) • Dates of trend and seasonality breaks as text output
BFAST—Spatial (R)	• "BFAST" object (see above) for each pixel; converted to raster objects in the postprocessing steps	• PNG figure • GeoTIFF file (multi-band) • Vector Shapefile • OGC WMS/WFS
Greenbrown—TrendRaster (R)	• RasterBrick with different trend and breakpoint statistics (e.g., date of trend breakpoints; slope, p-value, length of trend per segment) • Prepared plot figures based on RasterBrick	• PNG figure • GeoTIFF file (multi-band) • Vector Shapefile • OGC WMS/WFS
Greenbrown—Trend (R)	• Trend class, including prepared plot figures	• PNG figure • Results CSV (date, slope, p-value per segment) • Data CSV (trend-line per segment)
TIMESAT—Spatial (Command line)	• Fitted time-series data in binary headerless format (ENVI HDR) • Phenological values per season and parameter in a geospatial headerless format	• PNG figure • GeoTIFF file (multi-band as season) for each parameter • Vector Shapefile • OGC WMS/WFS
TIMESAT—Single (Command line)	• Fitted time-series text file • Phenological values per season within binary file	• PNG figure • Seasonality CSV (Phenological values per season) • Data CSV (time-series with original, fitted, start of season, and end of season data)

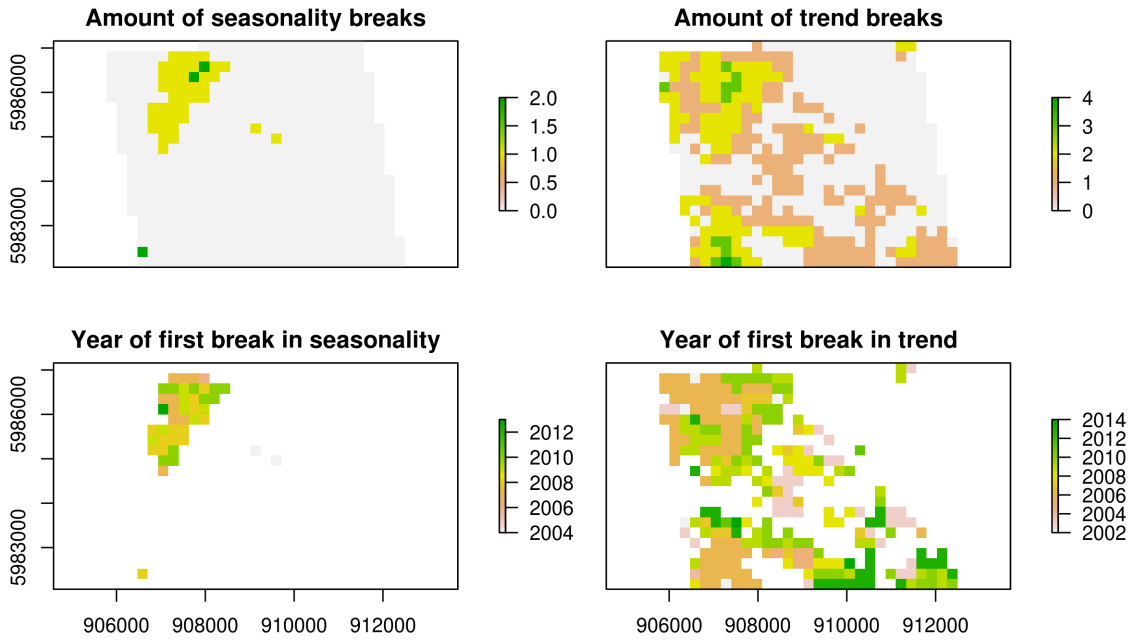


Figure 5.23: Graphic plots from geospatial data layers showing the outputs resulting from the breakpoint detection algorithm BFAST.

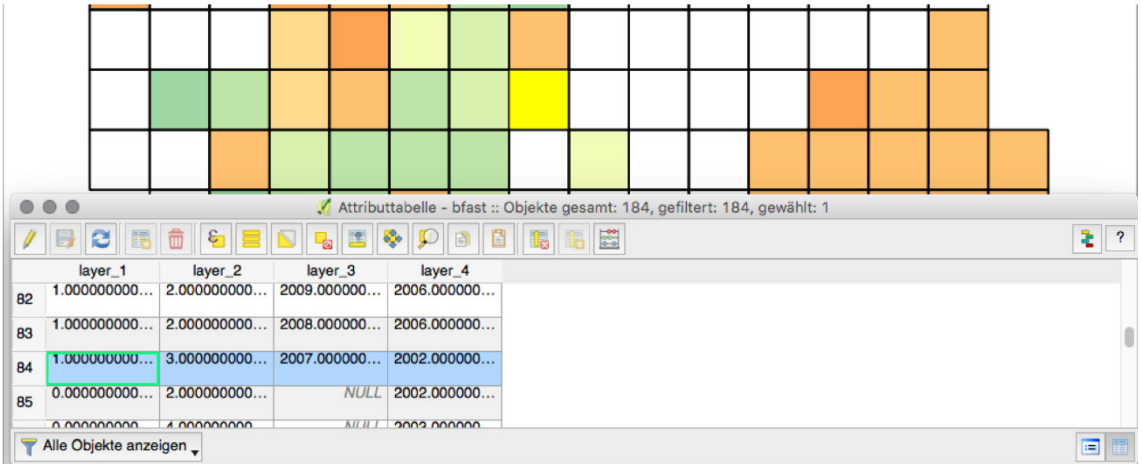


Figure 5.24: Visualization of the postprocessed vector Shapefile based on BFAST results.

5.3.4 Implementation: EO time-series data processing and analysis

The pyEOM Python library is extended with tools for EO time-series data processing and time-series data analysis. Tools for data access (Subsection 5.3.4.1) include data transformation from downloaded satellite scenes to the uniform time-series data structure for Landsat, Sentinel, and MODIS data based on different data providers. Predefined information for each dataset stored within the library is used to generate the metadata (Subsection 5.3.4.2). Time-series data analysis tools, which can be used with the integrated datasets, are registered and managed using the pyEOM library (Subsection 5.3.4.3).

5.3.4.1 Data access software

The data access component of the pyEOM library includes EO time-series data integration and management, as described in the following paragraphs.

EO time-series data integration

The *DataIntegration* class of the pyEOM library handles the processing and integration of EO time-series data into the uniform EO time-series data structure. For each individual satellite product (e.g., MODIS Level–3, Landsat, and Sentinel), various data structures and formats are considered when converting to the uniform data structure. Whereas MODIS Level–3 products contain individual bands in a single HDF file, Landsat and Sentinel data provide each band in an individual file. As such, for each of the satellite products, different processing steps are implemented. As an alternative, Google Earth Engine can be used to extract data. Listings 5.8 and 5.9 show example integration tasks for both the original data provider and Google Earth Engine using the pyEOM library.

MODIS Level–3 products on board the Terra and Aqua satellites are provided in the HDF-EOS data format, which can be processed using the *GDAL* library. Depending on the spatial resolution of the product, the data is either provided as a global HDF file in a WGS84 projection or in a fixed tiling schema in a sinusoidal projection. In the latter case, different tiles need to be merged if the area of interest contains multiple tiles. Thereafter, the relevant bands need to be extracted (e.g., NDVI, EVI, and quality bands have been considered for MODIS Vegetation Index data), clipped to the area of interest, have offset and scaling factors applied, and converted to the final output format (e.g., GeoTIFF).

Landsat data can either be downloaded as original satellite scenes or ordered using the USGS ESPA service, which already includes preprocessing steps, such as selection of bands, calculation of indices, clipping to the area of interest, and reprojection of data. The advantage in using the USGS ESPA ordering tool arises from the data itself as this can be automatically clipped to the area of interest. As such, all the scenes offer the same geospatial extent. For data downloads from both systems, each band is included in its own GeoTIFF file prepending the unique scene identifier in the filename.

Sentinel data can be downloaded as zipped archive files from different data providers (e.g., ESA/Copernicus Open Access Hub or national hubs). *GDAL* or *SNAP*[40] can be used to process the Sentinel data format and to convert it to the final output format (e.g., GeoTIFF). For Sentinel–1, the *pyroSAR*[41] Python library is used.

[40] http://step.esa.int/main/toolboxes/snap/
[41] https://github.com/johntruckenbrodt/pyroSAR

Google Earth Engine can be used in addition to the previously mentioned original data providers, as Landsat, Sentinel, as well as several MODIS Level–3 products are available. Especially for pixel-based extractions, the usage of Google Earth Engine reduces the amount of network transfer as only the resulting time-series data are transferred to the requesting application (Listing 5.9). Although the intention of Google Earth Engine is to conduct analyses within its system, it is possible to export data clipped to the area of interest, with offset and scaling factors applied, reprojected and converted to the final output format either to a Google Drive folder (the preferred location) or provided as download link (deprecated). Data exported to Google Drive can be further downloaded to the external application using Google Drive API requests.

```
from pyEOM import tasks
ingest = tasks.Ingestion({
    'dataset': 'MODIS/MOD13Q1',
    'geom': 'POLYGON((7.8 26.2, 7.4 25.6,8.5 25.0,9.0 25.8,7.8 26.2))',
    'start': '2001-01-01',
    'end': '2001-02-01',
    'qualityValue': '0;1',
    'qualityBand': 'PR',
    'publishPath': '/tmp/pyEOM',
    'format': 'HDF4Image',
    'EPSG': None,
    'resample': None,
    'source': 'LPDAAC',
    'userPwd': 'username:password'
})
output = ingest.start()
```

Listing 5.8: Integration of MODIS Vegetation Index data for an area of interest with further processing parameters applied.

```
from pyEOM.datasets import Landsat
source = Landsat.GEE({'dataset': 'LANDSAT/LC8_L1T', 'geom': 'POINT(11 51)'})
output = source.ingest()

from pyEOM.datasets import MODIS
source = MODIS.GEE({'dataset': 'MODIS/MOD13Q1', 'geom': 'POINT(11 51)'})
output = source.ingest()
```

Listing 5.9: Accessing pixel-based Landsat-8 and MODIS Vegetation index data for a given point of interest from Google Earth Engine.

EO time-series data management

The *DataManagement* class of the pyEOM library handles the uniform EO time-series data structure and provides export functions to the user- and application-driven output formats described in Subsection 5.3.2. This class understands the unique data structure defined in Subsection 5.3.1 and provides access to all the relevant metadata and data. The class is separated into single and spatial time-series data, as the handling and management of these are different. An instance of this class can be created using the path to the data directory, which contains the unique data structure. Further methods (e.g., the calculation of statistics) can be used thereafter (Listing 5.10).

```
01 from pyEOM import DataManagement
02
03 # Instance DataManagement class for a specific directory
04 data = DataManagement('/data3/pywps/23454234234')
09
10 # Retrieve statistics from dataset
11 stats = data.stats()
12
13 # Plot data
14 data.plot()
```

Listing 5.10: Using the DataManagement class of pyEOM based on the common data structure.

5.3.4.2 Raster time-series metadata

Metadata records list information about the data, visualization and access services, as well as the available time positions and time intervals (Table 5.7). In addition to general metadata, such as the title, abstract, keywords, and lineage, further metadata is used to describe the time-series data in detail. This includes information that is necessary for processing the data, such as scale factors, resolution, or no data value. To link to OGC-compliant services for visualization and downloading, the service endpoints (e.g., WCS URL), service type (e.g., WCS Protocol), and name of the layer (e.g., WCS Name) are provided in the metadata record. Based on this, a client can build requests for accessing the data using the standard-compliant OGC web services. Metadata records are described following the ISO 19115 specification (ISO 2003). A client can retrieve metadata records based on a specific identifier or a search result, parse the information, and visualize or download the data using the services provided. With the metadata, the client knows which time positions are available and which services (e.g., WMS, WCS, or any other HTTP link) can be used, in accordance with needs of the user. Client applications can further distinguish between time-series raster data as physical measurements or as classifications using the “Content Type” property. This distinction is important for aspects, such as providing the correct analysis processes, which differ for classification results (e.g., burned area), as opposed to continuous data, such as land surface temperature, vegetation indices, and snow cover. Table 5.7 shows a detailed metadata record as an example of a raster time-series dataset from the monthly MODIS Land Surface Temperature product.

Table 5.7: Metadata schema for a raster time-series dataset with example values from the monthly MODIS Land Surface Temperature product.

General metadata	
File Identifier	MODIS_MOD11_C3_LST_Day_Series
Title	Monthly Land Surface Temperature from MODIS Terra
Abstract	Time-series of monthly Terra MODIS daytime land surface temperature in Kelvin at 0.05 degrees spatial resolution. To retrieve actual values in Kelvin, a scale factor of 0.02 has to be applied. The no-data value is encoded as 0. Original MODIS data retrieved from NASA LPDAAC.
Keywords	MODIS, Terra, Temperature, Monthly, Series, Daytime
Lineage	MODIS HDF Level–2 product was converted to GeoTIFF using *gdal_translate* (Version 1.9)
Data information	
Description	Land Surface Temperature
Data Type	Raster
Content Type	Physical Measurement
SRS	EPSG:4326
BBOX	57.1301270 81.2734985 179.8292847 42.2901001
Columns	2,454
Rows	780
Resolution	0.05
Scale Factor	0.02
No Data Value	0
Time Begin	2000-03-01
Time End	2012-09-01
Time Interval	P1M
Dates	2000-03-01, 2000-04-01, …, 2012-08-01, 2012-09-01
Visualization and access services	
WMS URL	http://artemis.geogr.uni-jena.de/sibessc/modis
WMS Protocol	WebMapService:1.3.0:HTTP
WMS Description	MODIS Terra LST Day Monthly
WMS Name	mod11c3_lst_day
WCS URL	http://artemis.geogr.uni-jena.de/sibessc/modis
WCS Protocol	WebCoverageService:1.1.0:HTTP
WCS Description	MODIS Terra LST Day Monthly
WCS Name	mod11c3_lst_day

5.3.4.3 Time-series analysis software

The *DataAnalysis* class of the pyEOM library is an abstract class containing general methods for registering, managing, and executing analysis tools. This abstract class needs to be extended for each analysis tool. Examples are developed for the analysis tools BFAST, Greenbrown, and TIMESAT in the pyEOM library.

Management of analysis tools

The management of analysis tools involves the main relevant metadata and methods, which are relevant to conducting the analysis and to postprocessing the resulting outputs. Prior to executing an analysis, input and output data needs to be defined in the extended class of each analysis tool. In addition, the “run” method needs to be overwritten either using an own method that includes some processing steps (e.g., TIMESAT) or by linking to external scripts (e.g., R scripts for BFAST and Greenbrown). Output data can be automatically postprocessed: GeoTIFF files can be converted to vector data. Geospatial data in general can be provided as web services using the OGC WMS and WFS specifications, as described in Subsection 5.3.3.3.

Execution of analysis tools

The execution of the analysis tools BFAST, Greenbrown, and TIMESAT is integrated and connected to the data integration and management functions in the pyEOM library. As defined in Table 5.6, several post-processed output formats are created to provide user-aligned data formats (e.g., summarized statistics, a Pixel Shapefile, OGC web services, and plots). Listing 5.11 shows an example execution of the TIMESAT phenology tool (Lines 8–9) based on a previously integrated dataset (Line 5). Each of the analysis classes imported in Line 2 (i.e., TIMESAT, Greenbrown, and BFAST) provides a bridge between the analysis tool and pyEOM as a common time-series data handling and analysis execution framework.

```
01 from pyEOM import DataManagement
02 from pyEOM.analyses import TIMESAT, Greenbrown, BFAST
03
04 # DataManagement instance for a specific directory (unique identifier)
05 data = DataManagement('/data3/pywps/23454234234')
06
07 # Execute TIMESAT with default parameters
08 timesat_inst = TIMESAT(data=data, parameters={})
09 timesat_inst.run()
10
11 # Export results (e.g., MapServer config, GeoTIFF, Shapefile, PNG plots)
10 timesat_inst.export()
11
12 # Show result as plot
13 timesat_inst.plot()
```

Listing 5.11: Using TIMESAT analysis tool from the pyEOM library.

Chapter 6: Example Use Cases

Various web and mobile applications have been developed to demonstrate how the concepts and methods described in Chapter 5 can be used in applications and provided to users. Three projects are described, which share the main objective of providing user-aligned applications using a service-based middleware approach. However, each of the applications focuses on different aspects:

1. The Siberian Earth System Science Cluster (Section 6.1) focuses on EO data integration, standardized data distribution (downloading and visualization), time-series extraction, simple time-series analyses, and user management.

2. The Earth Observation Monitor (Section 6.2) focuses on EO data integration, common EO time-series data structure and formats, time-series extraction, enhanced time-series analyses, and user management.

3. The GEO-Wetlands Community Portal (Section 6.3) focuses on multi-source EO data discovery and access based on service brokering and additional web-based processing applications, such as Open Data Cube and R-Shiny.

All the use cases are based on the service-oriented middleware approach (Chapter 5.1). In addition, all make use of several findings from the unified EO time-series data structure and analysis (Chapter 5.3). The service brokering approach described in Chapter 5.2 is used in the GEO-Wetlands Community Portal.

6.1 The Siberian Earth System Science Cluster

The Siberian Earth System Science Cluster (SIB-ESS-C) was developed with the aim of providing operational tools for multi-source data access, analysis, and time-series monitoring in Siberia (Eberle et al. 2013). Data from remote sensing satellites, climate data from meteorological stations, and outcomes of research projects are stored in the SIB-ESS-C. All the geospatial data is described with standard-compliant metadata and provided within a standard-compliant metadata catalogue. The system comprises a metadata catalog that allows for data searching, as well as interoperable interfaces for data visualization, downloading, and processing. The advantage of representing different products within a single system is the integration of users' needs into web-based processing services. Concerning climate change and land monitoring, the SIB-ESS-C focuses on land-based information products. The objective of the middleware within the SIB-ESS-C is to build an operational web-based system in which data from different sources is provided and regularly updated. The middleware automatically collects data from external EO data archives to provide standard-compliant web services for data access and visualization. Datasets are then available for further visualization and analysis.

In relation to the user requirements for platforms and web technology, the following requirements are supported: multi-source EO and geospatial data; EO data visualization; data download; pixel extraction; time-series analysis; OGC standardization; metadata for geospatial data; user management; RESTful web services; and asynchronous web services. The following data formats are supported: OGC web services for data access, visualization, and data analysis; interactive charts for pixel time-series; and figures for time-series analysis results.

6.1.1 SIB-ESS-C Middleware

The middleware service provided by the SIB-ESS-C integrates EO time-series data from the NASA LPDAAC and the National Snow and Ice Data Center (NSIDC). In addition to MODIS Land products, climate data from meteorological stations is available from National Oceanic and Atmospheric Administration (NOAA) National Climatic Data Center (NCDC), which provides various datasets, such as hourly synoptic measurements (Integrated Surface Database, ISD; Lott et al., 2008), daily summaries from synoptic measurements (Global Surface Summary of the Day, GSOD; Lott, 2006), and daily measurements from different climate data networks (Global Historical Climatology Network, GHCN; Menne et al., 2012). The data sources provided for operational land monitoring in the SIB-ESS-C are listed in Table 6.1.

Table 6.1: Data sources integrated in the SIB-ESS-C middleware system.

Data	Provider	Available time ranges
MODIS Level–3 products from LPDAAC and NSIDC	NASA USGS	2000–2014
Global Surface Summary of the Day (GSOD)	NOAA	1929–2014 (depending on station)
Integrated Surface Database (ISD)	NOAA	1929–2014 (depending on station)
Global Historical Climatology Network Daily (GHCN-Daily)	NOAA	1832–2014 (depending on station)

Several data exploration tools have been made available in the SIB-ESS-C middleware and are provided as standard-compliant web services based on the OGC WPS specification, which are sourced from the SIB-ESS-C web portal:

- Time-series data extraction for a point or area of interest summarized with the mean, minimum, maximum, and standard deviation values.
- Automated calculation of time-series decomposition and BFAST for the time-series extracted (mean value).
- Climate station data plots with temporal aggregations (e.g., days, months, years, winter, spring, summer, fall, individual months).
- Kernel-density plots for the comparison of in-situ mean temperature station data and MODIS Land Surface Temperature time-series (1 km) data (Figure 6.1).

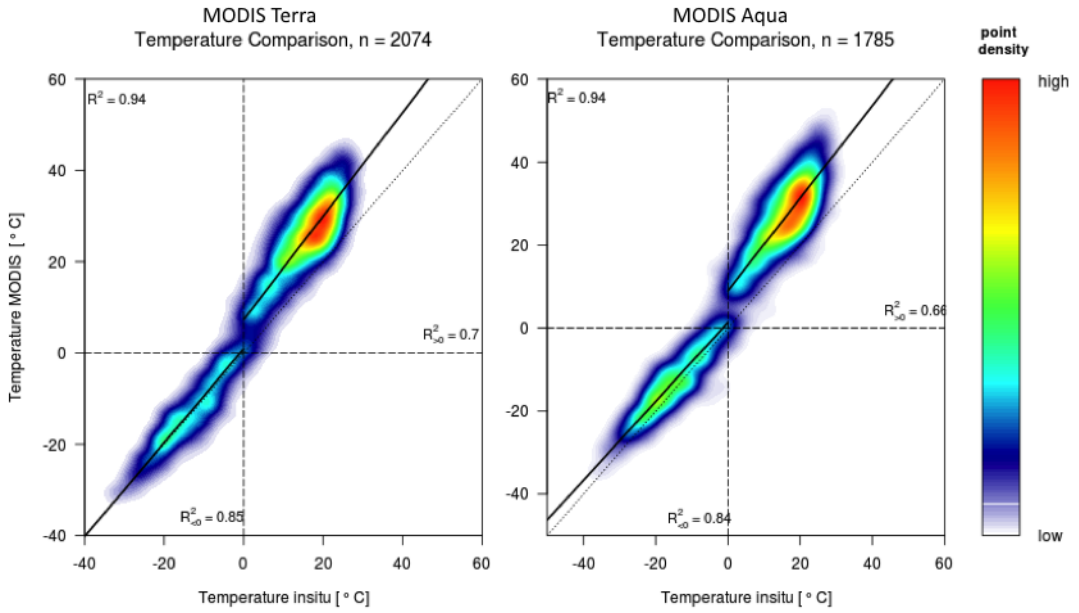


Figure 6.1: Comparison of MODIS Terra (left) and Aqua (right) LST estimates and air temperature records from an individual meteorological station (Aleiskaya). Each plot shows the complete available time period (2000/2002–2013).

The integrated datasets are made available via web services compliant with OGC specifications. This includes the CSW for metadata cataloguing and dataset search, data visualization and access for raster time-series data through WMS and WCS, and data access for climate station time-series data through the SOS. The time-series plotting service and the analysis tools are available as a WPS. In addition to providing visualization and download tools in the developed middleware services, the middleware also controls the data integration process, with each step logged in the system. This feature is integrated for on-demand processing. With the logging functions implemented in the web portal, the user is informed when the user-specific data integration is completed.

Open source tools were used to develop the middleware services. PostgreSQL with the PostGIS extension provides the database with the ability to store raster and vector data as well as other metadata. Data integration is achieved using Python scripting (e.g., to execute command line tools for data downloading and for raster time-series data processing). The scripting language R is used to plot the integrated time-series data. On the service level, MapServer (data visualization and downloading), istSOS (climate data provision), pycsw (metadata provision), and PyWPS (time-series plotting and analysis) are used to publish OGC-compliant web services. Drupal CMS is used for administration services to manage the areas of interest created and the datasets ingested by users. This backend system also provides the tools to handle the execution of external web services, the conversion from XML to JSON format, and to provide RESTful services for user registration and authentication.

6.1.2 Web portal

The middleware services are used in the SIB-ESS-C web portal. This web portal (Figure 6.2) provides functions that allow users to administer and manage the middleware services; it also allows easy access to the integrated datasets. Users are supposed to interact closely with the data to extract further information. Users can explore the data catalog, which contains data available within the middleware. The metadata catalog can be searched and the resulting records investigated. The data can then be visualized and downloaded. Open-source software was used to develop the web portal frontend using the jQuery library.[42] The interactive map viewer for visualizing the geospatial data is based on the OpenLayers library.[43]

[42] http://jquery.org
[43] http://openlayers.org

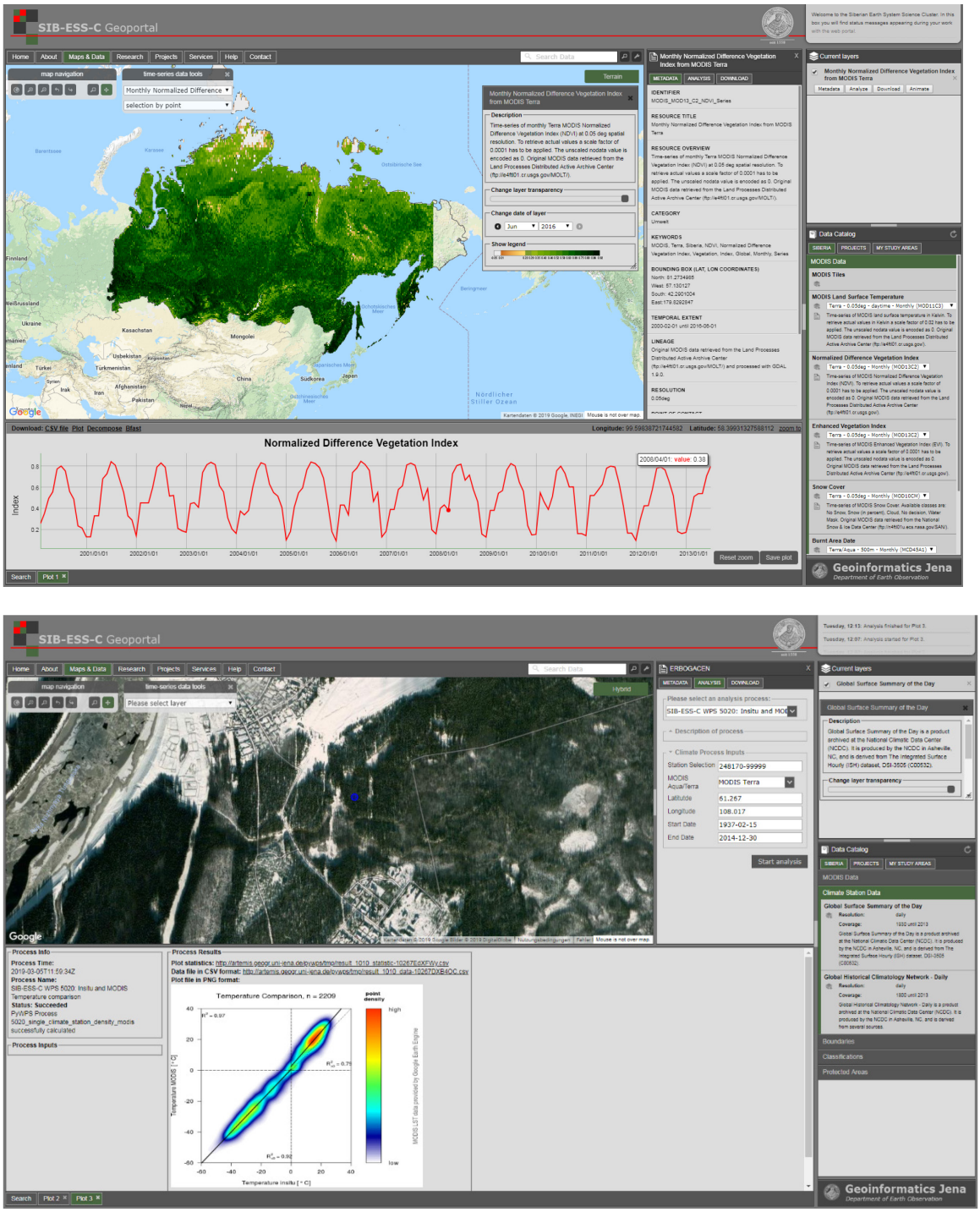


Figure 6.2: Screenshots of the Siberian Earth System Science Cluster Geoportal highlighting the visualization of a MODIS NDVI dataset (top), a time-series extraction with interactive charting (top), and a comparison plot between MODIS Land Surface Temperature and in-situ temperature measurements (bottom).

6.2 Earth Observation Monitor

The Earth Observation Monitor (EOM) is a successor project to the SIB-ESS-C that has the aim of ensuring access to and analysis tools for global spatial time-series data used for land monitoring on local scales. Scientists and stakeholders working in local areas must be able to perform time-series analyses without having to process any data on their own. The functions of the EOM are available using web services, allowing client applications to make use of these services. Based on the EOM middleware system (Subsection 6.2.1), three example clients were developed to show the possibilities of such a service-based infrastructure: The webEOM web portal (Subsection 6.2.2), the mobileEOM mobile application (Subsection 6.2.3) for time-series data access and analysis, and the MySeasons mobile application (Subsection 6.2.4) dedicated to vegetation phenology modeling. All these clients use the OGC WPS for data integration and analysis provided by the EOM middleware.

In relation to the user requirements for platforms and web technology, the following requirements are supported: multi-source EO and geospatial data; data downloads; pixel extraction; time-series analysis; support for various programming languages and analysis tools; visualization of geospatial time-series results; OGC standardization; user management; RESTful web services; asynchronous web services; service chaining; uniform specifications; and multiple output formats. The following data formats are supported: OGC web services for data access, visualization, and data analysis; interactive charts and figures for time-series data and analysis results; and summarized statistics for data and analysis results.

6.2.1 EOM Middleware

The EOM middleware system allows the integrating of global satellite and in-situ time-series data focusing on land monitoring. Table 6.2 lists the data available within the middleware. Vegetation Index (VI) and Land Surface Temperature (LST) products from the NASA MODIS sensor are available for integration into the middleware, based either on NASA data archive (LPDAAC) or Google Earth Engine. Although there is a limitation to the size for the area of interest that users can define, the coverage of the data is global. Globally available in-situ meteorological data from climate stations have also been connected to the middleware based on GSOD and GHCN data provided by the NOAA National Climatic Data Center. As a further development from the SIB-ESS-C, the data is downloaded and processed according to user demand. Thus, users can select their station and parameters of interest from these globally available climate stations.

Table 6.2: Data sources within the EOM middleware, including EO and in-situ time-series data.

Dataset	Type	Spatial res.	Temporal	Time range
NASA LPDAAC and Google Earth Engine				
MODIS VI	Raster	250 m–1 km	16-Day	2000–now
MODIS LST	Raster	1 km*–0.05 deg.	Daily, 8-Day	2000–now
NOAA NCDC				
GSOD	Sensor	–	Daily	1929–now
GHCN	Sensor	–	Daily	1832–now

* Only selected areas

Several geospatial tools have been integrated into the EOM middleware and are provided as standard-compliant web services using the OGC WPS specification:

- Single and spatial time-series data extraction for user-defined areas of interest, optional filtering and masking based on quality flags including the conversion of data into a common data format, and the calculation of time-series plots and decomposition analyses.
- Breakpoint analyses for EO vegetation time-series data using BFAST.
- Trend calculations for EO vegetation time-series data using Greenbrown.
- Phenological modeling for EO vegetation time-series data using TIMESAT.
- Random forest disturbance analyses for EO vegetation time-series data.
- Automated connection between EO time-series data and analysis tools based on the common time-series data format.
- Automated conversion of the outputs of the results of analysis to user-aligned data formats.
- Climate station data plots with temporal aggregations (e.g., days, months, years, winter, spring, summer, fall, and individual months).
- Kernel-density plots for the comparison of in-situ mean temperature station data and MODIS Land Surface Temperature time-series (1 km) data (see Figure 6.1).

The data integration and data analysis services are made available using the OGC WPS specification. As data integration and data analysis differ based on the geometry type (point extraction leading to single time-series data vs. polygon extraction leading to spatial time-series data), for all the tools, two processing services for single and spatial time-series data are provided. In addition to EO time-series data integration, a process is available for the in-situ climate station data integration. The resulting outputs from the analysis tools are available based on the OGC WMS for visualization and the OGC WFS for the raster-to-vector converted outputs.

Open source software has been used to build the EOM middleware and their web service infrastructure. In the backend, Drupal CMS is used for user registration and authentication, management of features created by registered users, as well as a proxy for external web services, and the conversion of XML responses from OGC web services to JSON. At the service level, MapServer is used for data visualization and access to provide OGC WMS and WFS. PyWPS is used for the processing services according to the OGC WPS specification. The analysis tools are provided through Python, R, and TIMESAT though all of them are run from the PyWPS Python process. With the Python library rpy2, a direct connection between Python and R sessions is used to undertake R-functions and exchange data between the programming languages. In Python, further libraries, such as GDAL, OGR, and Pandas, have been used to process data and generate plots.

6.2.2 webEOM

The focus of the web portal is to provide an easy-to-use client while making it possible to extract time-series data and execute time-series analysis functions. The webEOM map viewer (Figure 6.3) can be used to create the geometry of a study area. Based on this geometry, the system requests a list of available datasets, which are registered in the middleware. When conducting data integration, users can specify different parameters for the selected dataset, such as start and end dates, as well as filtering options. A processing directory for each integrated dataset is available to users, which contains any processed data for both the dataset and the analysis tools used. Time-series and decomposition plots, which are shown in the web portal, are generated automatically from the extracted time-series data. In addition to the images, CSV files containing statistics are available for download. Following data integration, users can select an analysis tool for execution, and individual parameters can be set for the selected analysis. The resulting data can either be visualized directly in the web portal or can be downloaded for further analysis. Spatial outputs can be interactively explored in the map viewer, and CSV files are plotted in an interactive chart window.

Several open-source libraries were used to develop the web portal: For the frontend user interface, the JavaScript library OpenLayers is used as a mapping library; dygraphs[44] is used to generate interactive charts; and jQuery provides standard JavaScript functions.

[44] http://dygraphs.com

Figure 6.3: Screenshots of the webEOM portal.

6.2.3 mobileEOM

During fieldwork, users cannot easily use web-based applications optimized for desktop computers. A mobile application can therefore foster the use of spatial time-series tools on mobile devices, which can be used more easily during fieldwork. With the mobileEOM application, users have access to a wide range of information, as the MODIS vegetation time-series data provides information about the environment from 2000 to the present.

The use of EO data with time-series analysis tools allows the identifying of breaks in the past and overall trends relating to the vegetation.

The mobile application for EOM was developed to provide access to time-series data and derived analyses on mobile devices. Using their current GPS location or a manually set position, users can extract vegetation time-series data, as well as view plots for data, trends, and breakpoint analysis directly on their mobile devices. An OGC WPS process was developed for the mobile application to integrate the existing services from the EOM middleware into a single web service call. Figure 6.4 shows screenshots of mobileEOM linked with a workflow chart of the OGC WPS process and interactions between them. This process uses the EOM WPS to extract the requested data from Google Earth Engine and plots the time-series and decomposition figures. In a second step, the time-series analysis services from EOM for breakpoint detection (BFAST) and trend calculations (Greenbrown) are executed. The resulting output of the mobileEOM process contains the values of the analysis tools and links to the figures.

The mobile application was developed as hybrid web application. Applications of this type are developed using web development technologies (e.g., HyperText Markup Language (HTML), Cascading Style Sheets, and JavaScript) and are then exported as native mobile applications. To access the sensors and functions of the mobile device (e.g., GPS and file storage), the software PhoneGap was used. A native application can then be published in Google Play Store and Apple App Store. The user interface was developed using jQuery Mobile and the interactive map was integrated using Google Maps API.

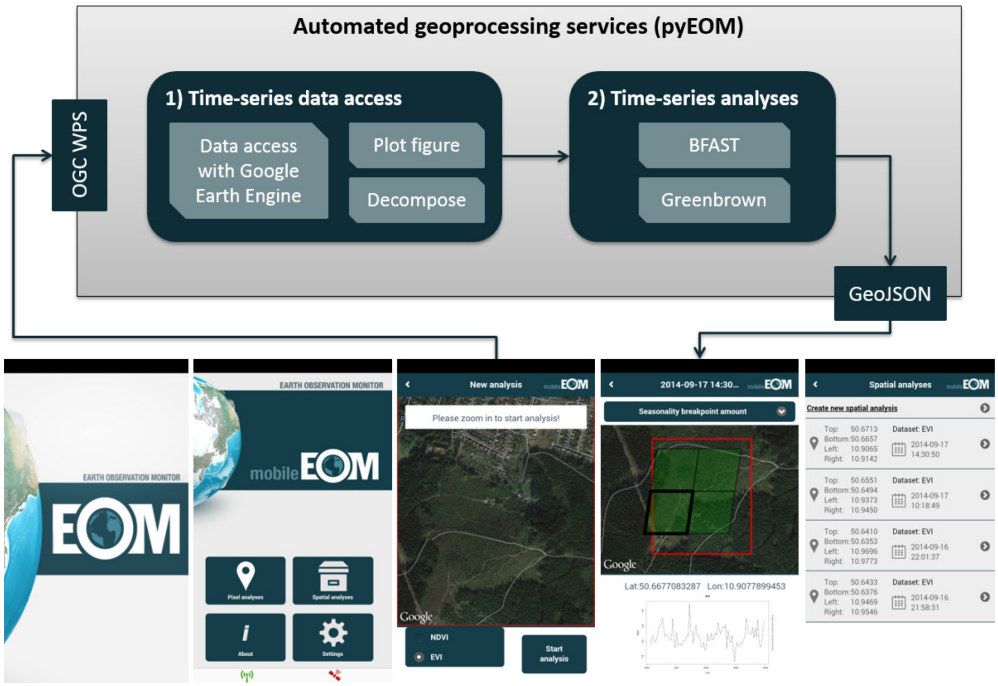


Figure 6.4: Interactions between the mobileEOM app and EOM middleware web services.

6.2.4 MySeasons App

The mobile application MySeasons was developed within the framework of the EU H2020 project, MyGEOSS. It combines EO time-series data with in-situ phenological data contributed by citizens. Users are able to collect data about the phenological cycle (e.g., the beginning of blossoming of specific plants) and submit these to a centralized data server. In addition, they are able to retrieve MODIS vegetation time-series data and a resulting phenological plot for the position where the data collection occurred. Users can use either the GPS position determined by the mobile device or a manually set position. The app was developed as hybrid web application. The Ionic Framework based on Apache Cordova (formerly known as PhoneGap) was used to export the web application to a native application for Android and iOS systems.

The app makes use of the EOM middleware processing services to retrieve the EO time-series data for the given point of interest (Figure 6.5). A web service was developed to integrate the existing EOM WPS (Figure 6.5, right) that undertakes time-series data integration and the TIMESAT analysis tool. The results from these are integrated into a single JSON output file, which is retrieved and further processed by the MySeasons application. This analysis works globally; the data is automatically requested on demand from Google Earth Engine by the EOM middleware. The phenological modeling tool TIMESAT is executed with a global parameter set. Figure 6.5 shows screenshots of the MySeasons application (left). In the map, the analysis can be conducted using the existing EOM services provided as an OGC WPS service (right). The resulting outputs are shown in the application (left bottom).

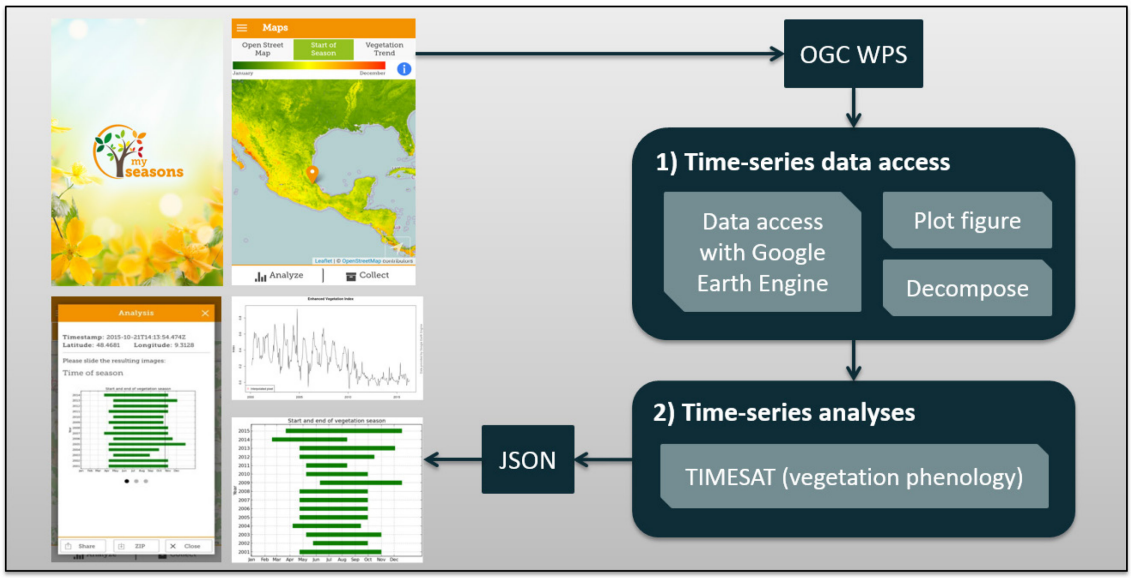


Figure 6.5: EOM middleware services (right) are used in the MySeasons App (left) to extract MODIS vegetation time-series data and phenological start and end dates derived on demand.

6.3 GEO-Wetlands Community Portal

The GEO-Wetlands Community Portal was developed within the framework of the EU-H2020 project Satellite based Wetland Observation Service, which led to the community portal of the Group on Earth Observation's (GEO) Wetlands Initiative. The objective of the portal is to share geospatial products and EO data, focusing specifically on wetland areas.

A user requirement survey undertaken in 2016 showed main issues to be addressed by the portal:

- Simple visualization tools for easily explore available datasets (EO data and thematic maps).
- Map production assistance and use case demonstrators.
- User-friendly tools for multi-source EO data discovery and access.
- Coordination with other portals and software.

In addition to the web portal (Subsection 6.3.2), web-based processing platforms are provided, which make use of the common EO time-series data structure: Open Data Cube for Wetlands (Subsection 6.3.3) and the Sentinel–1 Surface Water Dynamics Toolkit (Subsection 6.3.4).

In relation to the user requirements for platforms and web technology, the following requirements are supported: multi-source EO and geospatial data; data visualization; OGC standardization; user management (Open Date Cube only); RESTful web services; asynchronous web services (Open Date Cube only). The following data formats are supported: OGC services for data visualization (web portal only); interactive charts for data discovery; and figures for time-series analysis results (Open Date Cube only).

6.3.1 Middleware

The GEO-Wetlands Community Portal middleware bridges the gap between wetland users and geospatial data and includes EO data and thematic products available for individual wetlands. Geospatial thematic products (e.g., land use land cover, surface water dynamics, land surface temperature trend, and water quality) based on the outcomes of research projects are shared and made available to users. In addition to these products, available Landsat and Sentinel data are also shown to users based on the concept and implementation of the multi-source data discovery and access broker (Chapter 5.2). Although those datasets are not available immediately in the middleware, linked applications can be used to explore the EO time-series data (e.g., see Subsections 6.3.3 and 6.3.4).

Several tools from the regional multi-source data middleware concept have been integrated and linked to the geospatial products and wetlands available in the GEO-Wetlands Community Portal:

- Extraction of point-based values from time-series products (e.g., water quality, land surface temperature trend) based on a uniform time-series data structure and format.
- Automated EO data discovery and provision of download links for Landsat and Sentinel data for wetlands are integrated in the middleware.
- Automated data retrieval for USGS Landsat data, including the generation of full color images with GRASS GIS and the generation of an animated natural color visualization video.
- EO time-series data export to interactive exploration applications (e.g., Sentinel–1 Surface Water Dynamics Toolkit, Jupyter Notebooks, and Open Data Cube) to automatically link EO time-series data with web-based processing applications.

All the information and data provided by the middleware are published using RESTful web services from the Django Web Framework, which also handles multi-source data discovery, the access broker, and the links to external interactive exploration applications. For the extraction of point-based values from time-series products, OpenCPU is used to provide a RESTful web service that extracts data from the uniform EO time-series data structure.

Within the GEO-Wetlands Community Portal middleware available datasets, products, and wetlands are managed and user registration and authentication are provided. Django uses a PostgreSQL database backend with PostGIS extension that enables the storing geometry features and enables geospatial operations. Several extensions of Django, such as the Django REST framework, are used to publish RESTful web services. Other Python libraries are used for geospatial data discovery, access, and processing, such as GDAL, Shapely, OWSlib, NumPy, and Pandas. The pyroSAR framework has been used to automatically process Sentinel–1 data with the ESA SNAP toolbox in the background. A Jupyter Notebook was set up and linked with the middleware. This allows the middleware to automatically generate Jupyter Notebooks at the request of the user to enable users to work with EO data or products on their own without transferring data to the local computer.

6.3.2 Web portal

The web portal of the GEO-Wetlands Community Portal is the main entry point for discovering geospatial data and EO time-series data in the GEO-Wetlands Initiative. Research projects share geospatial products and link them to wetlands in a pre-defined data structure and format. An automated geospatial data publishing workflow was developed to process wetland-related products downloaded from a file server and published using OGC-compliant web services for data visualization and data download. The open source software GeoServer is used to make geospatial data available using the OGC Web Map (Tile) Service, OGC WFS for vector data, and OGC WCS for raster data. Time-series data is published using the OGC WMS with TIME extension. The web portal frontend was developed with HTML, Cascading Style Sheets, and JavaScript using open source libraries, such as AngularJS and Bootstrap. The communication with the middleware is based on RESTful web services.

For each of the wetlands, available EO data is presented in the Satellite data explorer (Figure 6.6). This demonstrates the use of the multi-source data discovery and access broker, which has been undertaken for all the wetlands integrated in the middleware. Users can interactively discover available EO time-series data with user-aligned data formats: interactive charts summarizing the available data per satellite mission and year, common metadata elements for all satellite scenes, as well as interactive visualization services, which can be added to the map.

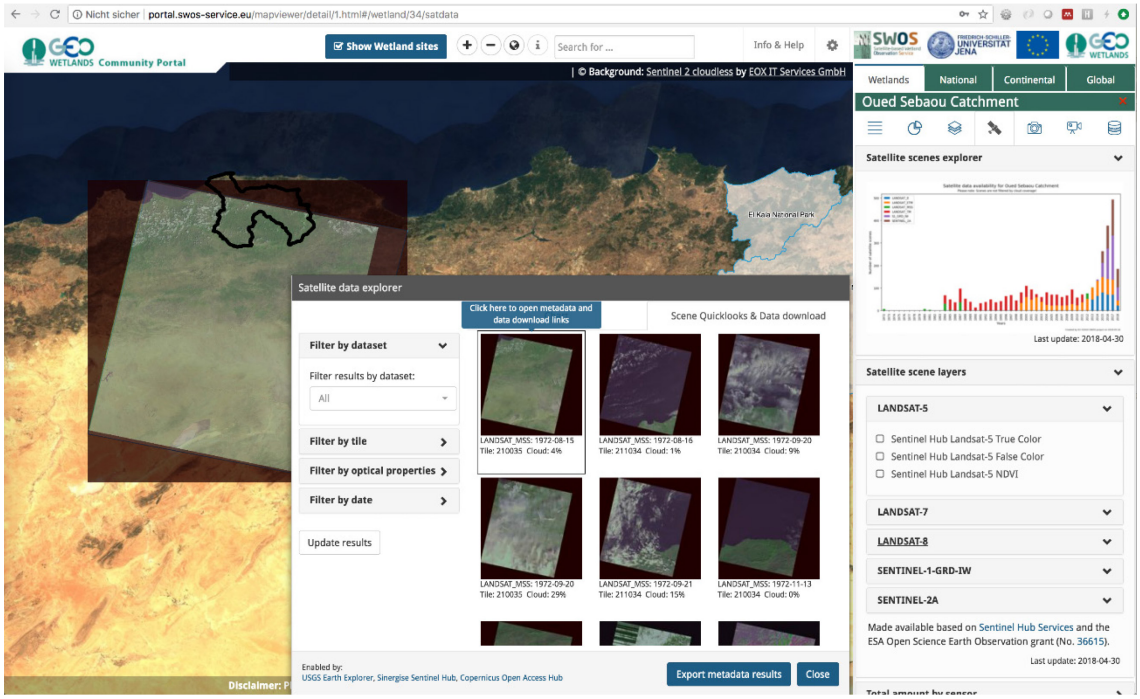


Figure 6.6: Screenshot of the GEO-Wetlands Community Portal highlighting the results of the EO data discovery broker for a wetland selected by the user.

6.3.3 Open Data Cube for Wetlands

An instance of the Open Data Cube software was setup within the GEO-Wetlands Initiative as first prototype to link the EO time-series data discovered and downloaded through the middleware system and ingested into the database. The Open Data Cube software makes use of a PostgreSQL database with the spatial PostGIS extension. The user interface of the Open Data Cube is based on the Django Web Framework, Celery and Redis for task processing, and bootstrap for frontend styling. In addition, a Jupyter Notebook server was set up with a direct connection to the Open Data Cube database. All these components need to be set up in the same infrastructure.

The Open Data Cube core application already provides several time-series analysis tools, [45] such as algorithms for water detection, fractional cover, spectral indices, urbanization, cloud cover statistics, and custom mosaics, which can be used directly once the data has been ingested into the database and registered with the user interface. Figure 6.7 shows an example screenshot from the user interface of the Open Data Cube for Wetlands with results from the water detection algorithm based on optical Landsat data. Currently only Landsat data has been ingested into the Open Data Cube for Wetlands. The time-series analysis tools provide their outputs in different formats, such as GeoTIFF, netCDF, and PNG. Python can be used with the Jupyter Notebook server connected to the Open Data Cube instance.

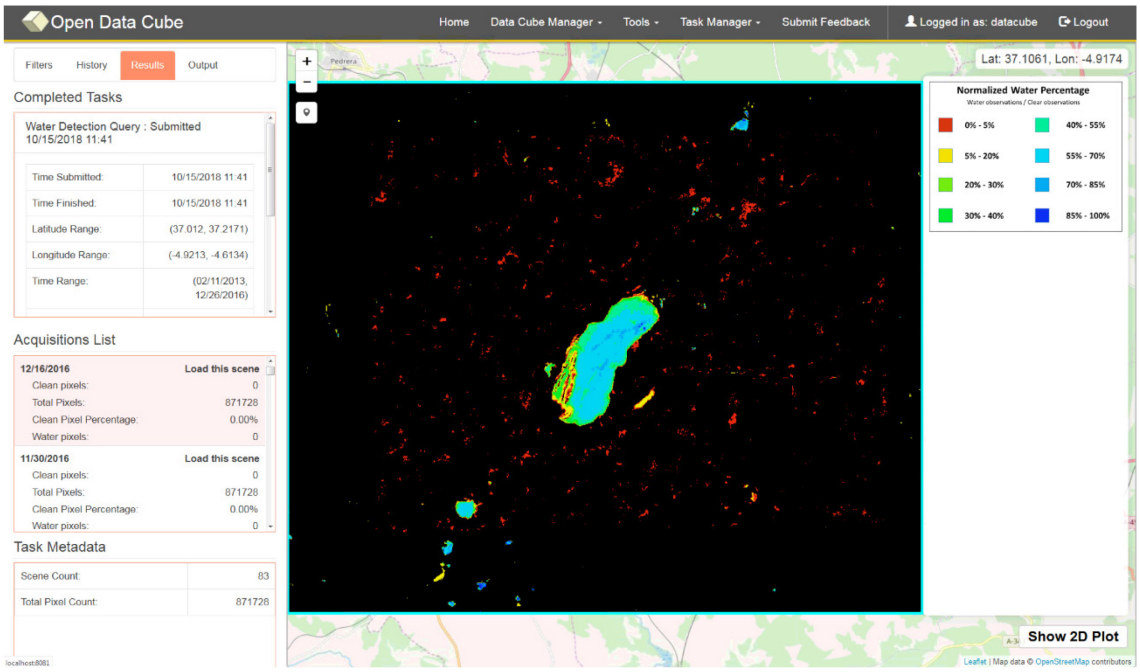


Figure 6.7: User interface of the CEOS Open Data Cube showing results of the water detection algorithm in the map from the Laguna de Fuente de Piedra wetland (Spain).

[45] https://www.opendatacube.org/data-cube-applications

6.3.4 Sentinel–1 Surface Water Dynamics Toolkit

The Sentinel–1 Surface Water Dynamics Toolkit allows interactive and web-based analysis of Sentinel–1 time-series data. The application was developed within R-Shiny[46], which allows the development of web applications directly in R without in-depth knowledge of web technologies. Figure 6.8 shows an example screenshot of the application. Sentinel–1 time-series data was automatically downloaded and preprocessed for the wetland of Laguna de Fuente de Piedra (Spain) for the year 2017. The original Sentinel has been preprocessed using pyroSAR and stored within the common EO time-series data structure, which is also optimized for analysis using R. The application can directly make use of standard R functions without conducting specific data processing in advance.

Within the application, users are able to use the toolkit to explore Sentinel–1 time-series data, calculate multi-temporal statistics, and create water dynamics products based on a simple thresholding (water/no water) algorithm (Martinis et al. 2015; Truckenbrodt et al. 2018). The user can visualize the available Sentinel–1 scenes and filter the result list according to various parameters. In the individual steps of the application, the user can analyze the time-series data without own data processing. The final water dynamics map, which is shown as a PNG figure within the application, can be downloaded as GeoTIFF.

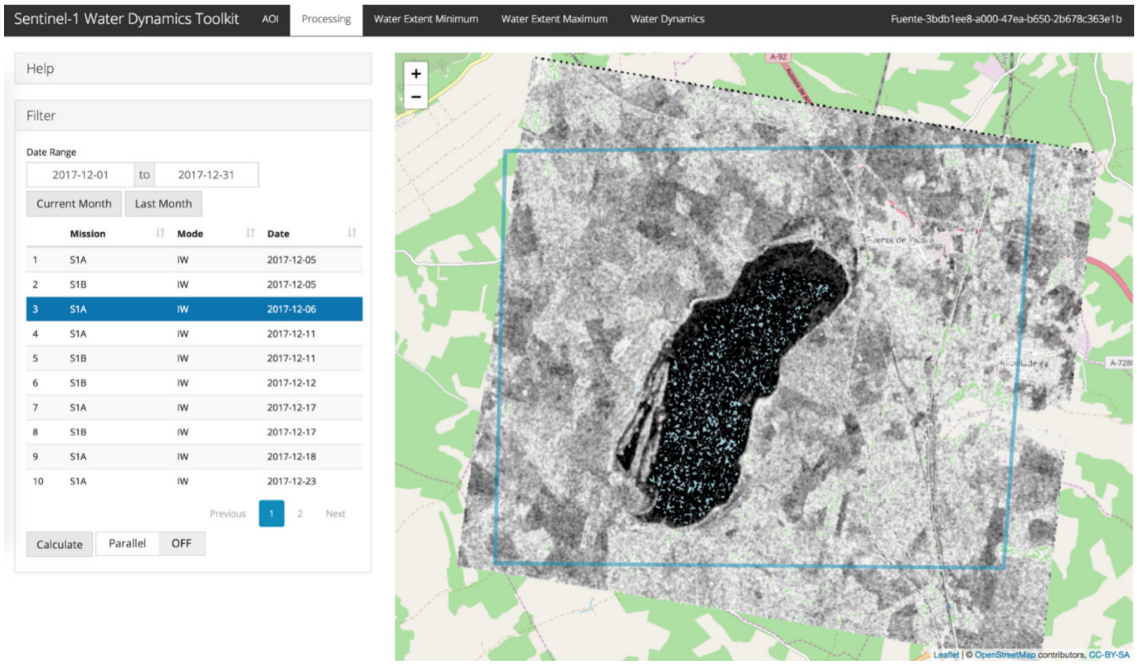


Figure 6.8: Sentinel–1 Water Dynamics Toolkit, developed within R and published as an R-Shiny-based web application.

[46] https://shiny.rstudio.com/

Chapter 7: Results and Discussion

Based on the concepts, methods, and example use cases designed and developed in this thesis, three overall results for a service-based exploration of EO time-series data can be summarized:

1. Centralization of EO time-series data at regional scales: A regional data middleware focuses on the centralization of multi-source EO data and tools and enables a simplification of application development through web services (Section 7.1).

2. Harmonization of EO time-series service interfaces: The OGC WPS specification enables the harmonization of the service interface for data discovery, access, and analysis. In addition, a discovery and access broker harmonizes requests to multi-source EO data providers (Section 7.2).

3. Standardization of EO time-series data structure and formats: A common EO time-series data structure enables simple use in geospatial applications, user-aligned output formats, and the simple use of time-series data in processing and analysis tools (Section 7.3).

In this chapter, these results are described and discussed in relation to other methods and implementations available in this field of research.

7.1 Centralization of EO time-series data at regional scales

The concept of the regional data middleware defined in this thesis focuses on providing EO time-series data for specific areas of interest selected either by individual users or the middleware provider. This allows the provision of web-based infrastructures independent of data providers, with the opportunity to link various geospatial data and analysis tools. As results of the thesis, the regional middleware approach (Subsection 7.1.1), application development based on centralized web services (Subsection 7.1.2), and regional use cases (Subsection 7.1.3) are summarized and discussed.

7.1.1 Regional data middleware approach

The concept of the regional data middleware has been designed and implemented to combine data discovery, access, and analysis. When hosted outside of cloud data providers, EO data needs to be transferred to an own server. However, datasets from several sources (e.g., Landsat, Sentinel, MODIS, and in-situ climate station data) can be combined and used in the middleware for further analysis. In addition, usage costs need to be calculated and compared carefully for both cloud-based and local server infrastructures. Depending on the size of the area of interest, one or the other solution is suitable. Regional analysis requires only the downloading of data for specific areas of interest. Focusing on regional data access and analysis in a self-developed platform provides the following advantages:

- Any analysis tool and additional data provision services can be made available.
- Users can follow the processing steps from data download to analysis results.
- Users can download EO data in the same format as used in the middleware and continue processing on their own infrastructure.

Any of the web-based infrastructures available for EO data access and analysis only fits for specific purposes. Google Earth Engine provides opportunities to easily develop algorithms on a global scale; however, only tools from the platform can be used (i.e., no external command-line tools). The ESA Thematic Exploitation Platforms focuses more on single scene analysis; thus, the satellite scenes selected by the user are copied to the analysis tool. Virtual environments, such as Amazon Web Services, Google Cloud, or Copernicus DIAS platforms, provide EO data and processing environments; however, users need to install and implement their data processing chains and analysis tools themselves. Soille et al. (2018) show similar results as data and algorithms need to be available in the same environment and that a shift from local workflows to the use of interactive visualization and processing is to be expected. Further, they recommended the use of a self-developed platform to be most flexible to the user requirements.

7.1.2 Application development

Multiple web and mobile applications that have been developed have demonstrated how users can work with EO time-series data without undertaking data processing on their own, and how the centralized services can be re-used in applications (Chapter 6). The focus on regional scales in these applications allows the hosting of EO time-series data, which can be directly used in analysis tools. In contrast with global or continental scales, the data storage and processing performance at regional and local scales is manageable. Three applications have been developed to demonstrate the concepts and methods described in this thesis:

- The Siberian Earth System Science Cluster (SIB-ESS-C),
- Earth Observation Monitor (EOM), including web and mobile applications, and
- GEO-Wetlands Community Portal, including a satellite scene explorer in the web portal, Open Data Cube for Wetlands, and the R-Shiny Sentinel–1 Surface Water Dynamics Toolkit.

All applications are based on web services of the self-developed middleware for time-series data discovery, access, plotting, and analysis. Application developers do not need to perform their own data downloads, processing, and analysis tasks. In addition, external applications are connected to the middleware, such as the Open Data Cube, Rasdaman, R-Shiny applications, and Jupyter Notebooks. Table 7.1 shows the criteria compiled on the basis of the user requirements for the evaluation of the applications developed. By combining all applications developed in this thesis, all requirements could be fulfilled. As each specific application has its focus, not all the requirements are relevant for each application (e.g., there is no EO data discovery in the EOM as the extraction of all available data is undertaken without specific filtering of data).

Table 7.1: Platforms developed in relation to user requirements compared to the review of existing platforms (Table 4.6).

	Siberian Earth System Science Cluster	Earth Observation Monitor	GEO-Wetlands Community Portal		
			Satellite scene explore	Open Data Cube	R-Shiny SWD App
EO time-series data	MODIS	MODIS	Landsat, Sentinel	Landsat	Sentinel-1
EO data visualization	☑	☒	☑	☒	☑
Time-series data download	Individual requests to web service, zipped archive	Zipped archive file	Links to each scene is provided	☒	☒
Analysis tools	Fixed set of tools	Fixed set of tools	N/A	Fixed set of tools	Surface Water Dynamics
Analysis output formats	Graphics	GeoTIFF, Graphics, CSV, interactive charts and maps	N/A	PNG, GeoTIFF, netCDF	GeoTIFF, interactive map
Data analysis with various programming languages	☑	☑	N/A	Python through Jupyter Notebook	R
EO data discovery	☒	☒	☑	☒	☑
Own user management	☑	☑	☒	☑	☒
Self-hosting	☑	☑	☑	☑	☑
Interfaces	REST, UI	REST, UI	REST, UI	Python, REST, UI	R, UI
OGC web services	OGC WMS, WCS, SOS, WPS	OGC WMS, WCS, SOS, WPS	N/A	☒	☒
Lineage information	☑	☑	N/A	☒	☒

Using regional data middleware with web services for EO data discovery, access, extraction, and analysis allows the following advantages:

- The automated access and processing of EO time-series data provided by web services allow simple integrations in web and mobile applications without the need for domain-specific processing knowledge.
- External applications (e.g., R-Shiny and Jupyter Notebooks) can directly make use of the common time-series data structure of the middleware.
- Data can be exported from the common time-series data structure and be automatically fed to external applications (e.g., Open Data Cube and Rasdaman).

Existing web-based platforms (e.g., Google Earth Engine and NASA Giovanni), focus primarily on pre-defined EO data and analysis tools. While Google Earth Engine can be integrated in applications using the Python library, others, such as NASA Giovanni, can only be used manually on the web portal and cannot be connected to linked external applications. Thus, the system architecture of web platforms needs to be clearly defined to enable the integration in several applications. The middleware approach allows a focus on individual scales and connections to any kind of external application.

7.1.3 Regional use cases

All the example applications developed have been used widely in research and education, focusing on regional aspects of environmental monitoring using EO time-series data. In the following paragraphs, statistics for the applications developed and two regional use cases for thematic experts and scientific research are described.

Usage statistics for EOM web and mobile applications

In total, for both web and mobile applications, 16,184 users from 150 countries worldwide were reached between January 2014 and July 2018, based on statistics from Google Analytics. In the EOM web portal, 14,456 analyses were conducted. Eighty percent of these were used for individual pixel time-series, and 20% for area time-series. The breakpoint detection algorithm had the most analysis conducted (45%), followed by phenological analyses (31%), and trend calculations (22%). The distribution of users per country is shown in Figure 7.1. Most users of the portal came from the USA, followed by Germany, Russia, the United Kingdom, and Brazil, while the most active users came from Germany, Iran, China, the USA, the United Kingdom, and Israel. Statistics for the mobile application show 2,373 users and 2,714 analyses conducted (70% of these for pixel time-series). Most users came from India, followed by Saudi Arabia, Germany, the United Kingdom, the USA, and the United Arab Emirates. The most active users came from India and Saudi Arabia, followed by the USA, Germany, Iran, and Italy.

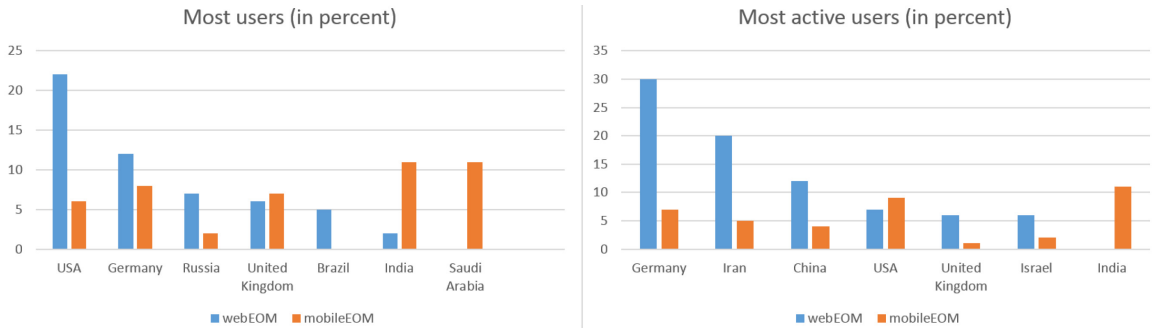


Figure 7.1: Top five countries with the most users and most active users of webEOM and mobileEOM as a percentage of the total numbers from January 2014 to July 2018.

EO time-series data visualization for thematic experts

Based on the visualization services of the SIB-ESS-C web portal, EO time-series data has been used by thematic experts to explain loss of vegetation in Siberia. The MODIS Level–3 products Vegetation Index (MOD13Q1, 16-day) and Burned Area (MCD45A1, monthly), both available for Siberia in the SIB-ESS-C middleware, have been investigated for an area near Yakutsk, Russia. A loss of vegetation could be identified in the MODIS Vegetation Index dataset between beginning of June and end of July 2012 (Figure 7.2, left and middle). Based on the MODIS Burned Area dataset, from July 2012, the loss of vegetation can be explained by fires which occurred in that month (Figure 7.2, right). As both EO time-series products have been made available in the SIB-ESS-C middleware with visualization services, thematic experts have been able to combine different types of thematic products in a single web-based system and without conducting data discovery, downloads, and visualizations on their own.

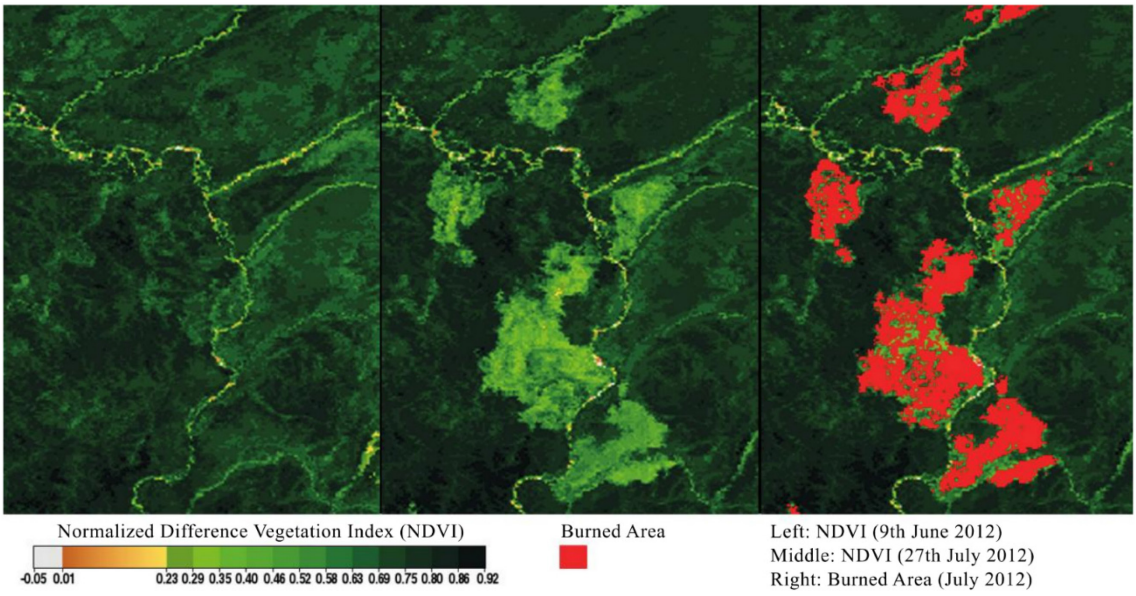


Figure 7.2: Combination of MODIS NDVI and MODIS Burned Area datasets (left: before the fire NDVI; middle: after the fire NDVI; right: burned areas) near Yakutsk, Russia (Eberle et al. 2013).

Deforestation monitoring for scientific research

Forest monitoring was conducted in several research projects with support of MODIS data discovery, access, and analysis by means of the EOM middleware. Breakpoint analyses were used to identify deforestation changes by location and time.

Change detection information was conducted in the Afromontane ecosystem of the Mambilla Plateau (Nigeria) using Landsat time-series and the EOM portal. EOM provided the 16-day MODIS Vegetation Index dataset for identifying changes based on breakpoint analysis. The Landsat image from 1988 was used as the reference and was compared with the images from 2001 and 2014 (Figure 7.3, left). Deforestation clearly occurred between 2001 and 2014, but cloud-free Landsat was unavailable to determine the dates of deforestation between these two dates. Time-series data from the NASA MODIS sensor from 2000 to 2014 were analyzed using the BFAST algorithm. The results from the web-based EOM analysis (Figure 7.3, right) show that deforestation began in 2002 (denoted by the blue area) and peaked in 2012 (denoted by the red area).

All the work associated with breakpoint analyses, including data access, has been undertaken by researchers using the EOM web portal, and without the need to download and process the required time-series data.

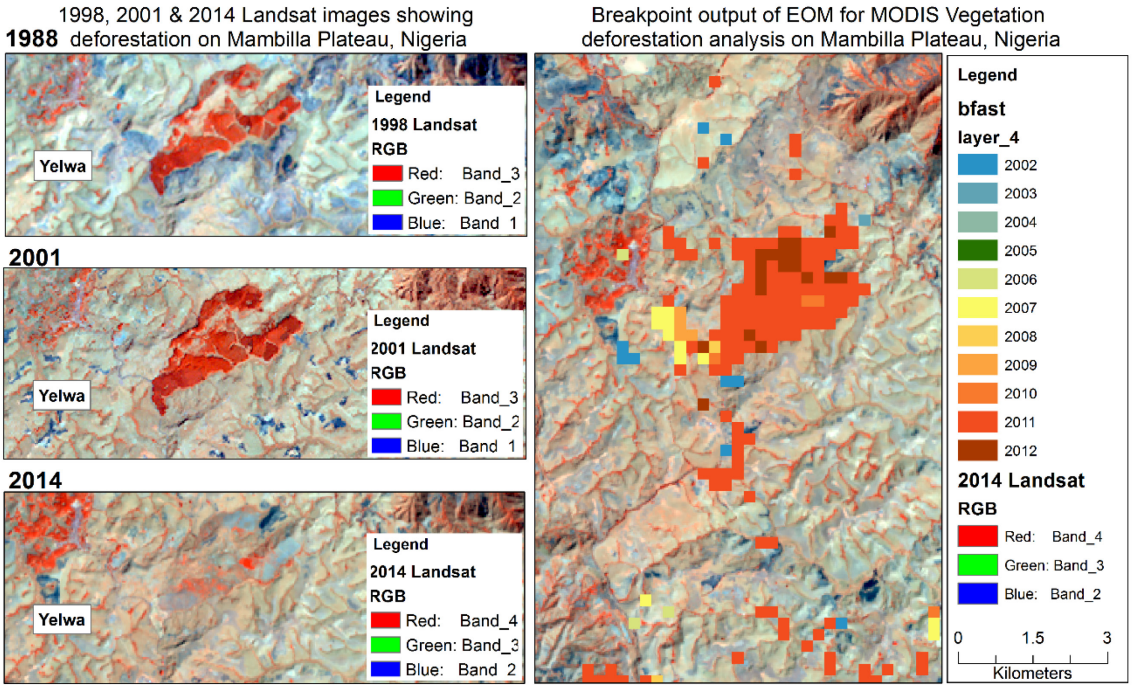


Figure 7.3: Twenty-six year Landsat time-series images showing a forest site near the village of Yelwa on Mambilla Plateau, Nigeria, before and after deforestation (left). Results of MODIS analysis overlaid on a Landsat image from 2014 show the pattern and years of deforestation near the village of Yelwa and neighboring forests (right).

7.2 Harmonization of EO Time-series Service Interfaces

As shown in the state-of-the-art and review chapters, diverse specifications for service interfaces are currently available that allowing for data discovery, access, and analysis. To simplify the use of web services for the exploration of EO time-series data, the service interfaces have been harmonized: Services for time-series data discovery, access, and analysis have been designed, developed, and implemented using the OGC WPS specification (Section 7.2.1). In addition, a brokering framework for multi-source EO data discovery and access was designed and developed based on services from multiple data providers (Section 7.2.2).

7.2.1 WPS-based EO web services

The services defined and developed for data discovery, access, extraction, and analysis aim to be re-used by application developers and by scientists who develop analysis algorithms. Although the individual service specifications for data discovery, access, and analysis have proved useful for machine-to-machine interaction, their utility for human–machine interaction lags. In order to provide a harmonized and uniform service interface and to allow multiple output formats for data discovery, access, and analysis, the OGC WPS specification was investigated and used as a generic processing service. The WPS specification allows the provision of standard-compliant web services and considers the most relevant properties—according to user requirements—for service execution, such as synchronous and asynchronous, as well as single and multiple output data formats. Using a single specification for all the tasks is considered to have standardized the service requests and responses, in contrast to various service specifications for each individual task (i.e., data discovery, access, and analysis). In this thesis, it has been shown that the WPS specification provides the necessary properties for all the tasks. The following advantages by the use of the WPS specification can be summarized:

- Uniform requests and responses for data discovery, access, and analysis.
- Multiple output formats to meet the different needs of user personas; all of the output formats can be changed depending on the user's request.
- The WPS specification allows a standardized provision of services, including synchronous and asynchronous execution and support for multiple outputs.

Although the WPS specification is not specifically designed to provide data discovery and data access, both of these tasks can also be considered to be processing tasks, which then suit the WPS specification. Especially the focus on a human-machine interaction is not yet defined on the tasks for EO data discovery and access in comparison to machine-to-machine communication. While research mostly focuses on technical details, such as

semantic processing (Sudmanns et al. 2018), process orchestration (Hofer et al. 2017), cloud-based processing (Veenendaal et al. 2016), and geospatial web technologies in general (Wagemann et al. 2018), the user-alignment of web services lags behind (Bordogna et al. 2016; Smith et al. 2016). This has been addressed in this thesis by introducing the harmonized WPS-based EO web services for data discovery, access, and analysis. For data analysis services, it has been shown that a direct connection between data access and data analysis is needed to enable users not only to get data or run analyses, but also to connect both services.

7.2.2 EO data discovery broker

A discovery broker connected to major EO data providers has been designed and implemented to achieve a uniform and standardized discovery of EO time-series data distributed across different data providers (Chapter 5.2). The brokering approach accesses web services based on a variety of service specifications and transfers metadata elements to a harmonized metadata structure. Using the brokering software, users do not need to connect to each of the data providers and their individual web services. In summary, the EO data discovery broker, as a result within this thesis, has the following advantages:

- Harmonized request method for all data providers connected to the broker.
- Harmonized metadata elements of resulting satellite scenes.
- Multiple output formats optimized for different user personas (e.g., GeoJSON, JSON, Shapefile, SQLite, CSV, Summary-Figure, and Summary-CSV).
- The discovery response has been automatically enhanced with the discovery results from multiple data providers for specific EO datasets.

In contrast with existing brokering services (e.g., GEOSS and ESA FedEO), multiple response output formats, which enable different user personas to work with the resulting output, and a full set of metadata elements are available. In comparison to other brokering research (Nativi Craglia et al. 2013; Previtali & Latre 2018), further enhancements have been integrated based on individual brokered resources, such as the retrieval of the best quick-look images and the insertion of web-based visualization services. Although the proposed data discovery broker requires more processing time than usual discovery services, users benefit from outputs that can be immediately interpreted and used in further work and a harmonized service interface.

The EO data discovery broker was implemented in the middleware and web portal of the GEO-Wetlands Community Portal, as described in the use case (Chapter 6.3).

7.3 Standardization of EO Time-series Data Structure and Formats

Specifications have been developed to standardize the EO time-series data structure and formats across multiple EO missions and data providers. Unification and interoperability are relevant components of a standardization process. Both of these matters need to be considered to make the exploration of EO time-series data simpler. Interoperability is important for enabling especially machine-to-machine communication; however, unification enables a harmonized set of data structure and formats to allow a uniform handling of EO time-series data using various processing and analysis tools (Subsection 7.3.1). In addition, user-aligned output formats were designed to meet the requirements of users (Subsection 7.3.2).

7.3.1 EO time-series data structure and format

A common time-series data structure and data format was designed to ensure uniform and standardized data provision and analysis (Chapter 5.3). Based on a review of analysis tools (e.g., Python, R, and GRASS GIS), a data structure and format that best fits all the tools have been implemented. Furthermore, automated workflows for exporting the data to additional data structures required by external geospatial tools (e.g., Rasdaman and Open Data Cube) have been provided to ensure simple use of the common data structure. Advantages of this data format can be summarized as follows:

- Direct usage of time-series data in Python and R.
- Direct provision of standard-compliant OGC services.
- Simple integrations for external geospatial tools (e.g., GRASS GIS, Rasdaman, and Open Data Cube).
- Standardized metadata records for time-series data.
- Extraction services to enable on-the-fly exploration of specific areas of interest.

In most cases, analysis and processing tools require input data with a specific structure for time-series data. Although there is no data structure that fits for all tools, commonalities can be found and further tools can easily be supported by creating additional files (e.g., a text file for the GRASS GIS space-time dataset and a recipe configuration file for the Rasdaman database). Data structures often depend on the type of analysis, whether it is a single-band or multi-band calculation. Thereafter, additional processing may be required. Several studies researched to overcome the current behavior of data search, order, download, and transformation into a web service-based exploration and analysis of time-series data in an efficient and optimized way (Colditz et al. 2008; Gallaher & Grant 2012; Van den Bergh et al. 2012). However, the transfer from research into production of

data providers is still not put into practice. The introduction of data cubes is currently the way forward in this direction, which is also discussed by data providers (ESA 2018).

7.3.2 User-aligned output formats

Multiple output formats have been defined and implemented in data discovery, access, and analysis that enable users to better interact with the resulting data (Sections 5.1.3, 5.2.3, and 5.3.3.3). Output formats defined for different user personas allow user-specific responses. In addition, statistics summarized as text and figures provide a simple overview of EO time-series data and the results of the analysis. Standardized and service-based output formats for geospatial data allow users to interactively explore the resulting data in web maps and GIS software without having to downloading complete datasets. Providing multiple user-aligned output formats has the following advantages:

- Overview of available data and time-series can be used for immediate analysis.
- Users can work with results in different data formats (e.g., CSV, JSON).
- Web services allow the interactive visualization of and access to geospatial data.
- Additional output formats allow users to easily explore the resulting data (e.g., providing summarized statistics and figures).

In most cases, traditional data formats (e.g., XML or JSON for data discovery) only focus on machine-to-machine interaction and users need to process these data formats on their own to analyze the results of the request. Although familiarizing users with these data formats can be seen as useful, users should focus on their real interests, for example, obtaining simple overviews of the data, directly visualizing the data, or directly using the resulting data in their software of choice. Thus, multiple and user-aligned output formats are required—as opposed to providing only a single format, which is the case with most of the standard-compliant OGC web services. This has also been researched and concluded by Bordogna et al. (2016) especially focusing on time-series data in a study case of a spatial data infrastructure. Although the demand on improving data access and data analysis especially for user personas other than developers has already been investigated (Budhathoki et al. 2008; Hennig & Belgiu 2011; Brown et al. 2014; La Torre et al. 2017), it has not yet been fully considered by data providers in terms of providing user-aligned output formats.

Chapter 8: Conclusions and Outlook

This final chapter provides responses to the research questions posed in the introduction, conclusions, and an outlook.

8.1 Responses to the research questions

Research questions were posed in the introduction of this thesis. They can be answered as follows:

1. **How to design a user-aligned discovery, access, and analysis for EO time-series data based on standard-compliant web services?**

Requirements for multiple user personas have been defined for data discovery, access, and analysis, which conclude that multiple data and output formats are often necessary for different user personas. The usual standardized web service specifications for data discovery and access have been designed for machine-to-machine communication. Thus, a uniform service specification has been defined and used in the applications developed to enable a human–machine interface. This specification is based on the standard-compliant OGC WPS, which was designed to provide processing services. Processing services have the ability to provide multiple output formats in the web service response and thus fulfill the requirements of the user personas. Within this thesis, this service specification has also been used for data discovery and access in addition to data analysis. Besides a uniform specification, user-aligned output formats are necessary to fulfill the requirements of user-aligned services, which have been defined for time-series data and analysis results.

2. **What are the technical requirements for accessing and processing multi-source EO time-series data?**

Multi-source EO data are often provided by several data providers that offer different data access services, data structures, and data formats. Therefore, data access still needs to be adjusted for each data provider. As structure and formats are diverse for multi-source EO time-series data, much information is necessary to process them, such as a no data value, a scaling factor, and quality flags. In addition, for the visualization of EO time-series data, styling information is required for each dataset based on valid data ranges and color definitions that depend on the thematic product (e.g., vegetation index, or temperature). Both access to and processing of multi-source EO time-series data have been standardized using a brokering approach linked to several data providers and adjusted for different EO dataset structures. All technical requirements for data access and processing have been centralized within the multi-source EO data broker.

3. What are the data requirements for analysis- and application-ready formats and how must EO time-series data hence be organized?

Based on a review of requirements for raster time-series data in programming languages and existing analysis tools, a uniform data specification for the organization of raster time-series data has been defined. Focusing on raster time-series data, each individual date of the time-series is stored in a separate geospatial raster file, which is linked with the date in both the filename and a text file. In addition to the storage of geospatial data, further information about the time-series, such as the start time, end time, and annual frequency are needed to ensure straightforward use in several geospatial tools ("analysis-ready data"). Although some tools can directly work with the uniform data structure, further data structures and configuration files are required for other tools and services. To overcome this issue, export tools have been integrated to automatically provide correct data structures and formats for individual geospatial tools (e.g., Rasdaman, Open Data Cube, and GRASS GIS), which have been defined as application-ready data. This enables the straightforward use of EO time-series data in geospatial tools to simplify data analysis.

8.2 Conclusions

Existing gaps between users and data providers as well as EO time-series data and geospatial analysis tools have been investigated in this thesis. In summary, the following conclusions can be drawn.

Brokering approaches facilitate the provision of user-aligned web services, specific multiple output formats, and uniform EO data discovery and access. This allows users to focus on their primary work rather than having to deal with different service specifications and data formats. Current web services for EO data discovery and access focus mainly on machine-to-machine communication and thus often do not consider the needs of users. Specifically, as concerns data access, services offering access to analysis-ready data are necessary. Web services for data discovery and access need to be simplified and focused to a greater extent on the requirements of users, for example, by providing various service response formats. Solutions can be provided either by each data provider (e.g., providing user-aligned web services) or in middleware systems based on the brokering approach to enhance the services of multiple data providers.

The focus on regional scales rather than on a global scale enables the development of processing platforms, which connect various multi-source EO data and analysis tools. Due to the limited size of the data required for specific regions, such a processing platform can be either set up on local server infrastructure or within cloud providers. Any specific geospatial data and analysis tool necessary for users of the platform can be integrated

and made available to them. The focus on individual regions has also been considered by other initiatives, such as Open Data Cube, which has been set up for individual countries.

Extraction services foster the immediate access of EO data for an area of interest. Linked with analysis services, information can be generated immediately (e.g., vegetation trends and changes). Especially for mobile applications, users need to obtain access to time-series data as soon possible as they wait for the resulting data in the field. As the statistics from the use of the EOM applications show a key interest in point-based time-series extractions, EO data archives need to be available in a pre-processed and cloud-optimized data format rather than as zipped archive files, which is often the case currently.

Cloud-optimized data formats are also necessary to increase access and processing performance. The straightforward use of EO data often depends on the data format that is provided by the data provider. In addition, cloud-optimized data formats need to be directly available with links to the various files. As data formats have often been adjusted to different user communities, such as netCDF data for climate scientists, various data formats and structures have been introduced with new EO missions. Thus, geospatial tools should be enabled to easily convert the original data format in other formats. Even more, analysis tools should support EO time-series data formats and structures.

A centralized platform for the exploration of EO time-series data is useful for focusing on the specific requirements of users. EO data directly linked to geospatial tools in web and mobile application allows users to focus on their primary interest, the monitoring of environmental changes. Each of the user personas discussed in this thesis focuses on different aspects. However, they have in common that they are not EO data-processing specialists. It is necessary that they are able to use user-aligned applications and web services that can handle all obstacles to the effective use of multi-source EO time-series data and their links to geospatial processing and analysis tools.

Web services for data access connected with data analysis tools enable on-the-fly interpretation of EO time-series data without the need for users to undertake data processing. Applications are ready to provide analysis tools and services for time-series data in web-based and local environments. However, EO data currently often does not have the correct data structure and is not in the correct format to be easily used in these applications. To link EO time-series data with geospatial applications, either the data needs to be converted into a suitable data structure or the applications need to be adjusted to meet the original data structure, which has been the focus with the development of a uniform EO time-series data format and structure. The workflow for publishing algorithms close to the data needs to be further simplified and standardized.

8.3 Outlook

Further research work within this field focuses still on bridging the gap between users and providers of EO data and services as well as on new approaches for geospatial data formats, interoperability, data cubes, and cloud-based architectures.

Although data formats have already been discussed in this thesis, new cloud-optimized data formats are currently investigated (COG 2018), which provides new ways of working with geospatial raster data. With such a new way to organize and access geospatial data based on simple web service requests, both users and developers of algorithms need to learn how to handle this kind of data, which is different from working with local files. A growing trend among satellite imagery providers is the provision of preprocessed "analysis-ready data" to simplify the use of EO time-series data (CEOS 2018a; Dwyer et al. 2018; Holmes 2018a; Siqueira et al. 2019). Several organizations (e.g., USGS and CEOS) have recently begun to define analysis-ready data for specific EO missions to establish a common understanding of EO data preprocessing.

In addition, the interoperability of searches for satellite imagery needs to be further researched, as there is currently no standardized specification, which focuses on raster time-series data as shown in this thesis. This is currently been addressed by the STAC initiative (Holmes 2017b), which commenced in November 2017. The lack of a clear standard for massive amounts of imagery data for use with current web technologies, such as the JSON format and RESTful architecture, has been identified as a core problem by 14 organizations (e.g., Amazon, DigitalGlobe, Google, Planet, Radiant Earth, Element84, and Development Seed). A native cloud geospatial architecture is foreseen that will make it easy to crawl and search cloud-optimized GeoTIFF datasets (Holmes 2017a). The STAC aims to standardize how geospatial data is made available in the Web. Analysis-ready data and STAC are closely connected. A first "Satellite Data Interoperability" workshop was hosted by the USGS in August 2018 to discuss these topics. Over 100 participants from international space agencies, governments, research, non-profit, and commercial organizations discussed workflows, standards, upcoming developments, and how to benefit from analysis-ready data and STAC (Holmes 2018b).

Further research is necessary to standardize data cubes for EO time-series data. CEOS aims to launch several national data cubes in the near future (CEOS 2018b). As there is not only a single data cube software, interoperability between and a standardized specification for them are discussed. This can also enable a federation of data cubes, which allows linking of data cubes hosted with different data providers (Baumann 2019).

The use of cloud-based architectures in an efficient and optimized manner for all kind of user personas is discussed as a research topic in combination with distributed processing of geospatial data. New technologies and the further development of specifications have been realized by the OGC in "testbeds," which are "collaborative efforts to define, design, develop, and test candidate interface and encoding specifications" (OGC 2018b). In recent OGC testbeds, a major focus has been EO and cloud computing (Testbed 13[47] in 2017), as well as exploitation platforms and big data cloud processing (Testbed 14[48] in 2018). The objective of the EO-related components in both testbeds has been to encourage the standardized deployment and execution of big data-processing applications in cloud environments (Simonis 2018). In addition, the standardization of web services to provide reproducible algorithms as executable web services need to be further researched, which is currently based on a containerization approach (Hu et al. 2018).

Cloud-based architectures are further researched by many organizations allowing to host geospatial and EO data as well as algorithms in the cloud. With the Copernicus satellites and new satellites that will be launched in the future, the provision of this data and the processing capabilities needs to be adjusted according to the increasing amount of data, which, for large areas, will not be able to be downloaded to local infrastructure in the foreseeable future. As such, big data projects, cloud evolution strategies, and cloud-based processing platforms have been defined and established by major international and national organizations (e.g., NASA and ESA). With the EU project openEO, a multi-cloud service specification and interface is being investigated, which aims to harmonize data access and data analysis between cloud environments (Schramm et al. 2019). In 2014, the US Government established a "National Plan for Civil Earth Observations" (U.S. Government 2014), which led to a "Big Earth Data Initiative." NASA's Earth Observing System Data and Information System contributes to this initiative by bringing its data into the cloud (Blumenfeld 2018). The European Commission launched the Copernicus DIAS initiative leading to multiple cloud data providers in Europe facilitating access to Copernicus data (Copernicus Observer 2017).

In summary, an important part of future research and innovation is based on analysis-ready data, increased interoperability with modern web-based technologies, the increased availability of geospatial data in clouds, and cloud-based processing architectures and platforms, all of them related to user-aligned exploration of EO time-series data.

[47] http://www.opengeospatial.org/pressroom/pressreleases/2751
[48] http://www.opengeospatial.org/projects/initiatives/testbed14

References

Acker, J. G. & Leptoukh, G. (2007) ‘Online Analysis Enhances Use of NASA Earth Science Data’, *Eos, Transactions American Geophysical Union*, 88(2), p. 14. doi: 10.1029/2007EO020003.

Adams, B. & Gahegan, M. (2014) ‘Emerging data challenges for next-generation spatial data infrastructure’, in Winter, S. and Rizos, C. (eds) *Proceedings of Research at Locate’14*. Canberra, Australia, pp. 118–129.

Appel, M., Lahn, F., Buytaert, W. & Pebesma, E. (2018) ‘Open and scalable analytics of large Earth observation datasets: From scenes to multidimensional arrays using SciDB and GDAL’, *ISPRS Journal of Photogrammetry and Remote Sensing*. The Authors, 138, pp. 47–56. doi: 10.1016/j.isprsjprs.2018.01.014.

Arenas, H., Aussenac-Gilles, N., Comparot, C. & Trojahn, C. (2017) ‘Semantic Integration of Geospatial Data from Earth Observations’, in Ciancarini, P., Poggi, F., Horridge, M., Zhao, J., Groza, T., Suarez-Figueroa, M. C., D’Aquin, M., and Presutti, V. (eds) *Knowledge Engineering and Knowledge Management*. Cham: Springer International Publishing (Lecture Notes in Computer Science), pp. 97–100. doi: 10.1007/978-3-319-58694-6.

Ariza-Porras, C., Bravo, G., Villamizar, M., Moreno, A., Castro, H., Galindo, G., Cabera, E., Valbuena, S., et al. (2017) ‘CDCol: A Geoscience Data Cube that Meets Colombian Needs’, in Solano, A. and Ordoñez, H. (eds) *Advances in Computing*. Cham: Springer International Publishing, pp. 87–99. doi: 10.1007/978-3-319-66562-7_7.

Aschbacher, J. & Milagro-Pérez, M. P. (2012) ‘The European Earth monitoring (GMES) programme: Status and perspectives’, *Remote Sensing of Environment*. Elsevier Inc., 120(2012), pp. 3–8. doi: 10.1016/j.rse.2011.08.028.

Astsatryan, H., Hayrapetyan, A., Narsisian, W., Asmaryan, S., Saghatelyan, A., Muradyan, V., Giuliani, G., Guigoz, Y., et al. (2015) ‘An interoperable cloud-based scientific GATEWAY for NDVI time series analysis’, *Computer Standards & Interfaces*. Elsevier B.V., 41, pp. 79–84. doi: 10.1016/j.csi.2015.02.001.

Atzberger, C. (2013) ‘Advances in remote sensing of agriculture: Context description, existing operational monitoring systems and major information needs’, *Remote Sensing*, 5(2), pp. 949–981. doi: 10.3390/rs5020949.

Bai, Y. & Di, L. (2011) ‘Providing access to satellite imagery through OGC catalog service interfaces in support of the Global Earth Observation System of Systems’, *Computers & Geosciences*. Elsevier, 37(4), pp. 435–443. doi: 10.1016/j.cageo.2010.09.010.

Baresi, L., Miraz, M. & Plebani, P. (2016) ‘A distributed architecture for efficient Web service discovery’, *Service Oriented Computing and Applications*. Springer London, 10(1), pp. 1–17. doi: 10.1007/s11761-015-0184-z.

Baumann, P. (2009a) ‘The OGC web coverage processing service (WCPS) standard’, *GeoInformatica*, 14(4), pp. 447–479. doi: 10.1007/s10707-009-0087-2.

Baumann, P. (2009b) ‘Web Coverage Processing Service (WCPS) Language Interface Standard. OGC 08-068r2’. Open Geospatial Consortium Inc.

Baumann, P. (2012) ‘OGC WCS 2.0 Interface Standard - OGC 09-110r4’. Open Geospatial Consortium Inc.

Baumann, P. (2017) *The Datacube Manifesto*. Available at: http://earthserver.eu/tech/datacube-manifesto (Accessed: 7 October 2018).

Baumann, P. (2019) ‘From analysis-ready data to analysis-ready services: Challenges and helpers for EO service providers’, in Soille, P., Loekken, S., and Albani, S.

(eds) *Proceedings of the 2019 conference on Big Data from Space*. Munich, Germany: Publications Office of the European Union, pp. 69–72.

Baumann, P., Dehmel, A., Furtado, P., Ritsch, R. & Widmann, N. (1998) 'The multidimensional database system RasDaMan', in *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*. Seattle, Washington, USA, pp. 575–577. doi: 10.1145/276305.276386.

Baumann, P., Mazzetti, P., Ungar, J., Barbera, R., Barboni, D., Beccati, A., Bigagli, L., Boldrini, E., et al. (2016a) 'Big Data Analytics for Earth Sciences: the EarthServer approach', *International Journal of Digital Earth*. Taylor & Francis, 9(1), pp. 3–29. doi: 10.1080/17538947.2014.1003106.

Baumann, P., Meissl, S. & Yu, J. (2014) 'OGC Web Coverage Service 2.0 Interface Standard - Earth Observation Application Profile'. Open Geospatial Consortium Inc.

Baumann, P., Merticariu, V., Dumitru, A. & Misev, D. (2016b) 'Standards-based services for big spatio-temporal data', *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, 41(July), pp. 691–699. doi: 10.5194/isprsarchives-XLI-B4-691-2016.

Baumann, P. & Rossi, A. P. (2016) 'Datacubes as a service paradigm', in *Proceedings of the 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, pp. 186–188. doi: 10.1109/IGARSS.2016.7729039.

Beaujardiere, J. De (2006) 'OpenGIS Web Map Server Implementation Specification - OGC 06-042'. Open Geospatial Consortium Inc.

Beaulieu-Jones, B. K. & Greene, C. S. (2017) 'Reproducibility of computational workflows is automated using continuous analysis', *Nature Biotechnology*. Nature Publishing Group, 35(4), pp. 342–346. doi: 10.1038/nbt.3780.

Berger, M., Moreno, J., Johannessen, J. a., Levelt, P. F. & Hanssen, R. F. (2012) 'ESA's sentinel missions in support of Earth system science', *Remote Sensing of Environment*. Elsevier Inc., 120, pp. 84–90. doi: 10.1016/j.rse.2011.07.023.

Van den Bergh, F., Wessels, K. J., Miteff, S., Van Zyl, T. L., Gazendam, A. D. & Bachoo, A. K. (2012) 'HiTempo: a platform for time-series analysis of remote-sensing satellite data in a high-performance computing environment', *International Journal of Remote Sensing*, 33(15), pp. 4720–4740. doi: 10.1080/01431161.2011.638339.

Bernard, L., Mäs, S., Müller, M., Henzen, C. & Brauner, J. (2014) 'Scientific geodata infrastructures: challenges, approaches and directions', *International Journal of Digital Earth*, 7(7), pp. 613–633. doi: 10.1080/17538947.2013.781244.

Birkin, M. (2013) 'Big Data Challenges for Geoinformatics', *Geoinformatics & Geostatistics: An Overview*, 01(01), pp. 2012–2013. doi: 10.4172/2327-4581.1000e101.

Blumenfeld, J. (2018) *NASA EOSDIS Role in the Big Earth Data Initiative*. Available at: https://earthdata.nasa.gov/eosdis-role-in-bedi (Accessed: 14 October 2018).

Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C. & Orchard, D. (2004) 'Web Services Architecture'. W3C. Available at: https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/ (Accessed: 8 October 2018).

Bordogna, G., Kliment, T., Frigerio, L., Brivio, P., Crema, A., Stroppiana, D., Boschetti, M. & Sterlacchini, S. (2016) 'A Spatial Data Infrastructure Integrating Multisource Heterogeneous Geospatial Data and Time Series: A Study Case in Agriculture', *ISPRS International Journal of Geo-Information*, 5(5), p. 73. doi: 10.3390/ijgi5050073.

Boulton, G. (2018) 'The challenges of a Big Data Earth', *Big Earth Data*. Taylor & Francis, 2(1), pp. 1–7. doi: 10.1080/20964471.2017.1397411.

Brauner, J., Foerster, T., Schaeffer, B. & Baranski, B. (2009) 'Towards a Research Agenda for Geoprocessing Services', in *Proceedings of the 12th AGILE International Conference on Geographic Information Science*. Hannover, Germany, pp. 1–12.

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E. & Yergeau, F. (2008) 'Extensible Markup Language (XML) 1.0 (Fifth Edition)'. W3C. Available at: https://www.w3.org/TR/2008/REC-xml-20081126/ (Accessed: 8 October 2018).

Brown, M. E., Carroll, M. L. & Escobar, V. M. (2014) 'User needs and assessing the impact of low latency NASA Earth observation data availability on societal benefit', *Space Policy*. Elsevier Ltd, 30(3, Part A), pp. 135–137. doi: 10.1016/j.spacepol.2014.05.002.

Budhathoki, N. R., Bruce, B. & Nedovic-Budic, Z. (2008) 'Reconceptualizing the role of the user of spatial data infrastructure', *GeoJournal*, 72(3–4), pp. 149–160. doi: 10.1007/s10708-008-9189-x.

Bychkov, I., Ruzhnikov, G., Fedorov, R. & Shumilov, A. (2015) 'Building the distributed WPS-services execution environment', *arXiv*, pp. 1–17. Available at: https://arxiv.org/abs/1503.07626.

Calaway, R. (2017) *foreach package - R documentation*. Available at: https://www.rdocumentation.org/packages/foreach/versions/1.4.4 (Accessed: 4 March 2019).

Caumont, H., Brito, F. & Boissier, E. (2014) 'Big Earth Sciences & the new Platform Economy', in *Proceedings of the 2014 conference on Big Data from Space (BiDS'14)*. Frascati, Italy: JRC conference e-proceedings series, pp. 407–410.

Celesti, A., Mulfari, D., Fazio, M., Villari, M. & Puliafito, A. (2016) 'Exploring Container Virtualization in IoT Clouds', in *Proceedings of the 2016 IEEE International Conference on Smart Computing, SMARTCOMP*. IEEE, pp. 1–6. doi: 10.1109/SMARTCOMP.2016.7501691.

CEOS (2018a) *CEOS Analysis Ready Data*. Available at: http://ceos.org/ard/ (Accessed: 7 October 2018).

CEOS (2018b) *The 'Road to 20' International Data Cube Deployments*. Available at: https://docs.wixstatic.com/ugd/8959d6_cfcba3751fe642bc9faec776ab98cb20.pdf (Accessed: 4 March 2019).

Čepický, J. & Becchi, L. (2007) 'Geospatial Processing via Internet on Remote Servers - PyWPS', *OSGeo Journal*, 1(May), pp. 1–4. doi: 10.1007/978-0-387-74674-6_2.

Čepický, J. & De Sousa, L. M. (2016) 'New implementation of OGC Web Processing Service in Python programming language. PyWPS-4 and issues we are facing with processing of large raster data using OGC WPS', *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, 41(June), pp. 927–930. doi: 10.5194/isprsarchives-XLI-B7-927-2016.

Clauss, K. (2018) *Earth Engine Apps: Sentinel-1 2017 time series*. Available at: https://kersten.users.earthengine.app/view/sentinel-1-2017time-series (Accessed: 2 March 2019).

Clinton, D. (2018) *OpenSearch 1.1 Draft 6*, *OpenSearch*. Available at: https://github.com/dewitt/opensearch/blob/master/opensearch-1-1-draft-6.md (Accessed: 1 March 2019).

COG (2018) *Cloud optimized GeoTIFF*. Available at: http://www.cogeo.org (Accessed: 7 October 2018).

Colditz, R. R., Conrad, C., Wehrmann, T., Schmidt, M. & Dech, S. (2008) 'TiSeG: A Flexible Software Tool for Time-Series Generation of MODIS Data Utilizing the Quality Assessment Science Data Set', *IEEE Transactions on Geoscience and*

Remote Sensing, 46(10), pp. 3296–3308. doi: 10.1109/TGRS.2008.921412.

Cooper, A. (2004) *The Inmates Are Running the Asylum: Why High Tech Products Drive Us Crazy and How to Restore the Sanity*. 2nd edn. Pearson Higher Education.

Copernicus Observer (2017) *The upcoming Copernicus Data and Information Access Services*. Available at: http://copernicus.eu/news/upcoming-copernicus-data-and-information-access-services-dias (Accessed: 14 October 2018).

Craglia, M., Hradec, J., Nativi, S. & Santoro, M. (2017) 'Exploring the depths of the global earth observation system of systems', *Big Earth Data*. Taylor & Francis, 1(1–2), pp. 1–26. doi: 10.1080/20964471.2017.1401284.

Cui, S., Dumitru, C. O. & Datcu, M. (2014) 'Semantic annotation in earth observation based on active learning', *International Journal of Image and Data Fusion*, 5(2), pp. 152–174. doi: 10.1080/19479832.2013.858778.

Davis, B. N., Werpy, J., Friesz, A., Impecoven, K., Quenzer, R. L., Maiersperger, T. & Meyer, D. J. (2015) 'Interactive Access to LP DAAC Satellite Data Archives Through a Combination of Open-Source and Custom Middleware Web Services', *IEEE Geoscience and Remote Sensing Magazine*, 3(4), pp. 8–20. doi: 10.1109/MGRS.2015.2505999.

DeVries, B., Verbesselt, J., Kooistra, L. & Herold, M. (2015) 'Robust monitoring of small-scale forest disturbances in a tropical montane forest using Landsat time series', *Remote Sensing of Environment*. Elsevier Inc., 161, pp. 107–121. doi: 10.1016/j.rse.2015.02.012.

Díaz, L., Remke, A., Kauppinen, T., Degbelo, A., Foerster, T., Stasch, C., Rieke, M., Schaeffer, B., et al. (2012) 'Future SDI - Impulses from Geoinformatics Research and IT Trends', *International Journal of Spatial Data Infrastructures Research*, 7, pp. 378–410. doi: 10.2902/1725-0463.2012.07.art18.

Dubois, G., Schulz, M., Skøien, J., Bastin, L. & Peedell, S. (2013) 'eHabitat, a multi-purpose Web Processing Service for ecological modeling', *Environmental Modelling & Software*. Elsevier Ltd, 41, pp. 123–133. doi: 10.1016/j.envsoft.2012.11.005.

Dutrieux, L. & DeVries, B. (2014) 'bfastSpatial: Set of utilities and wrappers to perform change detection on satellite image time-series'. Available at: https://github.com/loicdtx/bfastSpatial (Accessed: 16 April 2019).

Dutrieux, L. P., Verbesselt, J., Kooistra, L. & Herold, M. (2015) 'Monitoring forest cover loss using multiple data streams, a case study of a tropical dry forest in Bolivia', *ISPRS Journal of Photogrammetry and Remote Sensing*, 107, pp. 112–125. doi: 10.1016/j.isprsjprs.2015.03.015.

Dwyer, J., Roy, D., Sauer, B., Jenkerson, C., Zhang, H. & Lymburner, L. (2018) 'Analysis Ready Data: Enabling Analysis of the Landsat Archive', *Remote Sensing*, 10(9), pp. 1–24.

Eberle, J., Clausnitzer, S., Hüttich, C. & Schmullius, C. (2013) 'Multi-source data processing middleware for land monitoring within a web-based spatial data infrastructure for Siberia', *ISPRS International Journal of Geo-Information*, 2(3). doi: 10.3390/ijgi2030553.

Eberle, J. & Strobl, C. (2012) 'Web-based Geoprocessing and Workflow Creation for generating and providing Remote Sensing products', *Geomatica*, 66(1), pp. 13–26.

ECMA International (2017) 'ECMA-404 - The JSON Data Interchange Syntax.' ECMA International. Available at: https://www.ecma-international.org/publications/standards/Ecma-404.htm (Accessed: 8 October 2018).

Eklundh, L. & Jönsson, P. (2017) *TIMESAT 3.3 with seasonal trend decomposition and parallel processing - Software Manual*, *Lund and Malmo University, Sweden*.

Available at: http://web.nateko.lu.se/timesat/docs/TIMESAT33_SoftwareManual.pdf.

EOX IT Services GmbH (2018) *geotiff.js*. Available at: https://geotiffjs.github.io/ (Accessed: 7 October 2018).

ESA (2016) *Sentinel-3 family grows*. Available at: http://www.esa.int/Our_Activities/Observing_the_Earth/Copernicus/Sentinel-3/Sentinel-3_family_grows (Accessed: 6 March 2019).

ESA (2018) 'Procurement of EO Data Cube Facility Service: 2018-2022'. ESA.

ESA (2019) *Satellite missions*, *eoPortal Directory*. Available at: https://directory.eoportal.org/web/eoportal/satellite-missions/c-missions (Accessed: 6 March 2019).

Esch, T., Asamer, H., Boettcher, M., Brito, F., Hirner, A., Marconcini, M., Mathot, E., Metz, A., et al. (2016) 'Earth Observation-supported Service Platform for the Development and Provision of Thematic Information on the Built Environment–the TEP-Urban Project', *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLI-B8, pp. 1379–1384. doi: 10.5194/isprs-archives-XLI-B8-1379-2016.

European Commission (2018a) *Infrastructure for Spatial Information in the European Community*. Available at: https://inspire.ec.europa.eu/ (Accessed: 7 October 2018).

European Commission (2018b) *The DIAS: User-friendly Access to Copernicus Data and Information*. Brussels. Available at: https://www.copernicus.eu/sites/default/files/Copernicus_DIAS_Factsheet_June2018.pdf (Accessed: 2 March 2019).

European Commission (2019) *Copernicus Open Access Hub - Full Text Search*. Available at: https://scihub.copernicus.eu/twiki/do/view/SciHubUserGuide/FullTextSearch#Search_Keywords (Accessed: 1 April 2019).

Evangelidis, K., Ntouros, K., Makridis, S. & Papatheodorou, C. (2014) 'Geospatial services in the Cloud', *Computers & Geosciences*. Elsevier, 63, pp. 116–122. doi: 10.1016/j.cageo.2013.10.007.

Fallside, D. C. & Walsmley, P. (2014) *XML Schema Part 0: Primer Second Edition.* W3C. Available at: https://www.w3.org/TR/2004/REC-xmlschema-0-20041028/ (Accessed: 8 October 2018).

Farnaghi, M. & Mansourian, A. (2013) 'Disaster planning using automated composition of semantic OGC web services: A case study in sheltering', *Computers, Environment and Urban Systems*. Elsevier Ltd, 41, pp. 204–218. doi: 10.1016/j.compenvurbsys.2013.06.003.

Foerster, T., Brühl, A. & Schäffer, B. (2011) 'RESTful Web Processing Service', in *Proceedings of the 14th AGILE International Conference on Geographic Information Science*. Utrecht, The Netherlands, p. no pagination.

Forkel, M., Carvalhais, N., Verbesselt, J., Mahecha, M., Neigh, C. & Reichstein, M. (2013) 'Trend Change Detection in NDVI Time Series: Effects of Inter-Annual Variability and Methodology', *Remote Sensing*, 5(5), pp. 2113–2144. doi: 10.3390/rs5052113.

Friis-Christensen, A., Ostländer, N., Lutz, M. & Bernard, L. (2007) 'Designing Service Architectures for Distributed Geoprocessing: Challenges and Future Directions', *Transactions in GIS*, 11(6), pp. 799–818. doi: 10.1111/j.1467-9671.2007.01075.x.

Future Earth (2013) *Future Earth Initial Design: Report of the Transition Team*. Paris, France. Available at: http://www.futureearth.info/sites/default/files/Future-Earth-Design-Report_web.pdf.

Gallaher, D. & Grant, G. (2012) ‘Data Rods: High speed, time-series analysis of massive cryospheric data sets using pure object databases’, in *Proceedings of the 2012 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 5297–5300.

Gasperi, J. (2014) ‘Semantic Search Within Earth Observation Products Database Based on Automatic Tagging of Image Content’, *Proceedings of the 2014 Conference on Big Data from Space*, pp. 4–6.

Gebbert, S. & Pebesma, E. (2014) ‘TGRASS: A temporal GIS for field based environmental modeling’, *Environmental Modelling and Software*. Elsevier Ltd, 53, pp. 1–12. doi: 10.1016/j.envsoft.2013.11.001.

Geller, G. N. & Turner, W. (2007) ‘The model Web: A concept for ecological forecasting’, *Proceedings of the 2007 International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 2469–2472. doi: 10.1109/IGARSS.2007.4423343.

Gilles, M. (2006) *EO Application Profile for CSW 2.0*. Open Geospatial Consortium Inc.

Giuliani, G., Chatenoux, B., De Bono, A., Rodila, D., Richard, J.-P., Allenbach, K., Dao, H. & Peduzzi, P. (2017) ‘Building an Earth Observations Data Cube: lessons learned from the Swiss Data Cube (SDC) on generating Analysis Ready Data (ARD)’, *Big Earth Data*. Taylor & Francis, 1(1–2), pp. 100–117. doi: 10.1080/20964471.2017.1398903.

Giuliani, G., Nativi, S., Lehmann, A. & Ray, N. (2012) ‘WPS mediation: An approach to process geospatial data on different computing backends’, *Computers & Geosciences*. Elsevier, 47, pp. 20–33. doi: 10.1016/j.cageo.2011.10.009.

Gonçalves, P. (2014) ‘OGC OpenSearch Geo and Time Extensions’. Open Geospatial Consortium Inc.

Gonçalves, P. & Voges, U. (2016) ‘OGC OpenSearch Extension for Earth Observation’. Open Geospatial Consortium Inc., pp. 1–85.

Google (2017) *Google Earth Engine API - Landsat algorithms*. Available at: https://developers.google.com/earth-engine/landsat (Accessed: 8 October 2018).

Google (2018a) *Google Earth Engine API - Reference*. Available at: https://developers.google.com/earth-engine/api_docs#eealgorithmslandsatsimplecloudscore (Accessed: 8 October 2018).

Google (2018b) *SOAP, Representational state transfer - Google Trends*, *Google Trends*. Available at: https://trends.google.com/trends/explore?date=all&q=%2Fm%2F077dn,%2Fm%2F03nsxd (Accessed: 15 October 2018).

Gore, A. (1998) ‘The Digital Earth’, *Australian Surveyor*, 43(2), pp. 89–91. doi: 10.1080/00050348.1998.10558728.

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D. & Moore, R. (2017) ‘Google Earth Engine: Planetary-scale geospatial analysis for everyone’, *Remote Sensing of Environment*, 202, pp. 18–27. doi: 10.1016/j.rse.2017.06.031.

Granell, C., Havlik, D., Schade, S., Sabeur, Z., Delaney, C., Pielorz, J., Usländer, T., Mazzetti, P., et al. (2016) ‘Future Internet technologies for environmental applications’, *Environmental Modelling and Software*, 78, pp. 1–15. doi: 10.1016/j.envsoft.2015.12.015.

Guo, H., Liu, Z., Jiang, H., Wang, C., Liu, J. & Liang, D. (2017) ‘Big Earth Data: a new challenge and opportunity for Digital Earth’s development’, *International Journal of Digital Earth*. Taylor & Francis, 10(1), pp. 1–12. doi: 10.1080/17538947.2016.1264490.

Gutman, G. & Masek, J. G. (2012) 'Long-term time series of the Earth's land-surface observations from space', *International Journal of Remote Sensing*, 33(15), pp. 4700–4719. doi: 10.1080/01431161.2011.638341.

Hempelmann, N., Ehbrecht, C., Alvarez-Castro, C., Brockmann, P., Falk, W., Hoffmann, J., Kindermann, S., Koziol, B., et al. (2018) 'Web processing service for climate impact and extreme weather event analyses. Flyingpigeon (Version 1.0)', *Computers & Geosciences*, 110, pp. 65–72. doi: 10.1016/j.cageo.2017.10.004.

Hennig, S. & Belgiu, M. (2011) 'User-centric SDI: Addressing Users Requirements in Third- Generation SDI . The Example of Nature-SDIplus', *Geoforum Perspektiv*, 10(20), pp. 30–42. Available at: http://ojs.statsbiblioteket.dk/index.php/gfp/article/view/5820.

Herle, S. & Blankenbach, J. (2018) 'Enhancing the OGC WPS interface with GeoPipes support for real-time geoprocessing', *International Journal of Digital Earth*, 11(1), pp. 48–63. doi: 10.1080/17538947.2017.1319976.

Higgins, C. I., Williams, J., Leibovici, D. G., Simonis, I., Davis, M. J., Muldoon, C., Van Genuchten, P. & O 'hare, G. (2016) 'Citizen OBservatory WEB (COBWEB): A generic infrastructure platform to facilitate the collection of citizen science data for environmental monitoring', *International Journal of Spatial Data Infrastructures Research*, 11(January), pp. 20–48. doi: 10.2902/1725-0463.2016.11.art3.

Hofer, B. (2015) 'Uses of online geoprocessing technology in analyses and case studies: a systematic analysis of literature', *International Journal of Digital Earth*, 8(11), pp. 901–917. doi: 10.1080/17538947.2014.962632.

Hofer, B., Mäs, S., Brauner, J. & Bernard, L. (2017) 'Towards a knowledge base to support geoprocessing workflow development', *International Journal of Geographical Information Science*. Taylor & Francis, 31(4), pp. 694–716. doi: 10.1080/13658816.2016.1227441.

Holmes, C. (2017a) *A Cloud Native Geospatial Interoperability Sprint*, *Radiant Earth Insights*. Available at: https://medium.com/radiant-earth-insights/a-cloud-native-geospatial-interoperability-sprint-483d9c299595 (Accessed: 14 October 2018).

Holmes, C. (2017b) *Announcing the SpatioTemporal Asset Catalog (STAC) specification*, *Radiant Earth Insights*. Available at: https://medium.com/radiant-earth-insights/announcing-the-spatiotemporal-asset-catalog-stac-specification-1db58820b9cf (Accessed: 14 October 2018).

Holmes, C. (2018a) *Analysis Ready Data Defined*, *Planet Stories*. Available at: https://medium.com/planet-stories/analysis-ready-data-defined-5694f6f48815 (Accessed: 14 October 2018).

Holmes, C. (2018b) *The first Satellite Data Interoperability Workshop is happening next week!*, *Radiant Earth Insights*. Available at: https://medium.com/radiant-earth-insights/the-first-satellite-data-interoperability-workshop-is-happening-next-week-fae9539f81f9 (Accessed: 14 October 2018).

Hoyer, S. & Hamman, J. J. (2017) 'xarray: N-D labeled Arrays and Datasets in Python', *Journal of Open Research Software*, 5, pp. 1–6. doi: 10.5334/jors.148.

Hu, F., Xu, M., Yang, J., Liang, Y., Cui, K., Little, M. M., Lynnes, C. S., Duffy, D. Q., et al. (2018) 'Evaluating the Open Source Data Containers for Handling Big Geospatial Raster Data', *ISPRS International Journal of Geo-Information*, 7(4), p. 144. doi: 10.3390/ijgi7040144.

Huete, a, Didan, K., Miura, T., Rodriguez, E. ., Gao, X. & Ferreira, L. . (2002) 'Overview of the radiometric and biophysical performance of the MODIS vegetation indices', *Remote Sensing of Environment*, 83(1–2), pp. 195–213. doi: 10.1016/S0034-4257(02)00096-2.

Hüttich, C., Herold, M., Schmullius, C., Egorov, V. & Bartalev, S. A. (2007) 'Indicators of Northern Eurasia's land cover change trends from SPOT-VEGETATION time-series analysis 1998-2005', *International Journal of Remote Sensing*, 28(18), pp. 4199–4206. doi: 10.1080/01431160701442054.

International Organization for Standardization (2003) *ISO 19115:2003 Geographic information -- Metadata*. Geneva, Switzerland.

Janowicz, K. & Hitzler, P. (2017) 'Geospatial Semantic Web', in *International Encyclopedia of Geography: People, the Earth, Environment and Technology*. Oxford, UK: John Wiley & Sons, Ltd, pp. 1–6. doi: 10.1002/9781118786352.wbieg1158.

de Jesus, J., Walker, P., Grant, M. & Groom, S. (2012) 'WPS orchestration using the Taverna workbench: The eScience approach', *Computers & Geosciences*. Elsevier, 47, pp. 75–86. doi: 10.1016/j.cageo.2011.11.011.

De Jong, R., Verbesselt, J., Zeileis, A. & Schaepman, M. E. (2013) 'Shifts in global vegetation activity trends', *Remote Sensing*, 5(3), pp. 1117–1133. doi: 10.3390/rs5031117.

Jönsson, P. & Eklundh, L. (2004) 'TIMESAT—a program for analyzing time-series of satellite sensor data', *Computers & Geosciences*, 30(8), pp. 833–845. doi: 10.1016/j.cageo.2004.05.006.

Justice, C. ., Townshend, J. R. ., Vermote, E. ., Masuoka, E., Wolfe, R. ., Saleous, N., Roy, D. . & Morisette, J. . (2002) 'An overview of MODIS Land data processing and product status', *Remote Sensing of Environment*, 83(1–2), pp. 3–15. doi: 10.1016/S0034-4257(02)00084-6.

Kambatla, K., Kollias, G., Kumar, V. & Grama, A. (2014) 'Trends in big data analytics', *Journal of Parallel and Distributed Computing*, 74(7), pp. 2561–2573. doi: 10.1016/j.jpdc.2014.01.003.

Karantzalos, K., Bliziotis, D. & Karmas, A. (2015) 'A Scalable Geospatial Web Service for Near Real-Time, High-Resolution Land Cover Mapping', *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(10), pp. 4665–4674. doi: 10.1109/JSTARS.2015.2461556.

Kiehle, C., Greve, K. & Heier, C. (2007) 'Requirements for Next Generation Spatial Data Infrastructures-Standardized Web Based Geoprocessing and Web Service Orchestration', *Transactions in GIS*, 11(6), pp. 819–834. doi: 10.1111/j.1467-9671.2007.01076.x.

Kiemle, S., Molch, K., Schropp, S., Weiland, N. & Mikusch, E. (2016) 'Big Data Management in Earth Observation: The German satellite data archive at the German Aerospace Center', *IEEE Geoscience and Remote Sensing Magazine*, 4(3), pp. 51–58. doi: 10.1109/MGRS.2016.2541306.

Koswatte, S., McDougall, K. & Liu, X. (2015) 'SDI and crowdsourced spatial information management automation for disaster management', *Survey Review*, 47(344), pp. 307–315. doi: 10.1179/1752270615Y.0000000008.

Koubarakis, M., Bereta, K., Papadakis, G., Savva, D. & Stamoulis, G. (2017) 'Big, Linked Geospatial Data and Its Applications in Earth Observation', *IEEE Internet Computing*, 21(4), pp. 87–91. doi: 10.1109/MIC.2017.2911438.

Koubarakis, M., Sioutis, M., Garbis, G., Karpathiotakis, M., Kyzirakos, K., Nikolaou, C., Bereta, K., Vassos, S., et al. (2012) 'Building Virtual Earth Observatories Using Ontologies, Linked Geospatial Data and Knowledge Discovery Algorithms', in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pp. 932–949. doi: 10.1007/978-3-642-33615-7_34.

Kuenzer, C., Dech, S. & Wagner, W. (2015) 'Remote Sensing Time Series Revealing Land Surface Dynamics: Status Quo and the Pathway Ahead', in Kuenzer, C., Dech, S., and Wagner, W. (eds) *Remote Sensing Time Series*. Remote Sen. Cham: Springer, pp. 1–24. doi: 10.1007/978-3-319-15967-6_1.

Kuenzer, C., Ottinger, M., Wegmann, M., Guo, H., Wang, C., Zhang, J., Dech, S. & Wikelski, M. (2014) 'Earth observation satellite sensors for biodiversity monitoring: potentials and bottlenecks', *International Journal of Remote Sensing*, 35(18), pp. 6599–6647. doi: 10.1080/01431161.2014.964349.

Lankester, T. H. G. (2009) 'OpenGIS Web Map Services - Profile for EO Products'. Open Geospatial Consortium Inc.

Lasaponara, R. & Lanorte, A. (2012) 'Satellite time-series analysis', *International Journal of Remote Sensing*, 33(15), pp. 4649–4652. doi: 10.1080/01431161.2011.638342.

Lee, J.-G. & Kang, M. (2015) 'Geospatial Big Data: Challenges and Opportunities', *Big Data Research*. Elsevier Inc., 2(2), pp. 74–81. doi: 10.1016/j.bdr.2015.01.003.

Li, S., Dragicevic, S., Castro, F. A., Sester, M., Winter, S., Coltekin, A., Pettit, C., Jiang, B., et al. (2016) 'Geospatial big data handling theory and methods: A review and research challenges', *ISPRS Journal of Photogrammetry and Remote Sensing*. International Society for Photogrammetry and Remote Sensing, Inc. (ISPRS), 115, pp. 119–133. doi: 10.1016/j.isprsjprs.2015.10.012.

Liakos, P., Koltsida, P., Kakaletris, G., Baumann, P., Ioannidis, Y. & Delis, A. (2015) 'A Distributed Infrastructure for Earth-Science Big Data Retrieval', *International Journal of Cooperative Information Systems*, 24(02). doi: 10.1142/S0218843015500021.

Lopez-Pellicer, F. J., Rentería-Agualimpia, W., Béjar, R., Muro-Medrano, P. R. & Zarazaga-Soria, F. J. (2012) 'Availability of the OGC geoprocessing standard: March 2011 reality check', *Computers & Geosciences*, 47, pp. 13–19. doi: 10.1016/j.cageo.2011.10.023.

Lott, N. (2006) *Global Surface Summary of Day*, *National Climatic Data Center, Asheville, NC*. Available at: http://www.ncdc.noaa.gov/cgi-bin/res40.pl?page=gsod.html (Accessed: 1 March 2013).

Lott, N., Vose, R., Greco, S. A. Del, Ross, T. F., Worley, S. & Comeaux, J. (2008) 'The Integrated Surface Database: Partnerships and Progress', in *Proceedings of the 24th Conference in Institutional Information Processing System (IIPS)*. New Orleans, USA: American Meteorological Society.

Ma, Y., Wu, H., Wang, L., Huang, B., Ranjan, R., Zomaya, A. & Jie, W. (2015) 'Remote sensing big data computing: Challenges and opportunities', *Future Generation Computer Systems*. Elsevier B.V., 51, pp. 47–60. doi: 10.1016/j.future.2014.10.029.

MacKenzie, C. M., Laskey, K., McCabe, F., Brown, P. F., Metz, R. & Hamilton, B. A. (2006) 'Reference Model for Service Oriented Architecture 1.0', pp. 1–31. Available at: https://docs.oasis-open.org/soa-rm/v1.0/soa-rm.html.

Magan, V. (2016) *Two New Sentinel 2 Satellites Coming Up from Airbus Defence and Space*, *Satellite Today - Via Satellite*. Available at: https://www.satellitetoday.com/government-military/2016/01/26/two-new-sentinel-2-satellites-coming-up-from-airbus-defence-and-space/ (Accessed: 6 March 2019).

Martinis, S., Kersten, J. & Twele, A. (2015) 'A fully automated TerraSAR-X based flood service', *ISPRS Journal of Photogrammetry and Remote Sensing*, 104, pp. 203–212. doi: 10.1016/j.isprsjprs.2014.07.014.

Mazzetti, P. & Nativi, S. (2012) 'Multidisciplinary Interoperability for Earth Observations: Some Architectural Issues', *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5(3), pp. 1054–1059..

Mazzetti, P., Nativi, S. & Caron, J. (2009) ‘RESTful implementation of geospatial services for Earth and Space Science applications’, *International Journal of Digital Earth*, 2, pp. 40–61. doi: 10.1080/17538940902866153.

McKinney, W. (2011) ‘pandas: a Foundational Python Library for Data Analysis and Statistics’, *Python for High Performance and Scientific Computing*, (December), pp. 1–9.

Meng, X., Bian, F. & Xie, Y. (2009) ‘Research and Realization of Geospatial Information Service Orchestration Based on BPEL’, in *International Conference on Environmental Science and Information Application Technology*. Wuhan: IEEE, pp. 642–645. doi: 10.1109/ESIAT.2009.287.

Meng, X., Xie, Y. & Bian, F. (2010) ‘Distributed Geospatial Analysis through Web Processing Service: A Case Study of Earthquake Disaster Assessment’, *Journal of Software*, 5(6), pp. 671–679. doi: 10.4304/jsw.5.6.671-679.

Menne, M. J., Durre, I., Vose, R. S., Gleason, B. E. & Houston, T. G. (2012) ‘An Overview of the Global Historical Climatology Network-Daily Database’, *Journal of Atmospheric and Oceanic Technology*, 29(7), pp. 897–910. doi: 10.1175/JTECH-D-11-00103.1.

Miura, S. H. (2016) ‘Earth Observation data access interoperability implementation among space agencies’, in *Proceedings of the 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, pp. 3621–3623. doi: 10.1109/IGARSS.2016.7729938.

Müller, M., Bernard, L. & Brauner, J. (2010) ‘Moving Code in Spatial Data Infrastructures - Web Service Based Deployment of Geoprocessing Algorithms’, *Transactions in GIS*, 14, pp. 101–118. doi: 10.1111/j.1467-9671.2010.01205.x.

Müller, M., Bernard, L. & Kadner, D. (2013) ‘Moving code – Sharing geoprocessing logic on the Web’, *ISPRS Journal of Photogrammetry and Remote Sensing*, 83, pp. 193–203. doi: 10.1016/j.isprsjprs.2013.02.011.

NASA (2019a) *Earthdata - CMR Search - API Documentation*. Available at: https://cmr.earthdata.nasa.gov/search/site/docs/search/api.html (Accessed: 1 April 2019).

NASA (2019b) *MODIS Web*. Available at: https://modis.gsfc.nasa.gov/data/ (Accessed: 6 March 2019).

NASA LPDAAC (2014) *NASA LPDAAC - MODIS Overview*. Available at: https://lpdaac.usgs.gov/modis/overview.asp (Accessed: 8 October 2018).

Nash, E., Bobert, J., Wenkel, K., Mirschel, W. & Wieland, R. (2007) ‘Geocomputing Made Simple: Service-Chain Based Automated Geoprocessing for Precision Agriculture’, in *Proceedings of the GeoComputation 2007*. Maynooth, Ireland.

Nativi, S. & Bigagli, L. (2009) ‘Discovery , Mediation , and Access Services for Earth Observation Data’, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2(4), pp. 233–240. doi: 10.1109/JSTARS.2009.2028584.

Nativi, S., Craglia, M. & Pearlman, J. (2012) ‘The Brokering Approach for Multidisciplinary Interoperability: A Position Paper’, *International Journal of Spatial Data Infrastructures Research*, 7, pp. 1–15. doi: 10.2902/1725-0463.2012.07.art1.

Nativi, S., Craglia, M. & Pearlman, J. (2013) ‘Earth science infrastructures interoperability: The brokering approach’, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 6(3), pp. 1118–1129. doi: 10.1109/JSTARS.2013.2243113.

Nativi, S., Mazzetti, P., Craglia, M. & Pirrone, N. (2014) ‘The GEOSS solution for enabling data interoperability and integrative research’, *Environmental Science and Pollution Research*, 21(6), pp. 4177–4192. doi: 10.1007/s11356-013-2264-y.

Nativi, S., Mazzetti, P. & Geller, G. N. (2013) 'Environmental model access and interoperability: The GEO Model Web initiative', *Environmental Modelling & Software*. Elsevier Ltd, 39, pp. 214–228. doi: 10.1016/j.envsoft.2012.03.007.

Nativi, S., Mazzetti, P., Santoro, M., Papeschi, F., Craglia, M. & Ochiai, O. (2015) 'Big Data challenges in building the Global Earth Observation System of Systems', *Environmental Modelling & Software*. Elsevier Ltd, 68, pp. 1–26. doi: 10.1016/j.envsoft.2015.01.017.

Nebert, D., Whiteside, A. & Vretanos, P. P. (2007) *OpenGIS Catalogue Services Specification*. Open Geospatial Consortium Inc.

Neteler, M., Bowman, M. H., Landa, M. & Metz, M. (2012) 'GRASS GIS: A multi-purpose open source GIS', *Environmental Modelling and Software*. Elsevier Ltd, 31, pp. 124–130. doi: 10.1016/j.envsoft.2011.11.014.

OGC (2018a) *About OGC*. Available at: http://www.opengeospatial.org/about (Accessed: 7 October 2018).

OGC (2018b) *OGC Innovation Program*. Available at: http://www.opengeospatial.org/ogc/programs/ip (Accessed: 14 October 2018).

Ooms, J. (2014) 'The OpenCPU System: Towards a Universal Interface for Scientific Computing through Separation of Concerns', *eprint arXiv*, (2000), pp. 1–23.

Overdick, H. (2007) 'The Resource-Oriented Architecture', in *Proceedings of the 2007 IEEE Congress on Services (Services 2007)*. IEEE, pp. 340–347. doi: 10.1109/SERVICES.2007.66.

Pagani, G. A. & Trani, L. (2018) 'Data cube and cloud resources as platform for seamless geospatial computation', in *Proceedings of the 15th ACM International Conference on Computing Frontiers - CF '18*. New York, New York, USA: ACM Press, pp. 293–298. doi: 10.1145/3203217.3205861.

Pettorelli, N., Laurance, W. F., O'Brien, T. G., Wegmann, M., Nagendra, H. & Turner, W. (2014) 'Satellite remote sensing for applied ecologists: Opportunities and challenges', *Journal of Applied Ecology*, 51(4), pp. 839–848. doi: 10.1111/1365-2664.12261.

Planthaber, G., Stonebraker, M. & Frew, J. (2012) 'EarthDB: scalable analysis of MODIS data using SciDB', *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data - BigSpatial '12*, pp. 11–19. doi: 10.1145/2447481.2447483.

Plewe, B. (2007) 'Web Cartography in the United States', *Cartography and Geographic Information Science*, 34(2), pp. 133–136. doi: 10.1559/152304007781002235.

Previtali, M. & Latre, M. Á. (2018) 'A brokered Virtual Hub approach for the generation of web applications based on historical maps', *Applied Geomatics*. Applied Geomatics, 10(4), pp. 453–472. doi: 10.1007/s12518-018-0235-1.

Radiant Earth (2018) *SpatioTemporal Asset Catalog specification*. Available at: https://github.com/radiantearth/stac-spec (Accessed: 14 October 2018).

Reck, C., Storch, T., Holzwarth, S. & Schmidt, M. (2019) 'Online Data Access and Big Data Processing in the German Copernicus Data and Exploitation Environment (CODE-DE)', in Soille, P., Loekken, S., and Albani, S. (eds) *Proceedings of the 2019 conference on Big Data from Space*. Munich, Germany: Publications Office of the European Union, pp. 153–156.

Reid, W. V, Chen, D., Goldfarb, L., Hackmann, H., Lee, Y. T., Mokhele, K., Ostrom, E., Raivio, K., et al. (2010) 'Earth System Science for Global Sustainability: Grand Challenges', *Science*, 330(6006), pp. 916–917. doi: 10.1126/science.1196263.

Rieke, M., Bigagli, L., Herle, S., Jirka, S., Kotsev, A., Liebig, T., Malewski, C., Paschke, T., et al. (2018) 'Geospatial IoT—The Need for Event-Driven Architectures in

Contemporary Spatial Data Infrastructures', *ISPRS International Journal of Geo-Information*, 7(10), p. 385. doi: 10.3390/ijgi7100385.

Roy, D. P., Wulder, M. A., Loveland, T. R., C.E., W., Allen, R. G., Anderson, M. C., Helder, D., Irons, J. R., et al. (2014) 'Landsat-8: Science and product vision for terrestrial global change research', *Remote Sensing of Environment*. Elsevier B.V., 145, pp. 154–172. doi: 10.1016/j.rse.2014.02.001.

Santos, W. (2017) *Which API Types and Architectural Styles are Most Used?*, *ProgrammableWeb.com*. Available at: https://www.programmableweb.com/news/which-api-types-and-architectural-styles-are-most-used/research/2017/11/26 (Accessed: 28 February 2019).

Schade, S. (2015) 'Big Data breaking barriers - first steps on a long trail', *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XL-7/W3(MAY), pp. 691–697. doi: 10.5194/isprsarchives-XL-7-W3-691-2015.

Schaeffer, B., Baranski, B., Foerster, T. & Brauner, J. (2012) 'A Service-Oriented Framework for Real-time and Distributed Geoprocessing', in Bocher, E. and Neteler, M. (eds) *Geospatial Free and Open Source Software in the 21st Century*. Berlin, Germany: Springer Berlin Heidelberg, pp. 3–20.

Schramm, M., Pebesma, E., Wagner, W., Verbesselt, J., Dries, J., Briese, C., Jacob, A., Mohr, M., et al. (2019) 'OpenEO - A standardised connection to an between Earth Observation service providers', in Soille, P., Loekken, S., and Albani, S. (eds) *Proceedings of the 2019 conference on Big Data from Space*. Munich, Germany: Publications Office of the European Union, pp. 229–232.

Schut, P. (2007) *OpenGIS Web Processing Service*, *Open Geospatial Consortium*. Open Geospatial Consortium Inc.

See, L., Mooney, P., Foody, G., Bastin, L., Comber, A., Estima, J., Fritz, S., Kerle, N., et al. (2016) 'Crowdsourcing, Citizen Science or Volunteered Geographic Information? The Current State of Crowdsourced Geographic Information', *ISPRS International Journal of Geo-Information*, 5(5), p. 55. doi: 10.3390/ijgi5050055.

Shao, Y., Di, L., Bai, Y., Wang, H. & Yang, C. (2013) 'Federated Catalogue for Discovering Earth Observation Data Konzept für einen Zentralkatalog für Fernerkundungsdaten', *Photogrammetrie - Fernerkundung - Geoinformation*, 2013(1), pp. 43–52. doi: 10.1127/1432-8364/2013/0157.

Shelestov, A., Lavreniuk, M., Kussul, N., Novikov, A. & Skakun, S. (2017) 'Exploring Google Earth Engine Platform for Big Data Processing: Classification of Multi-Temporal Satellite Imagery for Crop Mapping', *Frontiers in Earth Science*, 5. doi: 10.3389/feart.2017.00017.

Shibasaki, R. & Pearlman, J. (2008) 'Guest Editorial', *IEEE Systems Journal*, 2(3), pp. 302–303. doi: 10.1109/JSYST.2008.928859.

Simonis, I. (2018) *Testbed 14: The Most Complex OGC Testbed Ever?* Available at: http://www.opengeospatial.org/blog/2773 (Accessed: 14 October 2018).

Sinergise (2019a) *Sentinel Hub - OGC API*. Available at: https://www.sentinel-hub.com/develop/documentation/api/ogc_api (Accessed: 1 April 2019).

Sinergise (2019b) *Sentinel Hub - Statistical info API documentation*. Available at: https://www.sentinel-hub.com/develop/documentation/api/fis-request (Accessed: 1 April 2019).

Siqueira, A. D. A., Lewis, A., Thankappan, M., Szantoi, Z., Goryl, P., Tadono, T., Rosenqvist, A., Ross, J., et al. (2019) 'CEOS Analysis Ready Data for Land - Supporting the Earth Observation community to get the best value from the big data wave from Space', in Soille, P., Loekken, S., and Albani, S. (eds) *Proceedings of*

the 2019 conference on Big Data from Space. Munich, Germany: Publications Office of the European Union, pp. 185–188.

Smith, J. P., Hunter, T. S., Clites, A. H., Stow, C. A., Slawecki, T., Muhr, G. C. & Gronewold, A. D. (2016) 'An expandable web-based platform for visually analyzing basin-scale hydro-climate time series data', *Environmental Modelling & Software*. Elsevier Ltd, 78, pp. 97–105. doi: 10.1016/j.envsoft.2015.12.005.

Soille, P., Burger, A., De Marchi, D., Kempeneers, P., Rodriguez, D., Syrris, V. & Vasilev, V. (2018) 'A versatile data-intensive computing platform for information retrieval from big geospatial data', *Future Generation Computer Systems*. Elsevier B.V., 81, pp. 30–40. doi: 10.1016/j.future.2017.11.007.

Sorg, J. & Kunkel, R. (2015) 'Conception and Implementation of an OGC-Compliant Sensor Observation Service for a Standardized Access to Raster Data', *ISPRS International Journal of Geo-Information*, 4(3), pp. 1076–1096. doi: 10.3390/ijgi4031076.

Strobl, P., Baumann, P., Lewis, A., Szantoi, Z., Killough, B., Purss, M., Craglia, M., Nativi, S., et al. (2017) 'The Six Face of Data Cube', in Soille, P. and Marchetti, P. G. (eds) *Proceedings of the 2017 conference on Big Data from Space*. Toulouse, France: Publications Office of the European Union, pp. 32–35.

Sudmanns, M., Tiede, D., Lang, S. & Baraldi, A. (2018) 'Semantic and syntactic interoperability in online processing of big Earth observation data', *International Journal of Digital Earth*. Taylor & Francis, 11(1), pp. 95–112. doi: 10.1080/17538947.2017.1332112.

Sun, A. (2013) 'Enabling collaborative decision-making in watershed management using cloud-computing services', *Environmental Modelling & Software*. Elsevier Ltd, 41, pp. 93–97. doi: 10.1016/j.envsoft.2012.11.008.

Svobodova, L. (1985) 'Client/Server Model of Distributed Processing', in Heger, D., Krüger, G., Spaniol, O., and Zorn, W. (eds). Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 485–498.

Swain, N. R., Christensen, S. D., Snow, A. D., Dolder, H., Espinoza-Dávalos, G., Goharian, E., Jones, N. L., Nelson, E. J., et al. (2016) 'A new open source platform for lowering the barrier for environmental web app development', *Environmental Modelling and Software*, 85, pp. 11–26. doi: 10.1016/j.envsoft.2016.08.003.

Szuba, M., Ameri, P., Grabowski, U., Meyer, J. & Streit, A. (2016) 'A Distributed System for Storing and Processing Data from Earth-Observing Satellites: System Design and Performance Evaluation of the Visualisation Tool', *Proceedings - 2016 16th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, CCGrid 2016*, pp. 169–174. doi: 10.1109/CCGrid.2016.19.

Tan, C. B., McMeekin, D. A., West, G. & Moncrieff, S. (2017) 'CIAO-WPS - Utilizing semantic web (web 3.0) techniques to assist in the automatic orchestration of geospatial processes and datasets', *Communications in Computer and Information Science*, 741(Gistam), pp. 32–48. doi: 10.1007/978-3-319-62618-5_3.

Tan, X., Di, L., Deng, M., Fu, J., Shao, G., Gao, M., Sun, Z., Ye, X., et al. (2015) 'Building an Elastic Parallel OGC Web Processing Service on a Cloud-Based Cluster: A Case Study of Remote Sensing Data Processing Service', *Sustainability*, 7(10), pp. 14245–14258. doi: 10.3390/su71014245.

Tan, Z., Yue, P. & Gong, J. (2017) 'An Array Database Approach for Earth Observation Data Management and Processing', *ISPRS International Journal of Geo-Information*, 6(12), p. 220. doi: 10.3390/ijgi6070220.

Tao, Y., Wang, X., Xu, X. & Liu, G. (2018) 'Container-as-a-service architecture for business workflow', *International Journal of Simulation and Process Modelling*, 13(2), p. 102. doi: 10.1504/IJSPM.2018.091692.

La Torre, G., Cavallo, M., D'Amico, V., Monteleone, S. & Catania, V. (2017) 'A Context-Aware Solution to Improve Web Service Discovery and User-Service Interaction', *Proceedings - 13th IEEE International Conference on Ubiquitous Intelligence and Computing,* pp. 180–187. doi: 10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.0047.

Trakas, A. (2010) 'The importance of standards bodies in EU funded projects', *CEN/TC 287 Workshop*. Malta: Open Geospatial Consortium Inc.

Traore, A., Ciais, P., Vuichard, N., McBean, N., Dardel, C., Poulter, B., Piao, S., Fisher, J., et al. (2014) '1982–2010 Trends of Light Use Efficiency and Inherent Water Use Efficiency in African vegetation: Sensitivity to Climate and Atmospheric CO2 Concentrations', *Remote Sensing*, 6(9), pp. 8923–8944. doi: 10.3390/rs6098923.

Truckenbrodt, J., Bongard, J., Schmullius, C. & Weise, K. (2018) 'Hyper-temporal Water Body Dynamics Mapping using Sentinel-1 Time Series Clustering', in *Proceedings of the 2018 ESA Mapping Water Bodies from Space*. Frascati, Italy.

Tsinaraki, C. & Schade, S. (2016) 'Big Data – a step change for SDI?', *International Journal of Spatial Data Infrastructures Research*, 11(2010), pp. 9–19. doi: 10.2902/1725-0463.2016.11.art2.

Tsou, M.-H. (2011) 'Revisiting Web Cartography in the United States: the Rise of User-Centered Design', *Cartography and Geographic Information Science*, 38(3), pp. 250–257. doi: 10.1559/15230406382250.

Tucker, C. J. (1979) 'Red and Photographic Infrared Linear Combinations for Monitoring Vegetation', *Remote Sensing of Environment*, 8, pp. 127–150.

Turner, W., Rondinini, C., Pettorelli, N., Mora, B., Leidner, A. K., Szantoi, Z., Buchanan, G., Dech, S., et al. (2015) 'Free and open-access satellite data are key to biodiversity conservation', *Biological Conservation*. Elsevier Ltd, 182, pp. 173–176. doi: 10.1016/j.biocon.2014.11.048.

U. S. Government (2014) 'National plan for civil Earth Observations'. Washington, D.C. Available at: https://obamawhitehouse.archives.gov/sites/default/files/microsites/ostp/NSTC/2014_national_plan_for_civil_earth_observations.pdf (Accessed: 14 October 2018).

Urban, M., Forkel, M., Eberle, J., Hüttich, C., Schmullius, C. & Herold, M. (2014) 'Pan-Arctic Climate and Land Cover Trends Derived from Multi-Variate and Multi-Scale Analyses (1981–2012)', *Remote Sensing*, 6(3), pp. 2296–2316. doi: 10.3390/rs6032296.

USGS (2018) *USGS ESPA On Demand Product Guide*. Available at: https://www.usgs.gov/media/files/eros-science-processing-architecture-demand-interface-user-guide.

USGS (2019a) *EarthExplorer Service Documentation - JSON API 1.4.0*. Available at: https://earthexplorer.usgs.gov/inventory/documentation/json-api?version=1.4.0 (Accessed: 1 April 2019).

USGS (2019b) *Landsat Missions: Landsat Collection 1*. Available at: https://www.usgs.gov/land-resources/nli/landsat/landsat-collection-1 (Accessed: 6 March 2019).

USGS (2019c) *Landsat Satellite Missions*. Available at: https://www.usgs.gov/land-resources/nli/landsat/landsat-satellite-missions (Accessed: 6 March 2019).

Veenendaal, B., Brovelli, M. A. & Li, S. (2017) 'Review of Web Mapping: Eras, Trends and Directions', *ISPRS International Journal of Geo-Information*, 6(10), p. 317. doi: 10.3390/ijgi6100317.

Veenendaal, B., Brovelli, M. A. & Wu, L. (2016) 'Cloud/web mapping and geoprocessing services - Intelligently linking geoinformation', *ISPRS Journal of Photogrammetry*

and Remote Sensing. International Society for Photogrammetry and Remote Sensing, Inc. (ISPRS), 114, pp. 243–244. doi: 10.1016/j.isprsjprs.2016.03.005.

Verbesselt, J., Hyndman, R., Newnham, G. & Culvenor, D. (2010a) 'Detecting trend and seasonal changes in satellite image time series', *Remote Sensing of Environment*. Elsevier B.V., 114(1), pp. 106–115. doi: 10.1016/j.rse.2009.08.014.

Verbesselt, J., Hyndman, R., Zeileis, A. & Culvenor, D. (2010b) 'Phenological change detection while accounting for abrupt and gradual trends in satellite image time series', *Remote Sensing of Environment*. Elsevier B.V., 114(12), pp. 2970–2980. doi: 10.1016/j.rse.2010.08.003.

Verbesselt, J., Zeileis, A. & Herold, M. (2012) 'Near real-time disturbance detection using satellite image time series', *Remote Sensing of Environment*. Elsevier Inc., 123, pp. 98–108. doi: 10.1016/j.rse.2012.02.022.

Vinhas, L., Ribeiro De Queiroz, G., Ferreira, K. R. & Camara, G. (2016) 'Web Services for Big Earth Observation Data', in *Proceedings of Geoinfo 2016*. Campos do Jordao, Brazil, pp. 166–177.

Vitolo, C., Elkhatib, Y., Reusser, D., Macleod, C. J. a. & Buytaert, W. (2015) 'Web technologies for environmental Big Data', *Environmental Modelling & Software*. Elsevier Ltd, 63, pp. 185–198. doi: 10.1016/j.envsoft.2014.10.007.

Wagemann, J., Clements, O., Marco Figuera, R., Rossi, A. P. & Mantovani, S. (2018) 'Geospatial web services pave new ways for server-based on-demand access and processing of Big Earth Data', *International Journal of Digital Earth*, 11(1), pp. 7–25. doi: 10.1080/17538947.2017.1351583.

Wolfe, R. E., Roy, D. P. & Vermote, E. (1998) 'MODIS land data storage, gridding, and compositing methodology: Level 2 grid', *IEEE Transactions on Geoscience and Remote Sensing*, 36(4), pp. 1324–1338. doi: 10.1109/36.701082.

Woodcock, C. E., Allen, R. G., Anderson, M., Belward, A., Bindschadler, R., Cohen, W. B., ... & Wynne, R. (2008) 'Free Access to Landsat Imagery', *Science*, 320(May), pp. 1011–1012. doi: 10.1126/science.320.5879.1011a.

Wosniok, C., Bensmann, F., Wössner, R., Kohlus, J. & Roosmann, R. (2014) 'Enriching the Web Processing Service', in *Proceedings of EGU Gerneral Assemlby 2014*. Vienna, Austria.

Wu, H., You, L., Gui, Z., Gao, S., Li, Z. & Yu, J. (2014) 'FAST: A fully asynchronous and status-tracking pattern for geoprocessing services orchestration', *Computers & Geosciences*. Elsevier, 70, pp. 213–228. doi: 10.1016/j.cageo.2014.06.005.

Wulder, M. A., Masek, J. G., Cohen, W. B., Loveland, T. R. & Woodcock, C. E. (2012) 'Opening the archive: How free data has enabled the science and monitoring promise of Landsat', *Remote Sensing of Environment*. Elsevier B.V., 122, pp. 2–10. doi: 10.1016/j.rse.2012.01.010.

Wyborn, L. & Evans, B. J. K. (2015) 'Integrating "Big" geoscience data into the petascale national environmental research interoperability platform (NERDIP): Successes and unforeseen challenges', in *Proceedings of the 2015 IEEE International Conference on Big Data (Big Data)*. IEEE, pp. 2005–2009. doi: 10.1109/BigData.2015.7363981.

Xiao, F., Shea, G. Y. K., Cao, J. & Web, S. (2016) 'Decentralized Orchestration of Composite Ogc Web Processing', in *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pp. 7–9. doi: 10.5194/isprs-annals-IV-4-W1-125-2016.

Xie, J. & Li, G. (2016) 'Implementing next-generation national Earth Observation data infrastructure to integrate distributed big Earth Observation data', in *Proceedings of the 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, pp. 194–197. doi: 10.1109/IGARSS.2016.7729042.

Xie, Y., Sha, Z. & Yu, M. (2008) ‘Remote sensing imagery in vegetation mapping: a review’, *Journal of Plant Ecology*, 1(1), pp. 9–23. doi: 10.1093/jpe/rtm005.

Xiong, X., Chiang, K., Sun, J., Barnes, W. L., Guenther, B. & Salomonson, V. V. (2009) ‘NASA EOS Terra and Aqua MODIS on-orbit performance’, *Advances in Space Research*. COSPAR, 43(3), pp. 413–422. doi: 10.1016/j.asr.2008.04.008.

Yang, C., Huang, Q., Li, Z., Liu, K. & Hu, F. (2017a) ‘Big Data and cloud computing: innovation opportunities and challenges’, *International Journal of Digital Earth*, 10(1), pp. 13–53. doi: 10.1080/17538947.2016.1239771.

Yang, C., Yu, M., Hu, F., Jiang, Y. & Li, Y. (2017b) ‘Utilizing Cloud Computing to address big geospatial data challenges’, *Computers, Environment and Urban Systems*, 61, pp. 120–128. doi: 10.1016/j.compenvurbsys.2016.10.010.

Yin, D., Liu, Y., Padmanabhan, A., Terstriep, J., Rush, J. & Wang, S. (2017) ‘A CyberGIS-Jupyter Framework for Geospatial Analytics at Scale’, in *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact - PEARC17*. New York, New York, USA: ACM Press, pp. 1–8. doi: 10.1145/3093338.3093378.

Yue, P., Zhang, C., Zhang, M., Zhai, X. & Jiang, L. (2015) ‘An SDI Approach for Big Data Analytics: The Case on Sensor Web Event Detection and Geoprocessing Workflow’, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(10), pp. 4720–4728. doi: 10.1109/JSTARS.2015.2494610.

Yutzler, J. (2017) ‘OGC GeoPackage Encoding Standard’. Open Geospatial Consortium Inc .

Zhang, Y., Wu, W., Wang, Q. & Su, F. (2017) ‘A Geo-Event-Based Geospatial Information Service: A Case Study of Typhoon Hazard’, *Sustainability*, 9(4), p. 534. doi: 10.3390/su9040534.

Zhao, P., Foerster, T. & Yue, P. (2012) ‘The Geoprocessing Web’, *Computers & Geosciences*, 47, pp. 3–12. doi: 10.1016/j.cageo.2012.04.021.

Zhu, Z. (2017) ‘Change detection using landsat time series: A review of frequencies, preprocessing, algorithms, and applications’, *ISPRS Journal of Photogrammetry and Remote Sensing*, 130, pp. 370–384. doi: 10.1016/j.isprsjprs.2017.06.013.

Appendix A

A.1 Discovery of EO time-series data

Within this section, the service endpoints (URLs) and parameters available to discover EO collections and EO scenes are given for the data providers that have been reviewed in Section 4.1. The references of the tables contain information about parameters presented here and further properties of the services.

ESA/Copernicus Open Access Hub (Sentinel) – OpenSearch

The following service endpoint can be used to query the catalogue:

- Open Access Hub: https://scihub.copernicus.eu/dhus/search

Table A.1: List of search parameters for Sentinel data (European Commission 2019).

Search Keyword	Syntax and Examples
q	Full text search
platformname	Sentinel-1, Sentinel-2, Sentinel-3
beginposition **endposition**	<keyword>:[<timestamp> TO <timestamp>]
footprint	footprint:"intersects(<*geographic type*>)"
orbitnumber **lastorbitnumber**	orbitnumber:<orbitnumber> orbitnumber:[<orbitnumber> TO <orbitnumber>]
orbitdirection	Ascending, Descending
polarisationmode	HH, VV, HV, VH, HH HV, VV VH
producttype	SLC, GRD, OCN, S2MSI1C
relativeorbitnumber	relativeorbitnumber:<relativeorbitnumber>
sensoroperationalmode	SM, IW, EW
cloudcoverpercentage	Possible values from 0 TO 100

USGS Earth Explorer (Landsat, MODIS, Sentinel-2)–self-developed REST API

Dependent on the function, three service endpoints are available for user login, discovery of EO collections, and discovery of EO scenes:

- Login: https://earthexplorer.usgs.gov/inventory/json/v/1.4.0/login
- Collections: https://earthexplorer.usgs.gov/inventory/json/v/1.4.0/datasets
- Satellite scenes: https://earthexplorer.usgs.gov/inventory/json/v/1.4.0/search

Table A.2: List of search parameters for collections within USGS Earth Explorer (USGS 2019a).

Search Keyword	Syntax and Examples
datasetName	Filter on dataset name (with wildcards)
spatialFilter	Spatial filter using bounding box values
temporalFilter	Temporal filter using start/end date

Table A.3: List of search parameters for scenes within USGS Earth Explorer (USGS 2019a).

Search Keyword	Syntax and Examples
datasetName*	Identifies the dataset
spatialFilter	Spatial filter using bounding box values
temporalFilter	Temporal filter using start/end date
months	Used to limit results to specific months
minCloudCover **maxCloudCover**	Used to limit results by minimum / maximum cloud cover
additionalCriteria	Used to filter results based on dataset specific metadata fields

* Mandatory

NASA CMR (Landsat, MODIS) – self-developed REST API

The following service endpoints can be requested:

- List dataset collections: https://cmr.earthdata.nasa.gov/search/collections.json
- List scenes: https://cmr.earthdata.nasa.gov/search/granules.json

Table A.4: List of search parameters for collections within NASA CMR (NASA 2019a).

Search Keyword	Syntax and Examples
entry_title	Filter on dataset collection name (with wildcards)
keyword	Filter dataset collections according to keywords
bounding_box	Spatial filter using bounding box values
temporal	Temporal filter using start/end date

Table A.5: List of search parameters for satellite scenes within NASA CMR (NASA 2019a).

Search Keyword	Syntax and Examples
concept_id*	Identifies the dataset
bounding_box	Spatial filter using bounding box values
temporal	Temporal filter using start/end date
cloud_cover	Limit results by minimum, maximum cloud cover
attribute	Filter results based on dataset specific metadata fields

* Mandatory

Google Earth Engine (Sentinel, Landsat, MODIS) – Python API

```
01 # filter collection by point geometry
02 geom_azraq = ee.Geometry.Point(36.83075, 31.79115)
03 sentinel1 = ee.ImageCollection('COPERNICUS/S1_GRD')
04 sentinel1 = sentinel1.filterBounds(geom)
05
06 # filter collection by polarization
07 sentinel1 = sentinel1.filter(
       ee.Filter.listContains('transmitterReceiverPolarisation', 'VV'))
08
09 # filter collection by orbit pass (ASCENDING, DESCENDING)
10 vvAscending = sentinel1.filter(
       ee.Filter.eq('orbitProperties_pass', 'ASCENDING'))
11 response = vvAscending.getInfo()
```

Listing A.1: Filtering Sentinel-1 Collection by point geometry, VV polarization and descending orbit direction using the Python-based Earth Engine library.

Sinergise Sentinel-Hub (Sentinel, Landsat) – OGC WFS

The following service endpoint can be requested for the OGC Web Feature Service (INSTANCE_ID need to be replaced by a commercial service key):

- Satellite scenes: http://services.sentinel-hub.com/ogc/wfs/{INSTANCE_ID}

Table A.6: List of parameters to use the Sentinel-Hub WFS service for scene search (Sinergise 2019a).

Keyword	Value(s)
service	WFS
version	2.0.0
request	GetFeature
typenames	Name of dataset collection (e.g., S1.TILE, S2.TILE, L8.TILE, L7.TILE, L5.TILE)
bbox	Filter by bounding box
outputFormat	XML (text/xml; default) or GeoJSON (application/json)
maxcc	Optional: Maximum cloud coverage for scenes
time	Optional: Filter through time (STARTTIME/ENDTIME/P1D)

CEOS WGISS Integrated Catalog (CWIC) – OGC CSW, OpenSearch

The following service endpoint has been requested in this thesis:

- OGC CSW: https://cwic.wgiss.ceos.org/cwicv1/discovery

GEODAB (Sentinel, Landsat, MODIS) – OGC CSW, OpenSearch

OGC CSW and OpenSearch have been used with the following URLs:

- OGC CSW: http://production.geodab.eu/gi-cat-StP/services/cswiso
- OpenSearch: http://production.geodab.eu/gi-cat-StP/services/opensearch

ESA FedEO (Landsat, Sentinel, MODIS) – OGC CSW, OpenSearch

The web service based on the OpenSearch specification is available at the following URL (description document):

- OpenSearch: http://fedeo.esa.int/opensearch/request

A.2 Access to EO time-series data

Within this section, the service endpoints (URLs) and parameters available to access EO scenes are given for the data providers that have been reviewed in Section 4.2. The references of the tables contain information about parameters presented here and further properties of the services.

USGS ESPA (Landsat, MODIS) – self-developed REST API

The following services can be requested (excerpt):

- Send order: POST https://espa.cr.usgs.gov/api/v0/order (Table A.7)
- List orders: GET https://espa.cr.usgs.gov/api/v0/list-orders
- Get order status: GET https://espa.cr.usgs.gov/api/v0/order-status/<ordernum>
- Get order details: GET https://espa.cr.usgs.gov/api/v0/order/<ordernum>
- Get item status: GET https://espa.cr.usgs.gov/api/v0/item-status/<ordernum>

Table A.7: List of parameters to order pre-processed satellite data using USGS ESPA (USGS 2018).

Search Keyword	Syntax and Examples
inputs*	List of satellite scenes separated by satellite sensor
products*	Products to generate (e.g., "bt", "sr", "sr_ndvi", "sr_evi", "sr_savi", "sr_msavi", "sr_ndmi", "sr_nbr", "pixel_qa")
format*	GeoTIFF, HDF-EOS2, ENVI, NetCDF
projection	Projection
image_extents	Bounding box values (output projection need to be specified)
resize	Value in meters
resampling_method	“nn”, “bil”, “cc”

* Mandatory

ESA/Copernicus Open Access Hub (Sentinel) – Open Data Protocol

The following services are available:

- Querying products: https://scihub.copernicus.eu/apihub/odata/v1/Products
- Querying collections: https://scihub.copernicus.eu/apihub/odata/v1/Collections

The metadata of a specific product identifier can be queried as follows, whereas the last part is the internal scene identifier. Within the response additional links for data download and quick-look images are included:

https://scihub.copernicus.eu/dhus/odata/v1/Products('2b17b57d-fff4-4645-b539-91f305c27c69')

Sinergise Sentinel-Hub (Sentinel, Landsat) – OGC WCS, FIS

The following service endpoint can be requested for the OGC Web Coverage Service (INSTANCE_ID need to be replaced by a commercial service key):

- OGC WCS: http://services.sentinel-hub.com/ogc/wcs/{INSTANCE_ID}
- FIS: http://services.sentinel-hub.com/v1/fis/{INSTANCE_ID}

Table A.8: List of parameters to use the Sentinel-Hub WCS service for download (Sinergise 2019a).

Keyword	Value(s)
service	WCS
version	1.0.0
request	GetCoverage
coverage	Name of product to download (e.g., NDVI, TRUE_COLOR)
bbox	Filter by bounding box
time	Mosaic images in time range (STARTTIME/ENDTIME/P1D)
format	Download file format (e.g., image/tiff for GeoTIFF)
evalscript optional	This parameter allows for a custom script or formula specifying how the output will be generated from the input bands.

Table A.9: List of parameters to use the Sentinel-Hub FIS for data extraction (Sinergise 2019b).

Keyword	Value(s)
layer	Preconfigured layer based on which the statistics are computed
crs	Coordinate reference system
time	Filter through time (STARTTIME/ENDTIME/P1D)
resolution	Spatial resolution in meters per pixel
geometry	Geometry in WKT format describing the region of interest
bbox	Bounding box describing the region of interest
style	Style overrides the one specified in the layer configuration
maxcc	Maximum cloud coverage for scenes

Example Google Earth Engine Python API request

```
# define geometry and image collection
geom = ee.Geometry.Point(36.83075, 31.79115);
collection = ee.ImageCollection('MOD13Q1')

# filter image collection by point geometry
collection = collection.filterBounds(geom)

# extraction time-series information from collection for geometry
data = collection.getRegion(geom, None, 'SR-ORG:6974', crs).getInfo()
```

Listing A.2: Data access for point-based extraction of time-series for MODIS MOD13Q1 Vegetation Index dataset using the Google Earth Engine Python API.

Selbstständigkeitserklärung

Ich erkläre, dass ich die vorliegende Arbeit selbstständig und unter Verwendung der angegebenen Hilfsmittel, persönlichen Mitteilungen und Quellen angefertigt habe.

Ort, Datum Unterschrift des Verfassers

Acknowledgements

I would like to express my sincere gratitude to my supervisor Prof. Christiane Schmullius, who gave me the opportunity to come to Jena to study, work, and do research. During the past years I have had great opportunities to further my education, to make national and international contacts, to visit various national and international conferences, and to extensively explore my research field of interest. I am also very grateful to her for her support, long patience, as well as good advices and hints when writing my doctoral thesis. I would also like to thank my second supervisor, Prof. Lars Bernard, who gave very good comments and hints for the polish of the thesis.

A big thank you also to Dr. Christian Hüttich, with whom I discussed major ideas of my work and who supported all concepts of my work. The exchange with him was and still is very important and helpful. I would also like to thank my long-time companions and friends at the department, Dr. Marcel Urban, Robert Eckardt, Dr. Christian Berger and Dr. Christian Thiel, with whom I enjoyed working and discussing. Many thanks also to all other employees of the Remote Sensing Department. I would also like to thank Dr. Christian Strobl from the German Aerospace Center for the scientific, technical, and social exchange over the past years.

I would like to thank Franziska Zander for her many critical comments, good suggestions, and repeated proofreading. My thanks also to Dr. Rene Höfer for the proofreading and the always interesting exchange within the recent projects.

For their cooperation and support in the implementation of the sample applications, many thanks also to John Truckenbrodt (R-Shiny Toolkit, pyroSAR), Felix Glaser (GEO-Wetlands Community Portal, Open Data Cube for Wetlands), Franziska Zander (GEO-Wetlands Community Portal), Marc Becker (R-Shiny Toolkit), Siegfried Clausnitzer (SIB-ESS-C, EOM), and Anna Homolka (SIB-ESS-C), which could only be implemented so numerously with their support. An additional thank you goes to my graphic designer and friend Tobias Burger for the design for all of the web portals and mobile applications.

Finally I would like to thank my friends and my family for their constant support, the many conversations, and wisdom. Special thanks to my parents, Elisabeth and Dieter, for their great guidance, motivation, and care.