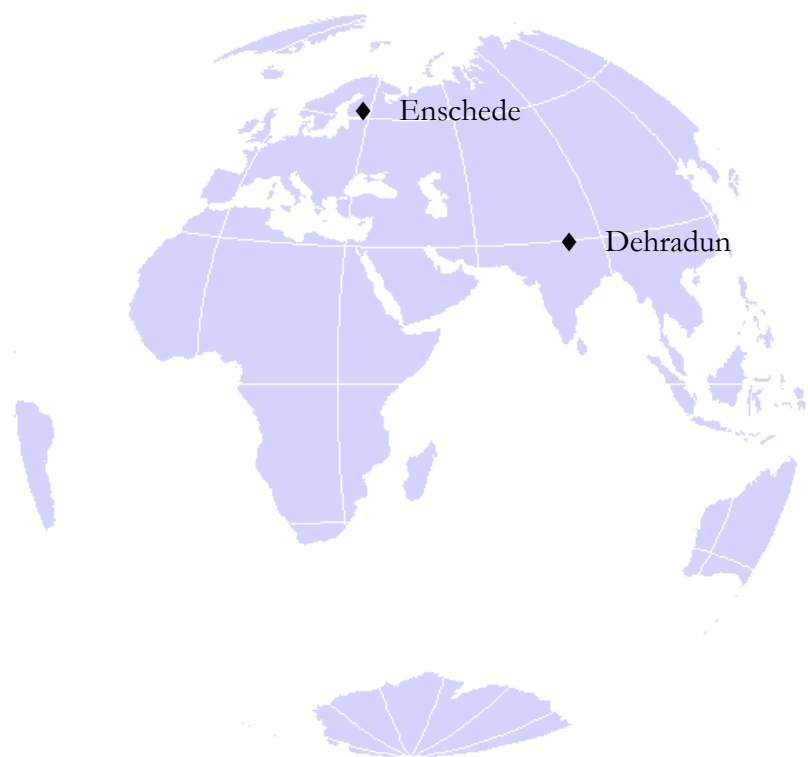


DEVELOPMENT OF OPEN GEO-SPATIAL CONSORTIUM'S WEB PROCESSING SERVICE FOR SPECIES NICHE MODELLING USING ENVIRONMENTAL VARIABLES

BURADA GIRIJA KALYANI



MSc in Geo-information Science and Earth Observation

Specialization: Geoinformatics

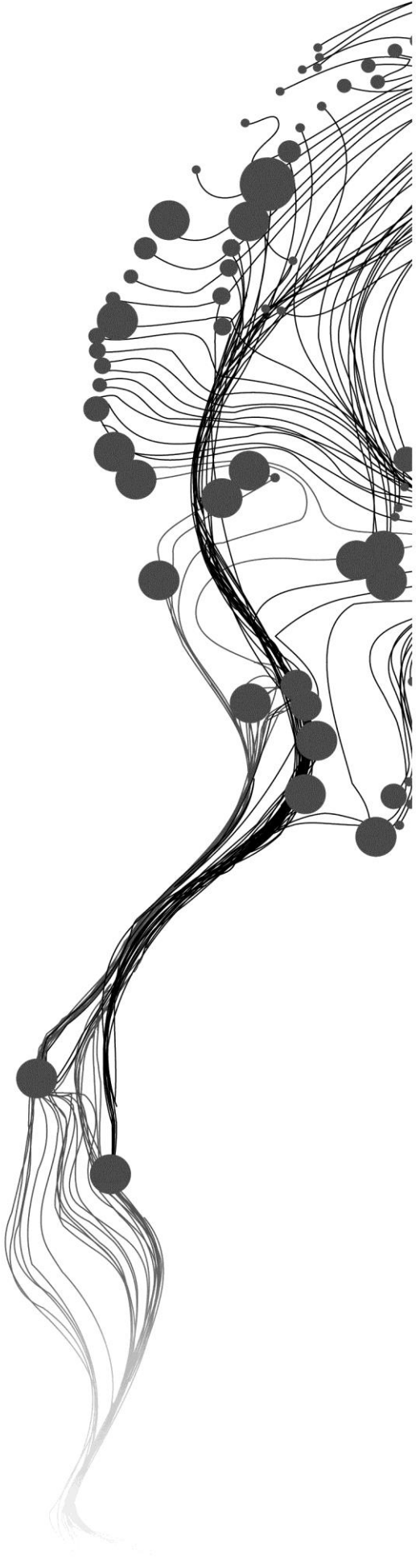
December, 2014



FACULTY OF GEO-INFORMATION
SCIENCE AND EARTH OBSERVATION,
UNIVERSITY OF TWENTE,
ENSCHEDE, THE NETHERLANDS



INDIAN INSTITUTE OF REMOTE SENSING
Indian Space Research Organisation
Department of Space, Government of India



Development of Open Geo-spatial Consortium's Web Processing Service for Species Niche Modelling using Environmental variables

BURADA GIRIJA KALYANI
December, 2014

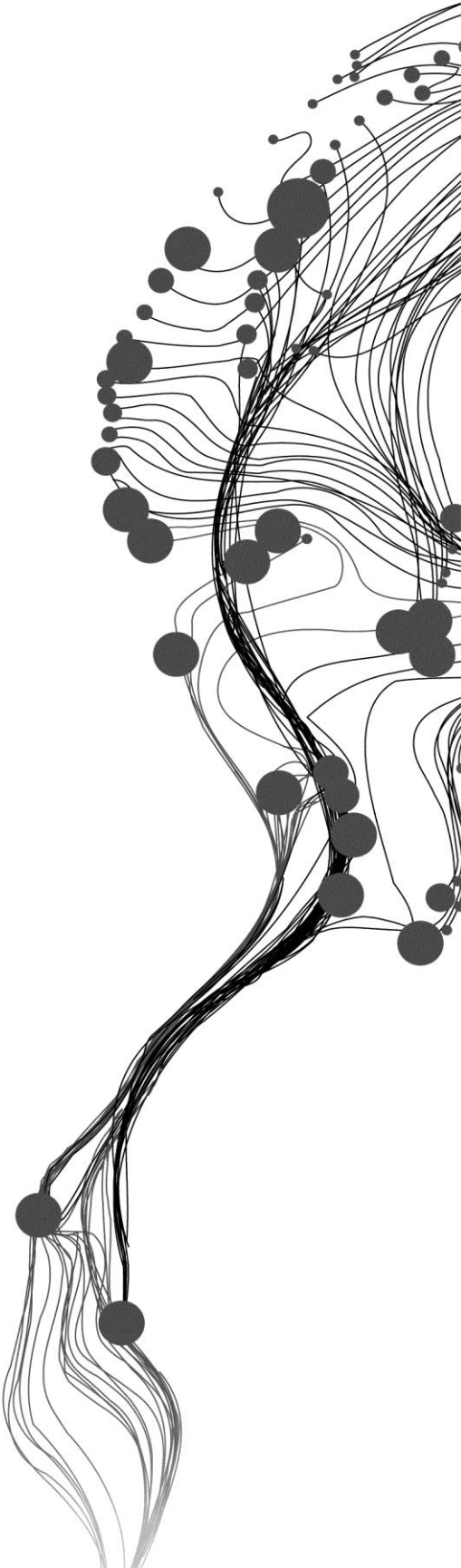
ITC SUPERVISOR

Dr. Alexey.A.Voinov

ISRO SUPERVISORS

Dr. Sameer Saran

Dr. Vinod. M.Bothale



Development of Open Geo-spatial Consortium's Web Processing Service for Species Niche Modelling using Environmental variables

BURADA GIRIJA KALYANI
Enschede, the Netherlands

[December, 2014]

Thesis submitted to the Faculty of Geo-information Science and Earth Observation of the University of Twente in partial fulfilment of the requirements for the degree of Master of Science in Geo-information Science and Earth Observation.

Specialization: Geoinformatics

THESIS ASSESSMENT BOARD:

Chairperson : Prof. Dr. M.J. Kraak
External Examiner : Dr. R.D. Garg (IIT, Roorkee)
ITC Supervisor : Dr. Alexey.A.Voinov
IIRS Supervisor : Dr. Sameer Saran
NRSC Supervisor : Dr. Vinod.M.Bothale

OBSERVERS:

ITC Observer : Dr. Nicholas Hamm
IIRS Observer : Dr. S. K. Srivastav



FACULTY OF GEO-INFORMATION
SCIENCE AND EARTH OBSERVATION,
UNIVERSITY OF TWENTE,
ENSCHEDE, THE NETHERLANDS



INDIAN INSTITUTE OF REMOTE SENSING
Indian Space Research Organisation
Department of Space, Government of India

DISCLAIMER

This document describes work undertaken as part of a programme of study at the Faculty of Geo-information Science and Earth Observation (ITC), University of Twente, The Netherlands. All views and opinions expressed therein remain the sole responsibility of the author, and do not necessarily represent those of the institute.

Dedicated to my papa. . . .

ABSTRACT

Species distribution prediction modelling plays a key role in bio-diversity research. There are increasing needs for species distribution modelling ranging from basic ecological and biogeography research to routine conservation practices. Currently forest scientists and conservationists are looking for web enabled customized tools to model species loss on the fly since they have enormous field data available with them to analyze and visualize. They have many desktop tools developed under open source initiative for understanding species loss with regard to climate change scenarios. Nevertheless web enabled Web Processing Services are still unavailable to them to perform modelling. We propose to publish an Open Geospatial Consortium's (OGC) Web Processing Service (WPS) that performs species distribution modelling process as a Web service and composite them into modelling systems using the scientific workflow approach. Our work demonstrates how a Species Distribution Model can be hosted as a WPS web service. As the requirements of Species Distribution Modelling deal with data, algorithms, calculation models and computing capability, we use 52 degree North Web Processing Service with R-backend (WPS4R) framework that allows to upload and expose R scripts dynamically as WPS processes with the support of high computation capabilities. We predict the distribution of the species *Hippophae Rhamnoides. L* (Sea Buckthorn) in the area Lahaul and Spiti (Bounding Box: 76.3700, 31.7439, 78.6541, 33.2653) district, Himachal Pradesh using Maximum Entropy (MaxEnt) modelling technique to predict the distribution of species. We use species occurrence data collected from Global Biodiversity Informatics Facility and DOS-DBT project. The primary advantage of this approach being, scientists no longer need to use the desktop tools while predicting the species distribution model. They can, on the fly, use their data, request for predicting the species by combining with geospatial processes to produce the prediction map over the web unswervingly. Henceforth the project aims to develop an OGC WPS services for species niche modelling making this applicable to a wide range of species of significant importance for the research and industry community.

Keywords: *OGC WPS, Species Distribution Modelling, MaxEnt, 52°North, R programming, GeoServer*

ACKNOWLEDGEMENTS

A research of this kind requires resources, support and motivation. I therefore take this opportunity to acknowledge all those people who have encouraged and supported me during the entire period of this M.Sc. programme.

This joint course was a wonderful platform to meet people across the globe and experience knowledge sharing in a multi-cultural environment. I would like to thank all the faculty members at IIRS, Dehradun ,at ITC, Enschede and NRSC, Hyderabad.

I am indebted to Dr. Y.V.N.Krishna Murthy, Director, IIRS, for permitting me to carry out the thesis at NRSC, Hyderabad and for his emotional support, constant enquiry, patient hearing and concern throughout the research period. I would like to thank Dr. V.K. Dhadwal, Director, NRSC, for providing state of the art infrastructure and facilities, which helped nurture and enhance my knowledge. Mr.P.L.N Raju has been a great source of motivation all the way through by providing a family like environment and Dr. Srivastav has consistently supported me and helped channelize my efforts into the right direction. I would also like to thank Dr. Nicholas Hamm for his constructive criticism which has helped me to organize my work in a better way that has helped me to finish it well on time.

I'm grateful to all the ITC and IIRS staff for providing me moral support during my sick times and expressing their support by granting me medical leave. I owe my deepest gratitude to all my supervisors for their patience and perennial support and guiding me through my MSc. Thesis. I would like to thank Dr. Alexey A.Voinov, my ITC supervisor, for his visionary approach towards problem definition of this research and critical review of thesis. Apart from the technical support, he also stood by me through my tough times of my health in ITC, as a guardian which would be remembered forever. His deep insight and interest in shaping my knowledge of simulation will always be remembered. It's my pleasure to have Dr.Sameer Saran as my IIRS supervisor, for his inestimable motivation, priceless guidance, invaluable suggestions and support to integrate the technologies for simulations. I would like to express my heartfelt gratitude and thankfulness to Mr. Vinod Bothale, my NRSC supervisor, for keeping me cool always, and allowing me to be flexible in timing keeping my health issues in mind, along with his profound technical knowledge and field experience, helped me frame constructive ideas in succeeding my research. Thanks for setting up wonderful work environment around me that always boosted my will-power sir. I am so thankful to Dr.Matteo, University of Minnesota, for his self-less help throughout the research phase.

I would be failing in my duty if I do not recognize the support and encouragement rendered by Girish Pujar, NRSC for all the technical help provided at intense times of need, for all the discussions, patient listening, insightful suggestions and motivation at every stage of the research.

It is my content to register my heartfelt thanks to my friends; Abhishek, Raj, Tanya and Pradeep from IIRS who have been my never ending sources of strength throughout this phase of research.

From NRSC, Special thanks to Sravanthi for always being there, spreading happiness, being available for stay backs and a consistent partner in burning the midnight oil in the lab. I was lucky to have wonderful Bhuvan staff (NRSC), especially, Aravind sir, Savitha maam, Jyothi maam and entire crew of Bhuvan who have been instrumental in keeping my positivity intact and My sister, my soul mate for keeping my spirits high even when things were not

looking too bright; and Ujwal, for being my all-time support emotionally and building faith when things weren't fine - Thank you so much for just being there.

It's unimaginable to achieve anything without love and support of the family; No amount of gratitude would suffice for the unconditional support and unfailing faith that my parents have shown by supporting me through this phase of my life. Mere words are not enough for everything they have done so far for me. I would be forever indebted for their love and support. My dad was the one who stood by me through all my difficult times. He's my strength and the one behind my research.

And above all I thank God, for giving me the strength in having embarked and completed this journey beautifully so. Even through the dark corners, he left a gleaming ray of hope each time.

TABLE OF CONTENTS

Abstract	
Acknowledgement.....	i
Table of Contents	lii
List of Figures.....	v
List of Tables.....	vii
1 INTRODUCTION.....	1
1.1 Background	1
1.2 Motivation And Problem statement.....	3
1.3 Research Identification.....	5
1.3.1. Research Objective.....	5
1.3.2. Research Questions.....	6
1.4 Innovation Aimed At.....	6
1.5 Thesis Structure.....	6
2 LITERATURE REVIEW.....	7
2.1 Species Distribution Modelling and available Modelling techniques.....	7
2.2 Maximum Entropy (MaxEnt) Technique.....	7
2.3 OGC Web Services.....	10
2.3.1. Types of OGC Web Services.....	10
2.3.1.1. Web Mapping Service (WMS).....	10
2.3.1.2. Web Coverage Services (WCS).....	11
2.3.1.3. Web Feature Services (WFS).....	11
2.3.1.4. Web Processing Services (WPS).....	11
2.3.2. Web Processing Service.....	12
2.4 Role of Web services in Environmental Modelling.....	13
2.5 Open Source softwares available for performing WPS.....	13
2.5.1. GeoServer.....	13
2.5.2. 52 ⁰ North WPS	14
2.5.3. PyWPS.....	14
2.5.4. Degree.....	14
2.5.5. Zoo.....	14
2.6 Related work in WPS with environmental modelling.....	14
3 STUDY AREA, SPECIES, DATA SETS AND SOFTWARE	16
3.1 Species and Study Area	16
3.1.1. Species: Hippophae Rhamnoides. L and its significance.....	16
3.1.2. Study area and Geographical Significance of the area.....	16
3.2 Softwares and tools.....	17
3.2.1. Apache Tomcat and Maven	17
3.2.2. Java (Eclipse IDE)	18
3.2.3. GeoServer	18
3.2.4. 52North WPS.....	19
3.2.5. R- Statistical Programming.....	19
3.2.6. Other software tools.....	19
3.3 Datasets Preparation.....	19

3.3.1. Species data	19
3.3.2. Environmental Data.....	20
4 RESEARCH METHODOLOGY AND DESIGN	22
4.1. Approaches and Methodologies of open source softwares.....	22
4.2. Work flow of the project.....	24
5 IMPLEMENTATION.....	25
5.1. First approach- GeoServer.....	25
5.1.1 Accessing source through GIT.....	25
5.1.2 Maven building the project.....	26
5.1.3 Eclipse IDE with Maven and Tomcat.....	28
5.1.3.1. Tomcat Installation	28
5.1.3.2. Integrating Tomcat with Eclipse.....	28
5.1.3.3. Maven Installation.....	30
5.1.4 Develop a Custom Process.....	32
5.1.4.1. Creating a java process class.....	32
5.1.4.2. Register the process with Spring Framework, Building the project and Deploying into publishing software.....	33
5.1.4.3.. Checking the availability of customized sample process being deployed into GeoServer.....	33
5.2 Second Approach- 52 0 North WPS	33
5.2.1. Configuring 52 ⁰ North WPS.....	33
5.2.2. Implementation of Species Distribution Model Using Maximum Entropy.....	35
6 RESULTS AND DISCUSSIONS.....	39
6.1 Deployment of a sample process class using GeoServer.....	39
6.2 Limitations with GeoServer with respect to Web Processing Services.....	41
6.3 Configuring 520North WPS.....	42
6.4 R-Script Calling MaxEnt Utility.....	46
6.5 Testing the distribution using MaxEnt software.....	48
7 CONCLUIONS AND RECOMMENDATIONS.....	50
7.1 Research Questions.....	50
7.2 Recommendations.....	51
References.....	53-56
User Guide.....	57
Expected errors while installation of softwares and how to solve them.....	58

LIST OF FIGURES

Fig: 1.1	Brief Description of Species Distribution Modelling.....	1
Fig: 1.2	Steps by step procedure for Species Distribution Modelling.....	2
Fig: 2.1	Diagrammatic representation of WPS specification.....	13
Fig:3.1	Hippophae Rhamnoides.....	16
Fig:3.2	Images showing during two extreme seasons; on left, shows the summers, on right shows the winters.....	17
Fig:3.3	Successful installation of Apache Tomcat server.....	18
Fig:3.4	Stack of environmental variables (on left) and its resultant (on right).....	21
Fig:4.1	General Methodology.....	23
Fig:4.2	Diagrammatic representation of executed methodology.....	23
Fig:4.3	Workflow of the project.....	24
Fig:5.1	Cloning GeoServer.....	25
Fig:5.2	Test successful	26
Fig:5.3	Maven loading workspaces of GeoServer.....	27
Fig:5.4	Web Achieve successfully built by GeoServer.....	27
Fig:5.5	Verifying the successful installation of Tomcat.....	28
Fig:5.6	Integrating Tomcat with Eclipse.....	29
Fig:5.7	Setting the port numbers for Tomcat within the Eclipse.....	29
Fig:5.8	Tomcat successfully installed within Eclipse.....	30
Fig:5.9:(a)	Successful installation of maven2eclipse plugin.....	30
(b)	Importing already built maven project externally.....	30
Fig:5.10:(a)	Maven fetching resources to build GeoServer.....	31
(b)	Building workspaces for launching the GeoServer project.....	31
Fig:5.11	Launching and Validating the GeoServer.....	31
Fig:5.12	Verification of successful installation of GeoServer on eclipse.....	32
Fig:5.13	Creating a Java process on Eclipse IDE.....	32
Fig:5.14	Successful built of 52 ⁰ North WPS.....	34
Fig:5.15	52 ⁰ North WPS successfully built on Eclipse.....	34
Fig:5.16	52 ⁰ North WPS running.....	35
Fig:5.17	Verification of successful installation of 52 ⁰ North WPS.....	35
Fig:5.18	Stack of Environmental variables used in the project.....	37
Fig:5.19:(a)	Re-projection with respect to GLCF data (reference data).....	37
(b)	Resultant image after re-projection.....	37
Fig:5.20:(a)	Shape file of the study area – Lahual Spiti.....	38
(b)	Clipping of Environmental data with respect to study area.....	38
Fig:5.21:(a)	Points spread on study area.....	38
(b)	Species points per pixel.....	38
Fig:6.1: (a)	List of Processes available before deploying the customized process.....	39
(b)	List of Processes available after deploying the customized process.....	39
Fig:6.2	Customized process available for selection.....	40
Fig:6.3	After selection of process, Request is sent to the server internally.....	40
Fig:6.4	Description and Execution of process.....	40
Fig:6.5	Contents in the jar file (on left) show no MaxEnt source files are accessible.....	41
Fig:6.6	Submitting credentials to 52 ⁰ North WPS in order to access the webAdmin console.....	42
Fig:6.7	Default repositories in 52 ⁰ North WPS	42

Fig:6.8	Default processes available in LocalAlgorithm repository in 52 ⁰ North WPS	43
Fig:6.9 : (a)	Testing the connection of Rserve to 52 ⁰ North WPS with Telnet.....	43
(b)	Successful installation of Rserve	43
Fig:6.10	Successful activation of new configuration.....	44
Fig:6.11	Configuration of R repository into 52 ⁰ North WPS and deploying the MaxEnt process.....	44
Fig: 6.12	Displaying the capabilities of 52 ⁰ North WPS server through GetCapabilities request.....	45
Fig: 6.13	WPS Test Client displaying the list of registered processes, available for users to select.....	45
Fig: 6.14	Graphical representation of Environmental variables contributing to the growth of Hippophae Rhamnoides (on left) & a snippet of script providing result image.....	47
Fig: 6.15	Prediction of distribution of Hippophae Rhamnoides using Maximum Entropy modelling technique.....	47
Fig: 6.16	Individual variable contribution graph and each Variable's denotation.....	48
Fig: 6.17	(a) Output produced by MaxEnt software.....	49
	(b) Species points used in R was collected from this area for modelling and this resembles the spread of the points shown by software.....	49

LIST OF TABLES

Table: 5.1: A tabular column describing bioclim variables and its significance – 36

1. INTRODUCTION

1.1. Background

All over the centuries humans have observed and recorded consistent relationships between species distributions and the physical environment[1]. Ecologists increasingly use species distribution models to address theoretical and practical issues including predicting the response of species to climate change, identifying and managing conservation areas, and seeking evidence of race among species. In all of these contexts, the core problem is to use information about where a species occurs (and where it does not) and about the associated environment to predict how likely the species is to be present in un-sampled locations. Species distribution models (SDMs) are numerical tools that combine observations of species occurrence or abundance with environmental estimates [1]. The most common strategy for estimating the actual or potential geographic distribution of a species is to characterize the environmental conditions that are suitable for the species, and to then identify where suitable environments are distributed in space[2].

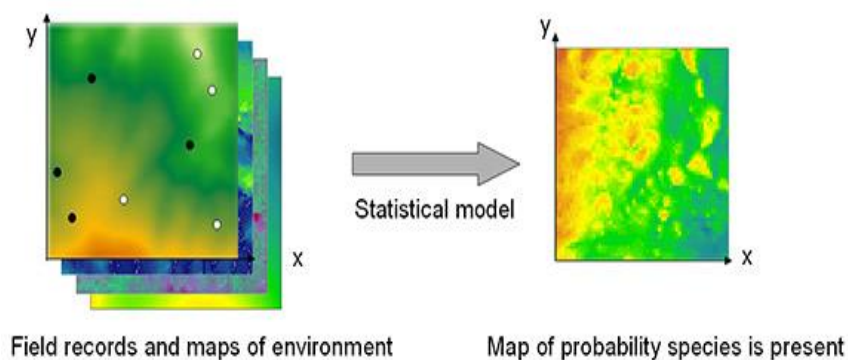


Figure 1.1: Brief Description of Species Distribution Modelling

(Source: ITC module notes-Species Distribution Modeling module)

SDM can be defined as a model that relates the species distribution data (occurrence or abundance at known locations) with information on the environmental and spatial characteristics of those locations [1]. Improved Geographical Information Systems (GIS) software and digital environmental layers permit development of new modeling techniques that create multivariate species' "niche models" covering large geographic areas[3]. These regional niche models incorporate hypotheses about a species' occurrence relative to various environmental variables that are available as GIS spatial layers[3] by combining the digital environmental layers such as elevation, slope, aspect, precipitation, temperature, soil type, land use, and especially vegetation. The modeling technique initially identifies how every individual variable is influencing the distribution of a species. Although it is often more statistical than biological, they serve as a working hypotheses that can guide further, possibly more experimental, investigation, besides assist in implementing and evaluating adaptive

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

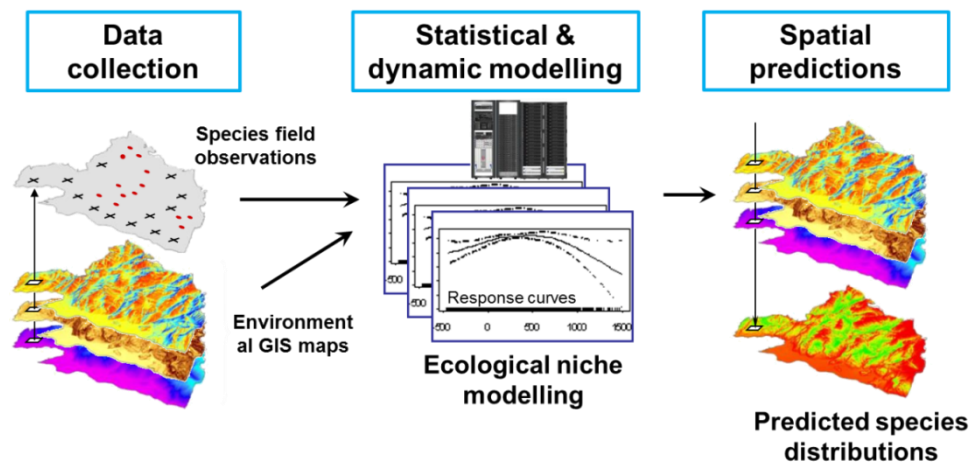


Figure 1.2: Steps by step procedure for Species Distribution Modelling

(Source: <http://www.unil.ch/idyst/en/home/menuinst/research-poles/geoinformatics-and-spatial-m/predictive-biogeography/advancing-the-science-of-eco.html>)

management decisions[3]. Secondly, they provide a spatially explicit assessment of habitat suitability. It is one thing to know what variables are important; knowing where the appropriate combination of variables occurs can be equally valuable. Third, if the model is robust, predictions about habitat suitability can be extended into areas where there is currently no information about the occurrence of a particular species. Such predictions may help to focus additional survey effort or guide the design of more efficient species' preserves.

The current research deals with the study of the species *Hippophae Rhamnoides*, a thorny shrub that grows 2 to 4 feet in height, whose berry is about half a centimeter in diameter and turns bright orange as it ripens by September, supplements health benefit and side effects. This berry has been used for centuries in Asia and Europe as a medicinal herbal product for its richness in antioxidant vitamins and healthy fatty acids. It is generally found in the cold and hilly areas of Ladakh in Jammu & Kashmir, Lahaul-Spiti in Himachal Pradesh and parts of Arunachal Pradesh and Sikkim. The study is confined to the district Lahaul-Spiti of Himachal Pradesh in Western Himalayas covering the Bounding box values of 31.7439 - 33.2653 N Latitudes and 76.3700 – 78.6541 E Longitudes. Recent studies explain that climate change threatens to commit 15-37% of species to extinction by 2050 accelerating a mass extinction precipitated by widespread land use changes. The need to assess these impacts and recommend solutions to policy-makers is correspondingly acute and has been highlighted by the Fourth Assessment Report of the Intergovernmental Panel for Climate change (IPCC, 2007) [1]. To address this problem, we propose the application of maximum-entropy (MaxEnt) techniques which have been so effective in other domains, such as natural language processing[4]. The idea of MaxEnt is to estimate the target distribution by finding the distribution of maximum entropy (i.e., that is closest to uniform) subject to the constraint that the expected value of each feature under this estimated distribution matches its empirical average. Such analyses

require robust infrastructures capable of integrating huge volumes of data from biodiversity archives, satellite remote sensing, and climate change data. The Geospatial Web provides data as well as processing functionality using web interfaces[5]. Such analyses are written by domain experts in scripting languages and rarely exposed as web services[5]. For data processing over the internet, the Web Processing Service (WPS) specification was released on 2005 by OGC in order to provide spatial processes through a standardized service interface based on the Hypertext Transfer Protocol(HTTP)[6].

Hence the project aims at developing a Web Processing Service using open source software for modeling the distribution of species. We have explored GeoServer and 52North WPS to deploy the model as a process, of which 52North WPS is adopted since the distribution of species is scripted in R and deployed into 52North WPS using an R-extended module, WPS4R. The process once deployed can be stored on server end and can be requested from the WPS_Client and executes the process.

1.2. Motivation And Problem Statement

Preserving the world biodiversity has become a major challenge, since human induced changes are responsible for major declines in many species[7]. Therefore, **Species Distribution Modelling** (SDM) aims to explain why does the species occur, where they do occur, and why they do not occur elsewhere [8].For instance, why does an oak tree not occur further south in hotter and dryer regions, and why it may not occur further north in colder and wetter regions? This kind of information is of great societal importance, and amongst the most fundamental of all ecological understanding [8]. Despite the significance to humans in terms of infrastructure, societal operations, and the ecosystem services upon which society depends, our ability to model, predict, and understand the impacts of these changes is very limited[9]. By combining features of the physical landscape and the biological information of the species under investigation, biodiversity researchers build up predictive models for species occurrence and distribution[7]. Ecological modelling can be improved by three possible ways. By improving the existing models, by developing a completely new model which involves a huge budget, and another better option being chaining interoperable model components since by chaining knowledge is shared and user is exposed to more knowledge. Knowledge needs to be embedded inside the models so that biodiversity assessments can be carried out[7]. Scientific knowledge is present not only in the species distribution map or other results of a particular model, but also inside the tools used to produce these results. Models are central tools for modern scientists and decision makers, and there are many existing frameworks to support their creation, execution and composition.[10].

To advance on biodiversity studies, scientists should exchange models and information related to their models, besides sharing data and conclusions. Collaboration among researchers involves inter-comparison between different scientific models and their results wherein modelling environmental systems with

large data keep varying spatially and temporally due to climatic changes[11]. Due to platform dependencies, computer hardware requirements and programming language incompatibilities it is not as easy as it is theoretically explained. Moreover dealing with an infrastructure where models can be easily plugged and played[12] is also a challenging task. Thus we propose a "Model as a web service" to increase model access and sharing. It relies on gradual, biological growth leading towards dynamic webs of interacting models, similar to the World Wide Web. Four basic principles that underlie in publishing a model as a web service is; open access to data, minimal barriers to entry(less restricted), service-driven, and scalability. Any implementation approach meeting these principles will be a step towards the long term vision. Though implementing a Model as a Web service encounters a number of technical challenges including information modelling, minimizing interoperability agreements, performance, and long term access, it is still adopted. To overcome these technical challenges architectural solutions exist [13]. Thus, the WPS architecture that can demonstrate models as interactive, independent and interoperable web service in a distributed environment[11] should be developed. A web service is "a software system designed to support interoperable machine-to-machine interaction over a network"[14]. Hence, an approach that enables individual models to operate and interact with each other using a web service is one of the methods of overcoming this limitation. Thus, to make models and their outputs more accessible, and maintained, interoperability should be achieved [11].

Web sources define interoperability as the ability of two or more systems or components to exchange information and to use the information that has been exchanged. To make data interoperable, two things should be acceptable. Initially, data should be made transferable & accessible seamlessly and secondly, access this distributed functionality seamlessly. In order to specify and set up an infrastructure of interoperable services, which encapsulate distinct parts of the overall functionality and make it accessible via well specified interfaces. . OGC® Standards support interoperable solutions that "geo-enable" the Web, wireless and location-based services and mainstream IT. The OGC is an international industry consortium of 477 companies, government agencies and universities participating in a consensus process to develop publicly available interface standards. "OGC proposes a set of Web services to cover geospatial data, including WMS (Web Map Service, handles a set of spatial layers by geographical extent as an image that can be used by several clients), WFS (Web Feature Service, the exchange of GML (Geography Markup Language) data), WCS (Web Coverage Service, for raster data and predictive habitat model outputs), WPS (Web Processing Service), and CWS (Catalog Web Service)[15].

A Web Processing Service (WPS), is a standard interface that provides rules for standardizing how inputs(requests) and outputs(responses) for geospatial processing services should be developed which defines how a client requests, and how the output is handled. Thus, creation of interoperable services by designing a

modelling service that builds from Open Geospatial Consortium (OGC) Web Processing Service (WPS) protocol. The data required by the WPS can be delivered across a network or they can be available at the server. Therefore, in order to expose the models as web services and to make inferences about the species, this interface is to be developed. They use XML (eXtensible Mark-up Language) for building web services technologies.

As Scientists, forest planners and conservationists working with biodiversity information employs a wide variety of data sources, statistical analysis, and modelling tools, they need tools that can handle huge volume of data from different sources, and may be available on various local and remote platform[7]. Modelling tools for species distribution tools deal with large data sets should be made interoperable for easy access of data, storage, management, and representation[7] from remote places and allow sharing of data and ideas. Thus, an OGC WPS is to be developed for an infrastructure that is interoperable, allows sharing of data and supports multiple web services, fast and has reliable access to calculations of the models.

1.3. Research Identification

Web Processing Service (WPS) specifications are developed by OGC for building various processing tools as interoperable services in web GIS environment. Currently forest planners and conservationist are looking for web enabled customized tools to model species loss on the fly since they have enormous field data available with them to analyze and visualize. Though there are many desktop tools developed under open source initiative for understanding species loss with respect to climate change scenarios, web enabled WPS services are still unavailable to them to perform modelling. Henceforth the project aims to develop OGC WPS services for species niche modelling using environmental variables. Thus, the research is identified with a view to develop a model that as an open source project so that it can easily accessible on web as a process for the clients (i.e. forest planners and conservationists) to perform their analyses.

1.3.1. Research Objective

To develop an OGC Web Processing Service for species niche modelling (typically an existing model like MaxEnt) using environmental variables

Sub objectives:

- 1.3.1.1. Perform a demo of the modelling technique considering a case study of a species *Hippophae Rhamnoides* (Sea Buckthorn)
- 1.3.1.2. Wrapping the model as a Web Processing Service (WPS)
- 1.3.1.3. Integration of the model generated with WPS and makes it available on web as a web processing service
- 1.3.1.4. Check the possible compatibilities of various soft wares and make sure that the combinations of software used so far for integrating the model as a web service aren't replicated.

1.3.2. Research Questions

1.3.2.1. How to integrate the chosen model within WPS?

1.3.2.2. What services of the modelling technique are generated as result?

1.3.2.3. How effective is the OGC WPS service for the implementation of Species Distribution Modelling?

1.4. Innovation Aimed At

The novelty of this project would be exposing the modelling technique as a web service. Despite the availability of the established WPS standards for Web-based geospatial modelling, the technical development end would be an addition to the field of bio-informatics and environmental modelling. The wide uptake of Web access standards helps us in processing and analysing the geospatial data on web through a web service called a Web Processing Service. The novelty lies in the combination of backend software used during the implementation. As of now, the methodology deals with incorporating an already existing model "MaxEnt" on the web to make it easily accessible by developing an Open Geospatial Consortium Web Processing Service using 52⁰North WPS with R-backend.

Thus, an R-script for MaxEnt species distribution modelling is coded and deployed in the configured backend of Web Processing Service web archive. Once the process is registered, it would be available for user to execute.

1.5. Thesis Structure

The research work is organized into the following chapters. Chapter 1, Introduction explains the need and purpose of the research. It consists of background of work, motivation, and requirement analysis, brief summary of how the research work proceeds and research identification.

Chapter 2 summarizes the various research efforts already done in the field. Subsequently in Chapter 3, explains the significance of the Study Area and the species along with the Data sets used and a brief introduction of the soft-wares explored and used. Chapter 4, the research plan and methodology is discussed in detail. Approaches chosen, limitations faced and how it is dealt. Followed by Chapter 5, the Results and discussions are explained as part of Chapter 5 while Chapter 6 presents conclusions and recommendations for future scope.

2. LITERATURE REVIEW

2.1. Species Distribution Modelling and available Modelling techniques

Environmental niche modelling, alternatively known as species distribution modeling, (ecological) niche modelling, predictive habitat distribution modelling, and climate envelope modelling refers to the process of using computer algorithms to predict the distribution of species geographically on the basis of a mathematical representation of their known distribution that is referred as realized niche. The extent to which such modeled data reflect real-world species distributions will depend on a number of factors like nature of the species, and exactitude of the models used and the selection of environmental data layers. The models explore how the occurrence of a species is associated with the environmental variables, and how a species responds to changes in the environment and climate. This helps the conservationists and forestry department to understand the possible threats to a species due to urbanization or climate changes[16]. Environmental niche models are usually referred to correlative models since they relate observed presences of a species to values of environmental variables at those sites in multiple web sources. Some models use absences, as well, but the most of the models use presence-only data, perhaps together with 'random background' data. Results can later be projected into new areas with known characteristics to predict the probability of presence of the species there and trace their potential distributions[17]. Most attempt to predict species' geographic ranges from occurrence (presence; or presence/absence) records (dependent variable) and environmental data from the same sites (independent variables)[18]. Species distribution models can be developed using a variety of algorithms, including heuristic models like BIOCLIM, statistical models like GAMs, combinatorial optimization like GARP and machine learning techniques like ANN and MaxEnt. Each modelling approach has important strengths and weaknesses. Performance is generally assessed using a test or validation data set. When constructing predictive models of the niche of a species, the goal is to predict which areas form part of its potential distribution. MaxEnt is the most widely used modeling technique/software that uses presence only data and performs well when there are few presence records available.

2.2. Maximum Entropy (MaxEnt) Technique

MaxEnt is a program for modelling species distributions from presence-only species records. Its popularity lies in its predictive accuracy that it leaves behind other methods[19]. MaxEnt, one of the most commonly used methods for inferring species distributions and environmental tolerances from occurrence data, allows users to fit models of arbitrary complexity[20]. The MaxEnt method does not need species absence records; instead it uses background environmental data for the entire study area. The method focuses on how the environment where the species is known to occur, relates on the environment across the rest of the study area. The idea is find the probability

distribution of maximum entropy (most spread out), subject to constraints imposed by information available regarding the species presences and the environmental conditions across the study area[4]. In MaxEnt, selected features are formed “behind the scenes”, in the same way as in regression, where the model matrix is augmented by terms specified in the model [21].

MaxEnt model predicts by estimating a distribution of the particular species across geographic space on certain environmental conditions. Those environmental conditions are what we call as environmental variables, predictors or inputs to a model. These factors show suitable habitat for the particular species like estimates of climate on which the species tend to grow, topography, and soil for plants, temperature conditions on which each species grow. The reason MaxEnt supports feature types like linear, hinge, product, quadratic, threshold and categorical is, since species' responses to the environment may be complex, it is usually desirable to fit nonlinear functions. MaxEnt by default allows all feature types. Hinge feature are similar but they even allow response gradient to change. MaxEnt first automatically rescales all features to have the range 0–1. Then, an error bound is calculated for each feature. MaxEnt could estimate feature error bounds only from the data, but to simplify model fitting and because the data are biased mostly with respect to time. Based on the conditional density of the covariates at the presence sites and the unconditional density of covariates across the study area, we will be able to calculate the probability of occurrence of that species. MaxEnt estimates the ratio of conditional density at presence site to unconditional density at study area which is “raw” output. The background data is distribution of these covariates that does not take any account of the presence locations. Thus, it uses the covariate data from the occurrence records and the background sample to estimate the ratio. This can be done by estimating that conditional density with occurrence data is consistent. Consequently this minimizes the relative entropy between these two probability densities defined in feature space. By providing appropriate background samples, of dealing with sample biases, and by modifying the model through feature type selection, the more accurate result can be obtained. Thus, biases should be dealt properly.

Statistical Description of the model: The geographic extent where species presence or occurrence is expected is the study area considered, assume it to be L . while modeling instead of the entire area, a certain of extent of area is chosen using the bounding box area values, let this be L_1 , can also be called as the subset of L , then the distribution of environmental variables in the study area (L) is conveyed by a finite sample – a collection of points from L with associated environmental variables, called a s background sample. These data may be supplied in the form of grids of covariates of the study area; as a default MaxEnt randomly samples 10,000 background locations from covariate grids [21]. The background sample includes only the presence locations and does not take any account of the absence locations. MaxEnt uses the covariate data from the occurrence records and the background sample to estimate the ratio $f_1(z)/f(z)$ [21]. By estimating that $f_1(z)$ is consistent with the occurrence data; though

many distributions are possible, it chooses the one that is closest to $f(z)$ [21]. Minimizing distance from $f(z)$ is sensible, because $f(z)$ is a null model for $f_1(z)$: without any occurrence data, so we cannot expect the species to prefer any particular environmental conditions over any others, so all we can do this predict that the species occupies environmental conditions proportionally to their availability in the given study area. Thus the obtained distance from $f(z)$ is taken to be the relative entropy of $f_1(z)$ with respect to $f(z)$ in MaxEnt which is also known as the Kullback-Leibler divergence[21]. Therefore, the amount of covariates present in the region can be calculated using background data and this provides the source for comparison with the density of covariates occupied by the species ($f_1(z)$). The species' distribution is thus estimated by minimizing the distance between $f_1(z)$ and $f(z)$ or by maximizing the entropy of the raw distribution since that is equivalent to minimizing the relative entropy of $f_1(z)$ relative to $f(z)$. We can define $f(z)$ to be the probability density of environmental variables across the study area L , $f_1(z)$ to be the probability density of covariates across locations within L , where the species is present and z denote vector of environmental variables. Entropy can be defined as "a measure of how much 'choice' is involved in the selection of an event". To ensure that the approximation satisfies any constraints on the unknown distribution that we are aware of, and that subject to those constraints, the distribution should have maximum Entropy[22]. This is the principle of maximum-entropy. This principle is useful explicitly only when applied to testable information. Any information can be said to be testable if it can be determined whether a given distribution is consistent with it[23]. Entropy maximization with no testable information takes place under a single constraint: the sum of the probabilities must be one (web source: wiki). For our purposes, the unknown probability distribution π is over a finite set X . We refer to the individual elements of X as points. The distribution π assigns a non-negative probability $\pi(x)$ to each point x , and these probabilities sum to 1[22]. Our approximation of π is also a probability distribution, and we denote it $\hat{\pi}$. The entropy of $\hat{\pi}$ is defined as;

$$H(\hat{\pi}) = - \sum_{x \in X} \hat{\pi}(x) \ln \hat{\pi}(x) \dots\dots\dots (1)$$

The entropy is non-negative and is at most the natural log of the number of elements in X . MaxEnt generates a probability distribution over pixels in the grid, starting from the uniform distribution and repeatedly improving the fit to the data[22]. This software [24]estimates a target probability distribution by finding the probability distribution of maximum entropy (i.e., that is most spread out) subject to a set of constraints that represent our incomplete information about the target distribution.[22]. For implementing MaxEnt, initially data should be formatted and run. It then builds models of species occurrence starting with a uniform distribution of probability values over the entire grid, and then conducts an optimization routine that iteratively improves model fit, measured as the *gain*. Taking the exponent of the final gain gives the mean probability of the presence samples compared to random background pixels the output will be generated when it is run. Thus, the software implementing Maximum Entropy

modeling generates a html file that summarize model performance, the importance and contribution of each predictor variable through jack-knife test and the shape of each predictor variable's influence as response curves, documentation of the options chosen, and links to raw data files of predictions and model coefficients that can be analyzed elsewhere[25].

2.3. OGC Web Services

Open Geo-spatial Consortium: OGC is a nonprofit, international standards organization that is leading the development of standards for geographic data related operations and services. To solve the interoperability problems, the Open Geospatial Consortium (OGC) has introduced some standards by publishing specifications for the GIS services. The Open Geospatial Consortium (OGC) defines a number of standards, both for data models and for online services, that has been widely adopted in the Geographical Information System (GIS) community leading to a number of software development efforts, online data archives, and application communities[26]. OGC has variety of contributors from different areas such as private industry and academia to create open and extensible software application programming interfaces for GIS [1]. OGC formed Open Geographic Information Systems (OpenGIS) to lead the development of geo-processing interoperability. Originally OGC specifications are not designed to be Web Service compatible[27].

2.3.1. Types of OGC Web Services

Web services: Web Services use Simple Object Access Protocol (SOAP) for messaging which is an XML protocol providing an envelope that encapsulates XML data for transfer through the web infrastructure over HTTP, or either through caches or proxies, with convention for Remote Procedural Calls (RPCs) and a serialization mechanism based on XML Schema data types. SOAP is being developed by World Wide Web Consortium (W3C) in cooperation with the Internet Engineering Task Force (IETF)[26].

2.3.1.1. Web Mapping Service (WMS)

WMS provide users with a means to serve geo-referenced maps available from a GIS server over the web. This service produces a map when requested by the user, answers the basic queries, and communicates their functionality to other programs through GetMap, GetFeatureInfo and GetCapabilities interfaces. Each interface has its own resemblance. In order to retrieve the attribute information connecting a mapped feature user must use the GetFeatureInfo interface, which retrieves the feature information with respect to the map.

2.3.1.2. Web Coverage Services (WCS)

WCS, OGC's raster service standard, retrieves "coverage" or geospatial information pertaining to multidimensional phenomenon at points in space

that vary across geographic region. Each WCS provides access to simple or "grid coverage" information via three operations: GetCapabilities, DescribeCoverage, and GetCoverage. The information (available data and metadata), although not displayed by WCS, may be portrayed by WMS, used in multi-valued coverage, or used in scientific models. Its functionality facilitates the sub-setting, scaling, re-projection, and format encoding of grid data. When operation requests are made using XML they may be encapsulated within a SOAP envelope. It is extended to WCPS and WCS-T.

Web Coverage Service – T (WCS-T): WCS-T is a transactional service operation that can add, modify, or delete grid coverage from a WCS server while providing the updated coverage metadata[28].

Web Coverage Processing Services (WCPS): WCPS is an extension of the WCS v. 1.1.2 and is based on the WCPS language interface standard[29]. The protocol-independent language permits processing requests of "coverage" or multi-dimensional digital geospatial information that varies spatially and/or temporally. It enables the, extraction, processing, and analysis functions pertaining to the sensor, image and statistics data represented by coverage.

2.3.1.3. Web Feature Services (WFS)

WFS interface permits users to access and manipulate geospatial feature information from distributed network sources. Basic operations include GetCapabilities, DescribeFeatureType and GetFeature operations. While complex operations can be performed through WFS-T service interface.

Web Feature Services- Transactional (WFS-T): WFS-T interfaces permit the user to create (insert), delete, update and lock feature instances, and retrieve or query features. The interfaces are written in XML, while GML is used to describe the features within the interface. So that transactions may be properly stored in a data store, transaction semantics are applied.

2.3.1.4. Web Processing Services (WPS)

WPS seems to offer users an unlimited capability to build and facilitate the geo-processing tools for vector and raster data while this implementation acts as middleware or an interface. In this service, the processes relate to calculations, algorithms or models involving geo-referenced data. In publishing, binding information and metadata are made available to facilitate discovery and access. In order to achieve interoperability, application profiles for each process offered are standardized. Whereas, users need not possess any interface knowledge, all they have to do is input data and execute the process. WPS provides for both platform-neutral and platform-specific specifications.

It uses DescribeProcess and Execute operations for encoding requests and responses, embedding data and metadata, referencing inputs and outputs, and requesting storage of outputs.

2.3.2. Web Processing Service

This research deals with an OGC Web Processing Service. This serves and executes geospatial processing on the web. It offers and defines a simple web-based method of finding, accessing, and using all kinds of calculations, processes and models and exposing as a web service. It just describes a method for publishing geospatial processes that operates on spatially referenced data, but does not specify what those processes should be. When a geo-processing service is published with WPS capabilities, the data can be accessed by any client that supports WPS. The WPS interface standardizes the way processes and their inputs/outputs are described, how a client can request, and how the output is handled[15].

WPS uses a HTTP and XML (eXtensible Markup Language) for describing processes and the data that is to be exchanged. WPS defines three operations for the discovery and execution of geospatial processes. The operations are: GetCapabilities, DescribeProcess, and Execute. **GetCapabilities** operation and requests are based on HTTP Get and Post which returns all metadata and processes available through the service in GML format. **DescribeProcess** request contains detailed information for a particular process. This information is necessary for a WPS client to issue subsequent Execute requests to do the actual geospatial processing. Once the process is described, it has to be executed. For performing the described geospatial process on web, an **Execute** request must be sent to the WPS service through a WPS client. It is implemented only as a POST request, there is no raster data I/O support, no process monitoring, no output storage abilities. This usually carries either the value or reference of each input/output parameter of the WPS process and also specifies how the result of the process should be sent back by the WPS service. The obtained results should be stored as simple web accessible resource with an URL. If the results are Raster/Vector results they can be stored directly as WMS layer, if results are in the vector form they can be stored directly as WFS layer and Raster results can be stored directly as WCS layer. Now, this means WPS can trade vector data, raster data, plain strings and numbers, spread sheets or word processor. After the process has been run, the outputs are set and a WPS response is generated. The WPS specification was developed after Simple Object Access Protocol/ Web Service Definition Language (SOAP/WSDL) technologies were standardized[10].

It is must to set up a tomcat server for hosting a web service. In order to setup a connection pool Tomcat needs a JDBC driver and the necessary pool configurations. First off, you need to find the JDBC driver for your database. Once that is done, the Tomcat configuration file is needed in order to setup the connection.

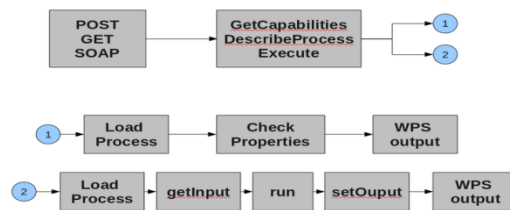


Fig : 2.1: Daigrammatic representation of WPS specification

2.4. Role of Web services in Environmental Modelling

With increasing mapping resources, the Web is becoming a union of collaborative interactions, discussions and planning platforms. In collaboration with this, the integration of Web services with geographic information systems (GIS) has become a new trend for using the Web to compose dynamic processing services for environmental planning so as to integrate services and combine collaborative environmental modelling, a framework was offered by the web services. By wrapping geo-processing services of interoperable standards, we can demonstrate an implementation where web service-enabled middleware adds core functionality to a Web processing Service. We can use geo-processing services to integrate with environmental simulation models using OGC compliant connectors with WPS (Web processing services).

Earth systems are coupled and therefore discovering and sharing geospatial data and processing resources across disciplines are critical for those working in geosciences and the environment. But, professionals trying to understand, use and sustainably manage water, waste, energy, pollution, forests, croplands, oceans and climate find that data and processing of various kinds of data often make discovery and access difficult, even though it is within a discipline. In order to overcome such limitations, OGC, environmental and natural resource researchers and managers work together with geospatial technology providers to develop standards and best practices that enable diverse systems to "talk to each other"[30].

2.5. Open Source softwares available for performing WPS

2.5.1. GeoServer – It is an open source software server designed for interoperability, written in Java, allowing users to share and edit geospatial data, it publishes data from any major spatial data source using open standards. Build up on robust libraries like JTS, GeoTools[31].

2.5.2. 52⁰ North WPS – 52North is a German non-profit group managed associated that develops open source geospatial software. 52North WPS uses Java in an open source implementation to provide a pluggable framework to orchestrate and execute geo-processes. 52⁰North's WPS supports GeoTiff, ArcGrid, and GML2 data types, while its invocation exposes a WSDL document important to some developers for the gridification of GWS [32].

The 52°North Web Processing Service enables the deployment of geoprocesses on the web in a standardized way. It features a pluggable architecture for processes and data encodings. The implementation is based on the current OpenGIS. This implementation was originally build for the research inside the ITC Generalization group but was by now extended immensely and is used in many research projects and production settings, also without 52°North direct participation. The focus of this implementation was to build up an extensible framework to provide algorithms for generalization on the web.

2.5.3. PyWPS - A python based open source for the implementation of OGC Web Processing Service. In order to interface the custom geospatial calculations or processes into a Service Oriented Architecture, with python backend, this is a way to go. PyWPS is completely written in python and well organized into packages and classes. But, this can only run from bash or as a CGI (Common Gate Interface) process.

2.5.4. Degree – Open source software for spatial data infrastructures and the geospatial web. It includes components for geospatial data management, including data access, visualization, discovery and security. Open standards are at the heart of degree. The software is built on the standards of the Open Geospatial Consortium (OGC) and the ISO Technical Committee 211 [32].

2.5.5. Zoo – Open source software, written in C. ZOO Services are example Web services which work with the *ZOO Kernel*. They are based on various existing Open Source libraries and tend to provide simple Web processing functions such as GIS format conversion; GIS file re-projection, basic spatial operations and raster operations. They are based on existing code and prove that the *ZOO Kernel* works with many different codes and languages[33].

2.6. Related work in WPS with environmental modelling

G. Dubois[12] in his paper illustrates a concept of eHabitat, a basic WPS for computing the likelihood of finding ecosystems with equal properties to those specified by a user. eHabitat can be used for ecological forecasting when chained with other services since when new areas are being selected to protect then it becomes a useful tool for decision-makers assessing different strategies. eHabitat can use virtually any kind of thematic data that can be considered as useful when defining ecosystems and their future persistence under different climatic or development scenarios. Thus this paper will present the architecture and illustrate the concepts through case studies which forecast the impact of climate change on protected areas or on the ecological niche of an African bird.

Anthony M. Castronova[11], in his paper illustrated that due to platform dependencies, architecture requirements, and programming language incompatibilities it is not always straight forward to create a modeling workflow even after ensuring coupling components developed by every developer are linkable. To overcome all these, a service-oriented approach that enables individual models to operate and interact with others using web services is being developed. Hence, this paper illustrates how the WPS protocol can be used to create modelling services, and then demonstrate how they can be brought into workflow environments using generic client-side code. Within the HydroModeler environment, a model coupling tool is built on the Open Modelling Interface standard, and hydrology model is hosted as a WPS service and used within a client-side workflow. Thus the server-side software follows an established standard that can be leveraged and reused within multiple workflow environments and decision support systems[11] .

3. STUDY AREA, SPECIES, DATA SETS AND SOFTWARE

3.1. Species and Study Area

3.1.1. Species: *Hippophae Rhamnoides. L* and its significance

The research deals with the species *Hippophae Rhamnoides Linn.*, also known as Sea-Buckthorn, an economically and ecologically important medicinal plant comprising of species which are winter hardy, deciduous, wind-pollinated multipurpose shrubs bearing yellow or orange berries with nitrogen-fixing ability[34]. These plants contain steroids, alkaloids and sugars; hence its fruits are rich in vitamin-C and seeds yield fatty oil. Due to its medicinal richness, it is used for the treatment of tumors and sun-burn preventing preparations. It is also commercially useful since it is used in medicines, skin creams, oils, jams, jellies, squashes and also wines. It treats many diseases like bowel irregularities, gastric ulcers, skin infection/wounds, influenza infections, cough and cold. Along with its medicinal qualities, this herb is also known to play a major role in cosmetic preparations since it can prevent the wrinkles under the eye region. Apart from cosmetic values, sea-buckthorn is also popular for its environmental importance. Horticulturists and landscapers treasure the sea buckthorn bush because it is an excellent retainer of soil. Although this plant has many excellent traits, it is still in an early phase of domestication and is prone to many pests and diseases that is destroying it and halting its commercial production. So by predicting the occurrence of this species, necessary measures can be taken for its protection.



Fig: 3.1: *Hippophae Rhamnoides* (Source: Wikipedia images)

3.1.2. Study area and Geographical Significance of the area

The genus *Hippophae* is distributed in the region from Pakistan to Himachal Pradesh regions and on riversides of Central Asia between 2100 and 3600 m. Studies say that it is most commonly grown in the cold and arid regions like Keylong, Pooh division and Baspa valley in Kinnaur between 2800 and 4200 m and is usually found in Spiti[34]. Though Sea buckthorn (SB) grows in cold regions of Asia, Europe, and North America, Indian Himalayas host world's second or third largest area under SB

(30,000 – 40,000 ha)[35]. Thus, the study area is confined to Lahaul-Spiti. In India, the Trans-Himalayan range is cold and arid at an altitude from 2500 m to 4500 m suitable for its growth.

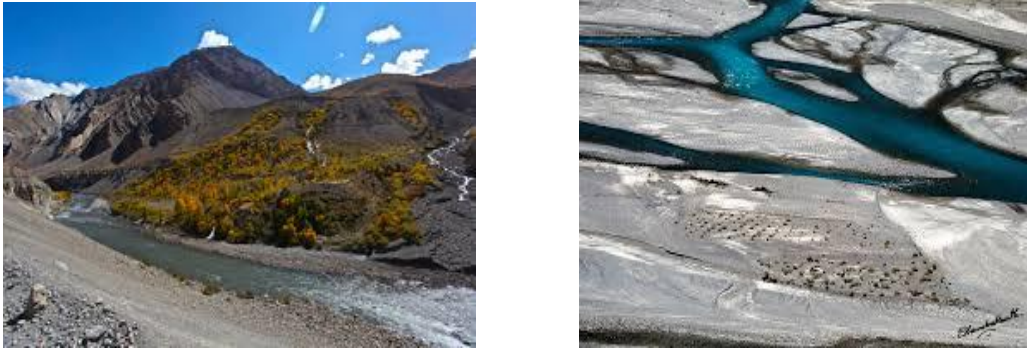


Fig : 3.2 : Images showing during two extreme seasons; on left, shows the summers, on right shows the winters

The growth of species is rich in adverse conditions where the temperature varies between -30 to 30°C and having annual precipitation only 9 cm[35]. Since the Himachal Himalaya is significant in its enormous arcade formation of geographical landmark with diverse topographical features and widely varying altitudes and climates, it is grown in this region. Lahaul and Spiti is said to have a noticeable latitudinal series of vegetation types with the tropical forest, leading to the eternal snow on the peaks. It is well known for its complicated climatic set-up since some regions are hot tropical, while the others are dry (cool) in temperate[34]. With reference to the document [34], Spiti, is said to have the annual snowfall of 72 cm and the annual rainfall of 5 mm with fluctuation in temperature ranging 30.5 and -19.5°C . It is cloudy most of the times and is cloudy for about 60 days in a year. The mountainous terrain extends from Baralacha Pass (5333 m) in Lahaul and Spiti to Kanan village of the Pooh sub-division in Kinnaur district. Hence, we justify that the area Lahaul-Spiti is satisfying the climatic conditions and geographic features which are of major concern to assess the distribution of species and the qualities of the species growth. Thus the area is appropriate for the study.

3.2. Softwares and tools

3.2.1. Apache Tomcat and Maven

Tomcat is one of several open source collaborations that are collectively known as Jakarta. It can also be used as a standalone service. It is a web container that allows running servlets and Java Server Pages (JSP) based web applications. Apache Tomcat by default provides a HTTP connector on port 8080, i.e., Tomcat can also be used as HTTP server. The following figure shows up when a system is installed with Apache Tomcat application server.

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

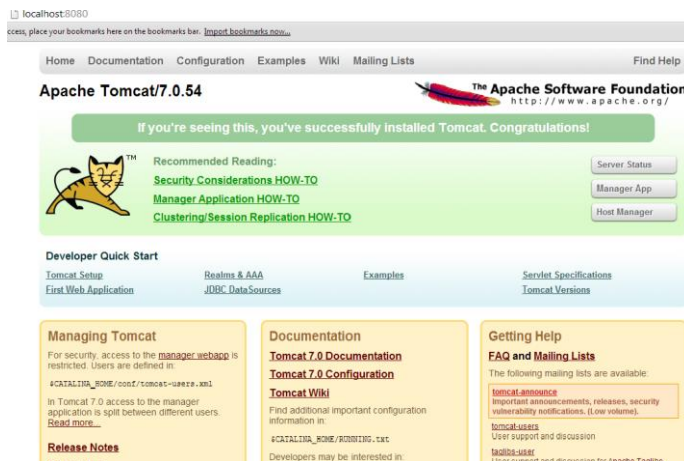


Fig. 3.3: Successful installation of Apache Tomcat server

3.2.2. Java (Eclipse IDE)

Eclipse is an Integrated Development Environment (IDE), contains a workspace and an extensible plugin system for customizing the environment. The Eclipse Software Development Kit (SDK) includes the Eclipse Java development tools (JDT), offering an IDE with a built-in Java compiler and a full model of the Java source files allowing code analysis leading to a wide usage of Java to develop applications. By means of various plug-ins, it is used for various softwares few of them being Ruby, R, Python, C, C++, PHP. Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT (C development tools) for C/C++ and Eclipse PDT (PHP development tools) for PHP, among others. Users can customize their own plugins and contribute their own ideas by installing plugins. The Eclipse Web Tools Platform (WTP) project is an extension of the Eclipse platform with tools for developing Web and Java EE applications. It includes source and graphical editors for a variety of languages, wizards and built-in applications to simplify development, and tools and APIs to support deploying, running, and testing apps. We used Eclipse IDE (Kepler) for our project with Web Tools Platform.

3.2.3. GeoServer

GeoServer is a free and very reliable Java-based open source software server that allows users to share and edit geospatial data. It is designed for interoperability, and publishes data from any major spatial data source using open standards. It aims to operate as a node within a free and open Spatial Data Infrastructure, similar to Apache HTTP Server, which is free and an open web server to publish Hyper Text Mark-up Language (HTML) files. The similar kind of functionality as Apache HTTP server is what GeoServer does, but with geospatial data. Though GeoServer provides RESTful services and packages Jetty, as an embedded web server, it also supports other servlets too like Apache Tomcat too.

3.2.4. 52⁰North WPS

This is also a Java based open source, but it has got web access to geospatial processing algorithms provided by Sextante, ArcGIS Server, R, GRASS 7 or custom developed functions. These can be either simple algorithms or complicated. Once the algorithm is scripted or designed, it can be deployed within 52⁰North WPS in a standardized way. It features a pluggable architecture for processes and data encodings and pluggable framework for algorithms and XML data handling and processing framework. It is built up on robust libraries like JTS, GeoTools, xmlBeans, servlet API, and derby.

3.2.5. R- Statistical Programming

R is an open-source implementation of the S language for statistical computing. To deal with statistical operations, R is the language dedicated for numerous math functionalities and is freely available. A majority of implementations of new statistical techniques have been made available as packages in R, which include the code as a library of functions and a specific library of use can be called when it is in use. R is an open source, where experts make their field-specific code and functions and make them freely available as packages. There are several packages for species richness, diversity, rarefaction, distribution. Wildlife biologists provide several packages for estimating occupancy & abundance from various forms of data: unmarked, mra, Rcapture, secr, PresenceAbsence. Thus we used a package 'dismo' (being the major one, and many other packages like sp, maptools, XML, data.frame were also used), which is exclusively available for modelling the distribution of species.

3.2.6. Other software tools

ArcGIS is an ESRI product, used in GIS environment for working with maps and geographic information like creating and using maps in wide applications; such as compiling geographic data; analyzing mapped information; sharing and discovering geographic information; and managing geographic information in a database. The system provides an infrastructure for making maps and geographic information available throughout an organization, across a community, and openly on the Web.

DIVA GIS is predominantly useful for mapping and examining biodiversity data, such as the distribution of species, or other 'point-distributions'. It reads and writes standard data formats such as ESRI shape files, so interoperability is not a problem. DIVA-GIS can run on Windows and Mac OSX as well.

ERDAS IMAGINE is designed by ERDAS for geospatial applications. ERDAS IMAGINE is meant primarily for geospatial raster data processing and allows the user to prepare, display and enhance digital images for mapping use in GIS and in Computer Aided Designs (CAD) software. It is a toolbox allowing the user to perform various geospatial operations on an image and make geographic problems understood. By manipulating imagery data values and positions, it is possible to see features that

would not normally be visible and to locate geo-positions of features that would otherwise be graphical. The level of brightness or reflectance of light from the surfaces in the image can be helpful with vegetation analysis, prospecting for minerals. Spatial models can also be developed in ERDAS, import/export of data for a wide variety of formats, orthorectification, mosaicking of imagery, stereo and automatic feature extraction of map data from imagery. We used ERDAS for data preparation

3.3. Datasets Preparation

3.3.1 Species data

Species data is the observation of where the species occurrence is found. Observation data for an SDM can come from many different sources, and an analyst needs to identify the most inclusive datasets available for a particular taxon and geographic region. Global Biodiversity Information Facility is one such facility that provides huge global data available on web and is accessible through web. While collecting the datasets, it is necessary to know whether both the presence and absence data is available. Typically, a model developed from an observation dataset with true absences will have fewer biases introduced than one generated from a presence-only dataset[22]. Though presence-absence data gives an accurate model, it is feasible to create models with presence-only data(in case of MaxEnt); that is often done through simulating absences by generating random points across the geographic region using environmental data.

The species chosen for our study is *Hippophae Rhamnoides*. The occurrence points are collected from GBIF, an international open data infrastructure, funded by governments. In India, occurrence points of the species *Hippophae Rhamnoides* was accurately sixty. Most of them covered the Northern part of India in the regions of Leh and Ladakh, while nine points belonged to our study area Lahaul and Spiti. With reference to these points, species points were collected from the "Biodiversity project" by National Remote Sensing Centre, ISRO. The research was carried out by 66 occurrence points, out of which the data was divided into training and testing sets of 46 and 20 respectively. The data available for the study typically consists of a set of geographic coordinates where the species has been observed that contain geo-referenced occurrence localities.

3.3.2 Environmental data

The species distribution modeling workflow involves collection of relevant environmental conditions that affect the growth of species called environmental variables or predictor variables which are stacked, and used in modeling to develop a statistical model expressing the relationship between known observations(occurrence points) and environmental variables, and then mapping this relationship to geographical space (study area concerned)[36]. Species distribution models need raster data formats to analyze; hence the data format provided is almost available in raster format or can be converted to required formats according to the necessity of the

model. Reference data for the study being thematic layers collected from Global Land Cover Facility (GLCF), with seven bands at a resolution of thirty meters, according to which the entire data was modified and re-projected. The data were collected from WorldClim at a resolution of 30 arc seconds, for nineteen bioclim variables, along with the minimum temperature and maximum temperature variables, precipitation variables of October month was selected. The slope and aspect variables were derived from altitude data. For MaxEnt, the raster formats are given in the ascii format. Choosing environmental variables is very important in SDM since the environmental variables are the driving forces for the distribution of niche, as they are meant to reflect occurrence of species. Hence, there should be a direct or indirect relationship to a species' physiology or its needed resources with environmental variables. Environmental variables include climate, topographic factors, temperature, precipitation and the distributions of other species, and disturbances. Usually, it is said that the environmental variables correlate with one another. Thus if the model is performed to understand the ecological distribution as well as spatial prediction, it is better to reduce the number of highly correlated variables[36]. Thus the data preparation is done.

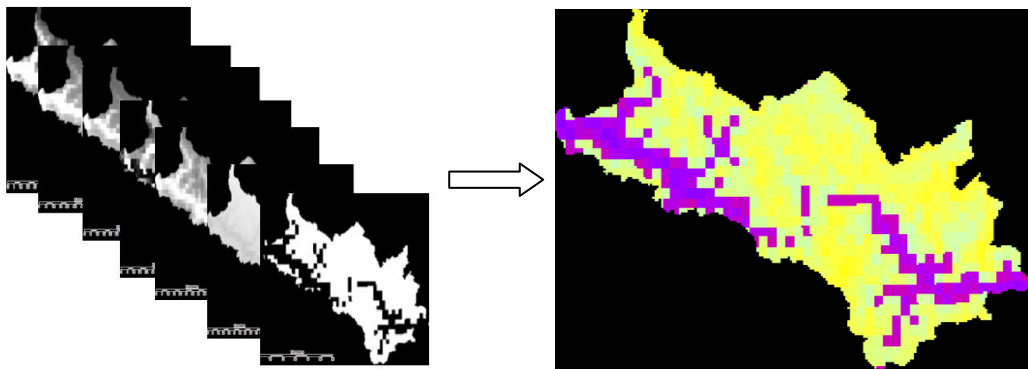


Fig: 3.4: Stack of environmental variables (on left) and its resultant (on right)

Environmental variables, such as average temperature, average rainfall, elevation, precipitation and bioclim variables have been measured or estimated across a geographic region of interest. The goal is to predict which areas within the region satisfy the requirements of the species' ecological niche, and thus form part of the species' *potential distribution*.

4. RESEARCH METHODOLOGY AND DESIGN

4.1. Approaches and Methodologies of open source softwares

We proposed to explore all the possible approaches available for developing a web processing service and then develop a customized process, deploy it into web as a process and make it available to multiple users as a service. Thus, making our customized process available as a service was the primary agenda. In order to achieve this, we initially explored all the open source softwares which can be used to develop a process and thereby allowing the process to be deployed in the software by using the available open source as an interface.

In our exploration, we learnt about five open source softwares; GeoServer, 52⁰North, PyWPS, deegree, Zoo. All of these softwares have similar characteristics except that they differ with respect to backend functionality (i.e., it differs with software) and usage of interfaces. GeoServer is written in Java, an open source software server designed for interoperability, which allows the clients to share and edit geospatial data building on libraries like Java Topology Suite and GeoTools. Being a core Java project many plug-ins are connected to various remote sensing softwares so as to build the project. 52⁰ North WPS, is also open source geospatial software written in Java that provides a pluggable framework to organize and execute geo-processes and enables the deployment of geo-processes on the web in a standardized way. It features a pluggable architecture for processes and data. PyWPS is a python based open source, is used to interface the customized geospatial calculations or processes into a Service Oriented Architecture, with python backend and though it is said to be well organized into packages and classes, it is limited to bash (command prompt) to run a process and there is no proper CGI (Common Gate Interface) process available making it difficult to novice users. Degree, also a java-based open source software framework that contains the services needed for portal components (deegree iGeoPortal), mechanisms for handling security and access control issues (deegree iGeoSecurity) and storage / visualization of 3D geo-data (deegree iGeo3D)[37]. Zoo, is an open source software, written in C. These services work based on existing Open Source libraries and tends to provide basic functionalities so as to prove that the Zoo kernel works with many different codes and languages.

Our goal was to establish a framework where user can provide species input through an interface, and the request connects the publishing software (which may be 52⁰North WPS, GeoServer, PyWPS, deegree, Zoo). This publishing software connects the developed process and processes the species distribution modelling internally and provides the species distribution prediction to the user.

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

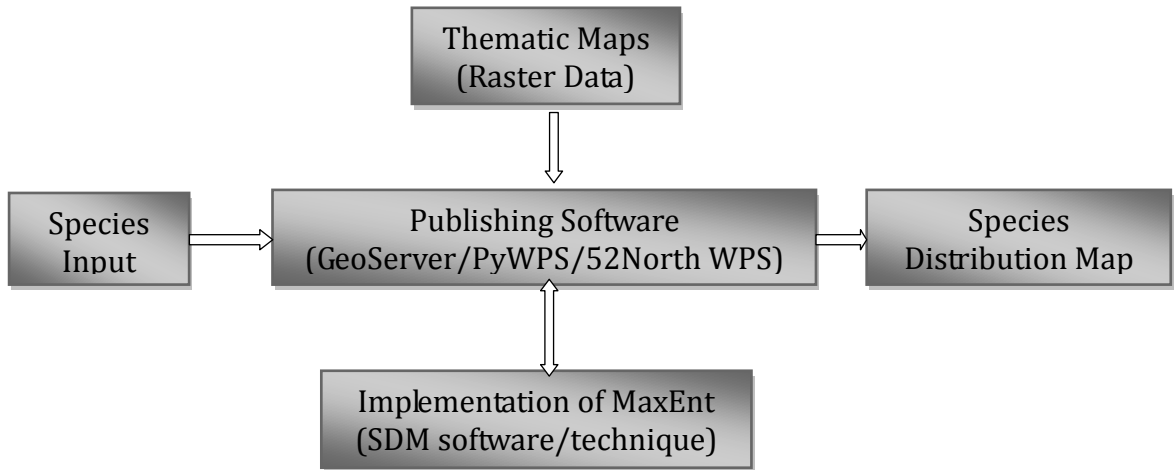


Fig: 4.1: General Methodology

The availability of approaches was shown in the diagram above (4.1). Among which, we have chosen 52⁰ North WPS, for our research. With 52⁰North WPS, we provide the species input through R-interface as shown in the figure (4.2)

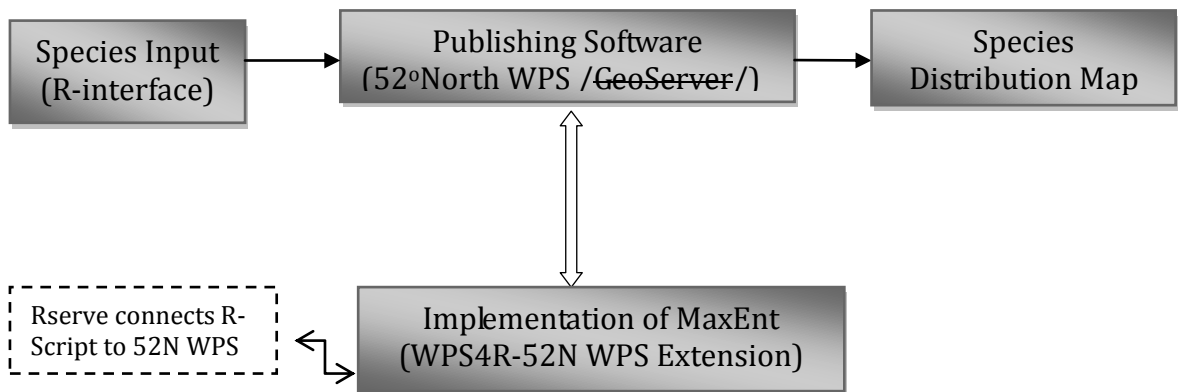


Fig: 4.2: Diagrammatic representation of executed methodology

In the above mentioned approach (as per figure 4.2.), User is expected to provide species input through R, and the publishing software 52⁰North WPS receives the input through R-script, connects the R-statistical programming software through Rserve, a TCP/IP server which allows other programs to use facilities of R [38] from various languages without the need to initialize R or link against R library. It responds to the requests from clients and listens for any incoming connections and processes. Rserve Client-side implementations are available for popular languages such as C/C++, Java and supports remote connection, authentication and file transfer.

```
> library(Rserve)
> Rserve()
Starting Rserve...
"G:\Software\R-31~1.1\library\Rserve\libs\x64\Rserve.exe"
>
```

4.2. Work flow of the project

The figure (4.3) shows the flow of the project. To begin with, so as to develop any open source software, we need to set up the source files of that particular project and build the project on the local system. Once the project is built as expected, we can develop our particular processes and customize the project as per our requirement. Once the process is developed, we have to register it with the open source software and allow users to access the process whenever the request is sent through a Uniform Resource Locator (URL).

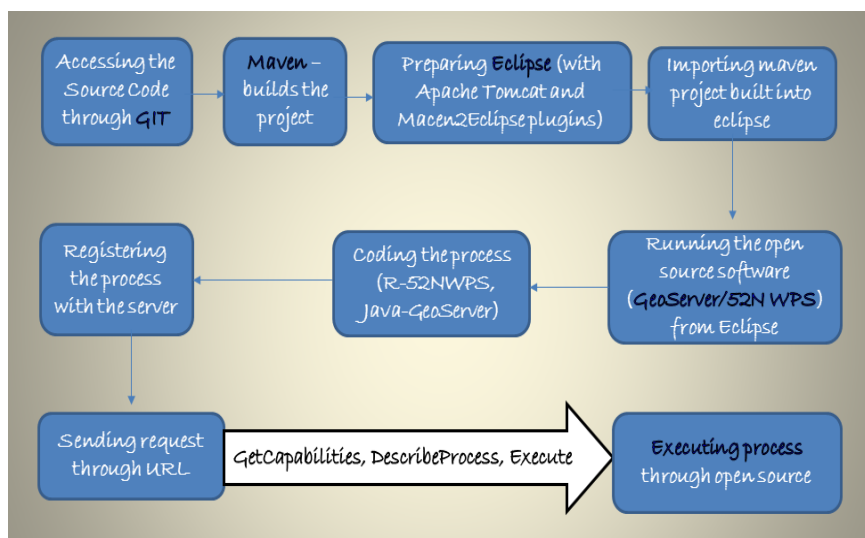


Fig: 4.3: Workflow of the project

User send a “GetCapabilities” request to know what process the server is capable of performing, it means, the services registered with the server will be displayed to the user. The user chooses the process he wishes to perform among those registered and the server describes the process when “DescribeProcess” is requested. After which the process will be executed.

5. IMPLEMENTATION

Our goal was to develop a customized process and make it available as service on web for any user for which we have explored all the possible approaches and softwares. We started with GeoServer, and the following methodology was implemented.

5.1. First approach- GeoServer

GeoServer was installed step wise referring to the document [39]. For this research project, we accessed the source code of GeoServer using Git, standard Maven tool for building the project were used and standard practices suggested in GeoServer development documentation were studied. The integrated development environment (IDE) used in this study is Eclipse (Kepler) and Java programming language is used for software development and customization. For this, Java Run Time Environment (JRE) and Java Development Kit (JDK) along with Tomcat were installed in a computer system. For integrating project built by Maven and Java code within a common platform, we used Eclipse IDE. The installations required for the application development are as mentioned below. To understand, let's start step by step installation process.

5.1.1. Accessing source through GIT

GeoServer source code is stored on Github, a Git repository web-based hosting service, which offers all of the distributed revision control and source code management (SCM) functionality of Git as well as adding its own features. Git, is strictly a command-line tool,

```
[INFO] [GitCommitIdMojo] build.timestamp = 14-Jul-2014 18:09
[INFO] [GitCommitIdMojo] build.build.time = 14.07.2014 @ 23:46:04 IST
[INFO] [GitCommitIdMojo] build.branch = master
[INFO] [GitCommitIdMojo] build.commit.user.email = christian.mueller@os-solutions.at
[INFO] [GitCommitIdMojo] build.build.user.name = GIRIJA-KALYANI
[INFO] [GitCommitIdMojo] build.commit.user.name = Christian Mueller
[INFO] [GitCommitIdMojo] build.commit.id.abbrev = 8d0d048
[INFO] [GitCommitIdMojo] build.commit.id = 8d0d048988615371b53fd7c410b99d03e3b7f4ae
[INFO] [GitCommitIdMojo] build.commit.message.full = Merge branch 'GEOS-6514'

[INFO] [GitCommitIdMojo] -----
[INFO] [GitCommitIdMojo] Finished running.
[INFO]
[INFO] --- maven-jar-plugin:2.4:test-jar (default) @ geoserver ---
[WARNING] JAR will be empty - no content was marked for inclusion!
[INFO] Building jar: C:\Users\del1\Documents\Github\geoserver\src\target\geoserver-2.6-SNAPSHOT-tests.jar
[INFO]
[INFO] >>> maven-source-plugin:2.2.1:jar (attach-sources) > generate-sources @ geoserver >>>
[INFO]
[INFO] --- git-commit-id-plugin:2.0.4:revision (default) @ geoserver ---
[INFO] [GitCommitIdMojo] Running on 'C:\Users\del1\Documents\Github\geoserver\src' repository...
[INFO] [GitCommitIdMojo] Initializing properties...
[INFO] [GitCommitIdMojo] Using maven project properties...
[INFO] [GitCommitIdMojo] Loading data from git repository...
[INFO] [GitCommitIdMojo] -----
[INFO] [GitCommitIdMojo] git properties loaded
[INFO] [GitCommitIdMojo] build.commit.message.short = Merge branch 'GEOS-6514'
[INFO] [GitCommitIdMojo] build.build.user.email = giri.kalyani18@gmail.com
[INFO] [GitCommitIdMojo] build.hudson.id = -1
[INFO] [GitCommitIdMojo] build.commit.time = 17.06.2014 @ 21:07:47 IST
[INFO] [GitCommitIdMojo] build.timestamp = 14-Jul-2014 18:09
[INFO] [GitCommitIdMojo] build.build.time = 14.07.2014 @ 23:46:06 IST
[INFO] [GitCommitIdMojo] build.branch = master
[INFO] [GitCommitIdMojo] build.commit.user.email = christian.mueller@os-solutions.at
[INFO] [GitCommitIdMojo] build.build.user.name = GIRIJA-KALYANI
[INFO] [GitCommitIdMojo] build.commit.user.name = Christian Mueller
[INFO] [GitCommitIdMojo] build.commit.id.abbrev = 8d0d048
[INFO] [GitCommitIdMojo] build.commit.id = 8d0d048988615371b53fd7c410b99d03e3b7f4ae
[INFO] [GitCommitIdMojo] build.commit.message.full = Merge branch 'GEOS-6514'

[INFO] [GitCommitIdMojo] -----
[INFO] [GitCommitIdMojo] Finished running.
```

where-as Github provides a web-based graphical interface and desktop as well as mobile integration. We cloned into GeoServer's repository through Git a command line tool for accessing the source code as per the instruction given in [40], and Figure (5.1) shows cloning into Github for accessing source code of GeoServer to run its developing back-end. Once the source is downloaded using Git, the project should be built using Maven which is described in the following session.

Fig. 5.1: Cloning GeoServer

5.1.2. Maven building the project

Maven, a build automation tool, is used for building Java projects automatically. Maven can manage a project's build, reporting and documentation from a central piece of information. It addresses two aspects of building software: First, it describes how software is built, and second, it describes its dependencies. An XML file describes the software project being built, its dependencies on other external modules and components, the build order, directories, and required plug-ins based on the concept called project object model (POM). It comes with pre-defined targets for performing certain well-defined tasks like compilation of code and its packaging. Maven dynamically downloads Java libraries and Maven plug-ins from one or more repositories such as the Maven 2 Central Repository, and stores them in a local cache. This local cache of downloaded artifacts can also be updated with artifacts created by local projects and public repositories can also be updated as well. Maven is built using a plugin-based architecture that allows it to make use of any application controllable through standard input. Theoretically, this would allow anyone to write plugins to interface with build tools for any other language. In reality, support and use for languages other than Java has been minimal.

For installation of Maven, the session “maven guide” from GeoServer's development tool[41] was referred, studied and executed as shown in the figure below. The figure (5.2) shows the maven tool downloading the repositories required for building the project and saving them in system's maven's local repository.

The command `'mvn clean install -P restconfig'` as shown in figure (5.2), builds all the modules and installs it in the *local repository*. The local repository is automatically created in the home directory (or if expected to store in a different location, we have to define the alternative location), where all the downloaded binaries and the projects built are stored.

```
F:\Sample\practice\buffer>mvn clean install -P restconfig
[INFO] Scanning for projects...
[INFO]
[INFO] -----
[INFO] Building buffer 1.0-SNAPSHOT
[INFO] -----
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ buffer ---
[INFO] Deleting F:\Sample\practice\buffer\target
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ buffer ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory F:\Sample\practice\buffer\src\main\resources
[INFO] --- maven-compiler-plugin:2.5.1:compile (default-compile) @ buffer ---
[INFO] Compiling 1 source file to F:\Sample\practice\buffer\target\classes
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ buffer ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory F:\Sample\practice\buffer\src\test\resources
[INFO] --- maven-compiler-plugin:2.5.1:testCompile (default-testCompile) @ buffer ---
[INFO] Compiling 1 source file to F:\Sample\practice\buffer\target\test-classes
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ buffer ---
[INFO] Surefire report directory: F:\Sample\practice\buffer\target\surefire-reports

-----
T E S T S
-----
Running buffer.AppTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.006 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ buffer ---
[INFO] Building jar: F:\Sample\practice\buffer\target\buffer-1.0-SNAPSHOT.jar
[INFO] --- maven-install-plugin:2.4:install (default-install) @ buffer ---
[INFO] Installing F:\Sample\practice\buffer\target\buffer-1.0-SNAPSHOT.jar to C:\Users\dell\.m2\repository\1.0-SNAPSHOT\buffer-1.0-SNAPSHOT.jar
[INFO] Installing F:\Sample\practice\buffer\pom.xml to C:\Users\dell\.m2\repository\buffer\buffer-1.0-SNAPSHOT-1.0-SNAPSHOT.pom
[INFO] BUILD SUCCESS
[INFO]
```

Fig: 5.2: Test successful

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

The figure (5.3) is a screen shot showing how the maven is loading the GeoServer's workspaces before building the project.

```
6 Jul 23:22:02 INFO [storage.DefaultStorageFinder] - *** Found System environ
Program Files\GeoServer 2.5.1\DATA_DIR, using it as the default prefix. ***
6 Jul 23:22:02 INFO [storage.DefaultStorageFinder] - *****
*****
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'burg'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'capitals'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'cite_lakes'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'den'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'giant_polygon'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'grass'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'green'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'line'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'poi'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'point'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'polygon'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'poly_landmarks'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'pophatch'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'population'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'rain'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'raster'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'restricted'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'simple_roads'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'simple_streams'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded style 'tiger_roads'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded default workspace cite
6 Jul 23:22:02 INFO [org.geoserver] - Loaded workspace 'cite'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded workspace 'it.geosolutions'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded workspace 'nunc'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded workspace 'sde'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded workspace 'sf'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded workspace 'tiger'
6 Jul 23:22:02 INFO [org.geoserver] - Loaded workspace 'topp'
6 Jul 23:22:03 INFO [org.geoserver] - Loaded store 'arcGridSample', enabled
6 Jul 23:22:03 INFO [org.geoserver] - Loaded coverage store 'arcGridSample'
6 Jul 23:22:05 INFO [org.geoserver] - Loaded coverage 'Arc_Sample', enabled
6 Jul 23:22:05 INFO [org.geoserver] - Loaded coverage 'arcGridSample'
6 Jul 23:22:06 INFO [org.geoserver] - Loaded layer 'Arc_Sample'
6 Jul 23:22:06 INFO [org.geoserver] - Loaded store 'ing_sample2', disabled
6 Jul 23:22:06 INFO [org.geoserver] - Loaded coverage store 'ing_sample2'
```

Fig: 5.3: Maven loading workspaces of GeoServer

```
INFO: maven summary
INFO: GeoServer ..... SUCCESS [ 4.804 s ]
INFO: Core Platform Module ..... SUCCESS [ 3.282 s ]
INFO: Open Web Service Module ..... SUCCESS [ 3.528 s ]
INFO: Main Module ..... SUCCESS [06:31 min]
INFO: GeoServer Security Modules ..... SUCCESS [ 0.043 s ]
INFO: GeoServer JDBC Security Module ..... SUCCESS [ 26.495 s ]
INFO: GeoServer LDAP Security Module ..... SUCCESS [05:22 min]
INFO: Web Coverage Service Module ..... SUCCESS [ 1.198 s ]
INFO: Web Coverage Service 1.0 Module ..... SUCCESS [ 22.946 s ]
INFO: Web Coverage Service 1.1 Module ..... SUCCESS [ 35.083 s ]
INFO: Web Coverage Service 2.0 Module ..... SUCCESS [ 1.124 s ]
INFO: Web Feature Service Module ..... SUCCESS [ 11.181 s ]
INFO: Web Map Service Module ..... SUCCESS [01:46 min]
INFO: KML support for GeoServer ..... SUCCESS [ 11.339 s ]
INFO: GeoWebCache (GWC) Module ..... SUCCESS [02:54 min]
INFO: REST Support Module ..... SUCCESS [ 0.433 s ]
INFO: REST Configuration Service Module ..... SUCCESS [ 9.281 s ]
INFO: GeoServer Web Modules ..... SUCCESS [ 0.039 s ]
INFO: Core UI Module ..... SUCCESS [01:44 min]
INFO: MFS UI Module ..... SUCCESS [ 0.554 s ]
INFO: GWC UI Module ..... SUCCESS [ 5.629 s ]
INFO: MFS UI Module ..... SUCCESS [ 0.413 s ]
INFO: Demoes Module ..... SUCCESS [ 35.364 s ]
INFO: WCS UI Module ..... SUCCESS [ 0.669 s ]
INFO: Security UI Modules ..... SUCCESS [ 0.029 s ]
INFO: Security UI Core Module ..... SUCCESS [ 0.444 s ]
INFO: Security UI JDBC Module ..... SUCCESS [ 0.528 s ]
INFO: Security UI LDAP Module ..... SUCCESS [ 0.579 s ]
INFO: REST UI Module ..... SUCCESS [ 0.461 s ]
INFO: GeoServer Web Application ..... SUCCESS [ 27.943 s ]
INFO: Community Space ..... SUCCESS [ 0.046 s ]
INFO: GeoServer Extensions ..... SUCCESS [ 0.035 s ]
INFO:
INFO: BUILD SUCCESS
INFO:
INFO: Total time: 21:51 min
INFO: Finished at: 2014-07-15T00:12:43+05:30
INFO: Final Memory: 75M/100M
```

Fig: 5.4: Web Achieve successfully built by GeoServer

Figure (5.4) shows that the maven has built the GeoServer on the local system successfully. So as to develop a process, it is recommended to configure all the necessary plugins on a common interface so that interoperability issues wouldn't arise. Thus, we use eclipse IDE (Kepler) as an interface which connects, Maven tool, Apache Tomcat server and Java all integrated on a common environment/platform.

5.1.3. Eclipse IDE with Maven and Tomcat

Java Software Development Kit is installed referring [42] followed by Eclipse IDE , followed by Tomcat and Maven installations subsequently integrating them within Eclipse.

5.1.3.1. Tomcat Installation

Apache Tomcat Server version 7.0 was downloaded and installed by following instructions from [43], and the port number is set for local host. Server was configured and checked by opening local host in browser with defined port number (fig:5.5).

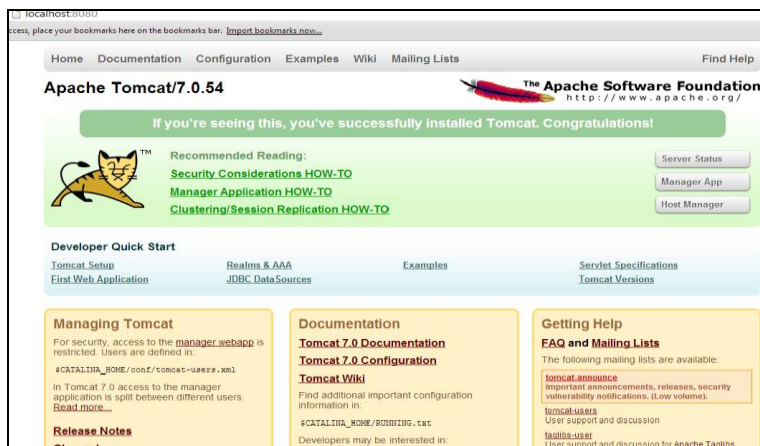


Fig: 5.5. Verifying the successful installation of Tomcat

5.1.3.2. Integrating Tomcat with Eclipse

In order to integrate tomcat with eclipse, the document [44] was referred. By selecting New, Server, Apache, Tomcat v7.0, then by navigating to the folder where tomcat has been already installed (*C:\apache-tomcat-7.0.34*) and port numbers are configured as shown in 5.7.

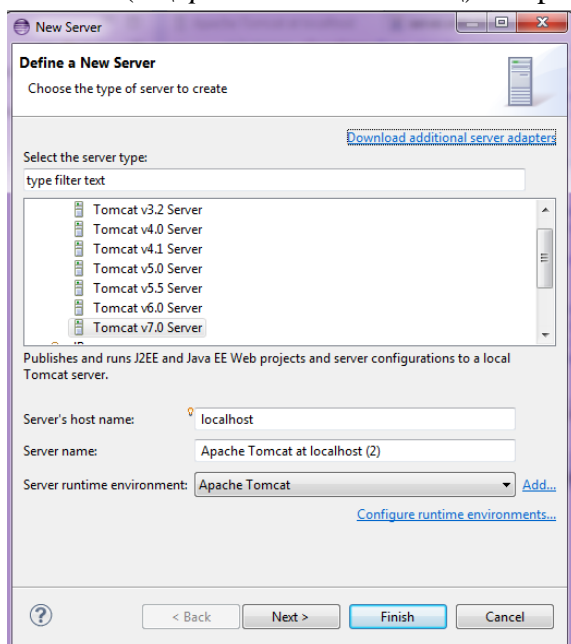


Fig: 5.6: Integrating Tomcat with Eclipse

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

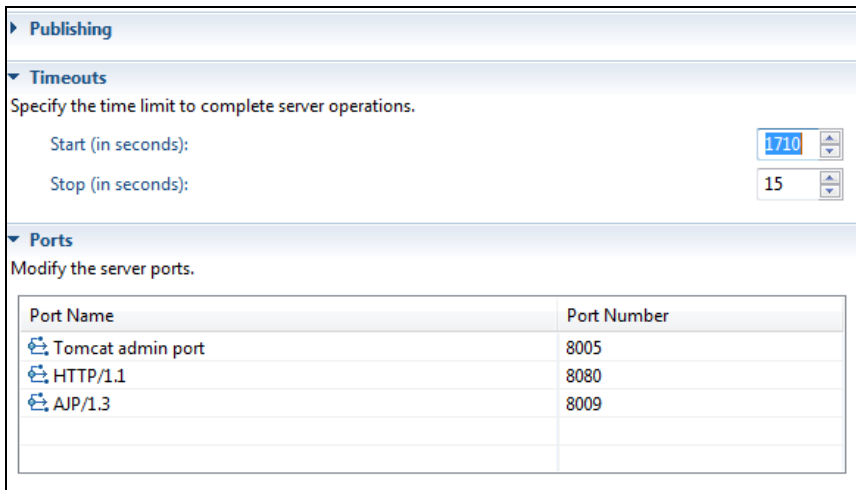


Fig: 5.7: Setting the port numbers for Tomcat within the Eclipse

Setting up the port numbers accordingly it is verified if tomcat is properly integrated with eclipse as per screenshot mentioned in figure (5.8) which shows tomcat has successfully integrated with eclipse.

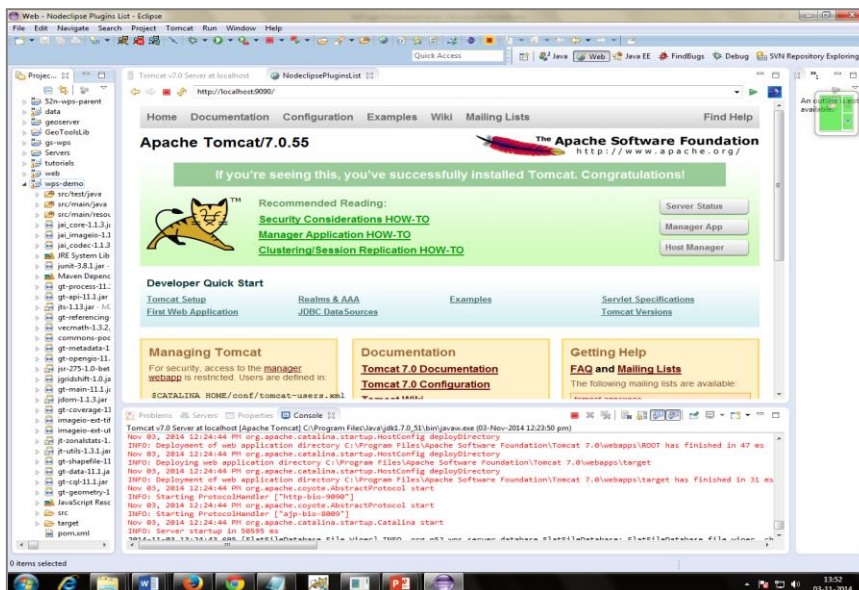


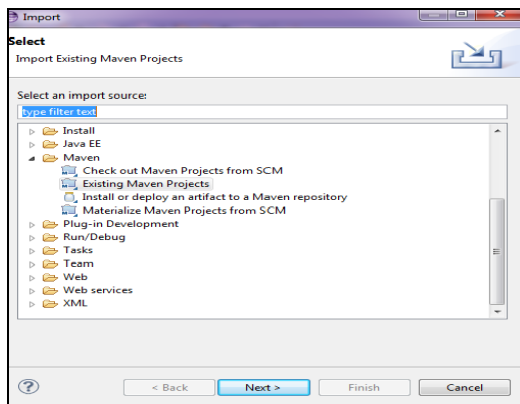
Fig: 5.8: Tomcat successfully installed within Eclipse

5.1.3.3. Maven Installation & integration with eclipse

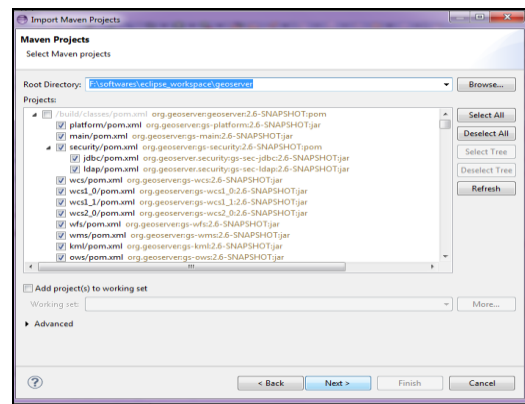
Apache Maven is downloaded from [45], and installed as per instructions. Though maven can build projects externally, in order to import maven projects into eclipse, the plugin m2e (maven2eclipse) must be installed. So the plugin was downloaded from Eclipse marketplaces and installed. After successful installation, the “maven project” options would be shown in import option as shown in the figure(5.9(a)) and target the projects built by maven externally can be

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

imported into eclipse directly or even let the maven2eclipse plugin build the project internally. We chose the initial procedure, by externally importing into eclipse as shown in the figure (5.9(b)).



(a)

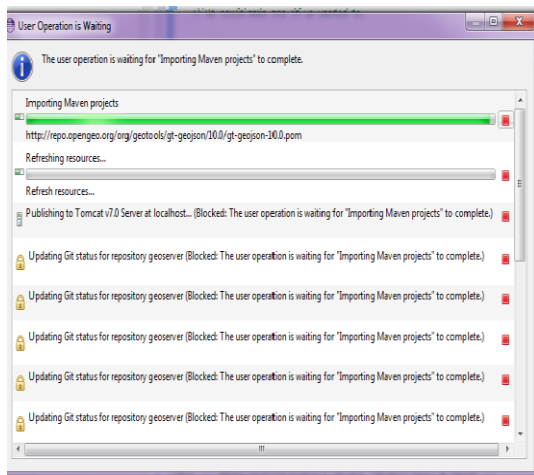


(b)

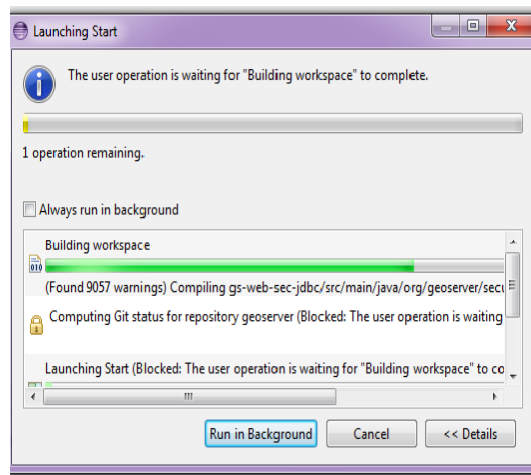
Fig: 5.9:

- (a): Successful installation of maven2eclipse plugin
- (b): Importing already built maven project externally

The figure (5.9(a)) shows that maven plugins for Eclipse has been loaded appropriately. And only if the maven-eclipse plugins are loaded appropriately, we get an option to choose the existing maven projects into eclipse. As the GeoServer project that is externally built by maven is imported into eclipse, the workspaces repeat building within eclipse in order to launch the project. The figures (5.10(a) & (b)) show how the maven project is fetching its resources and launching the start after building the workspaces respectively.



(a)



(b)

Fig: 5.10:

- (a): Maven fetching resources to build GeoServer
- (b): Building workspaces for launching the GeoServer project

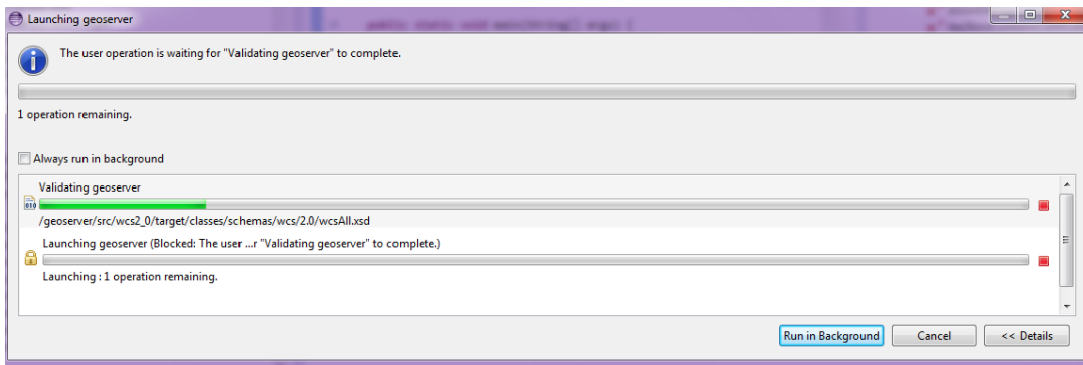


Fig: 5.11: Launching and Validating the GeoServer

When launching the project after building, the project connects to the apache tomcat server that is already running as shown in the figure (5.11).

On successful launch, GeoServer will be integrated within the eclipse (fig: 5.12) and the GeoServer main page will be accessed through eclipse.

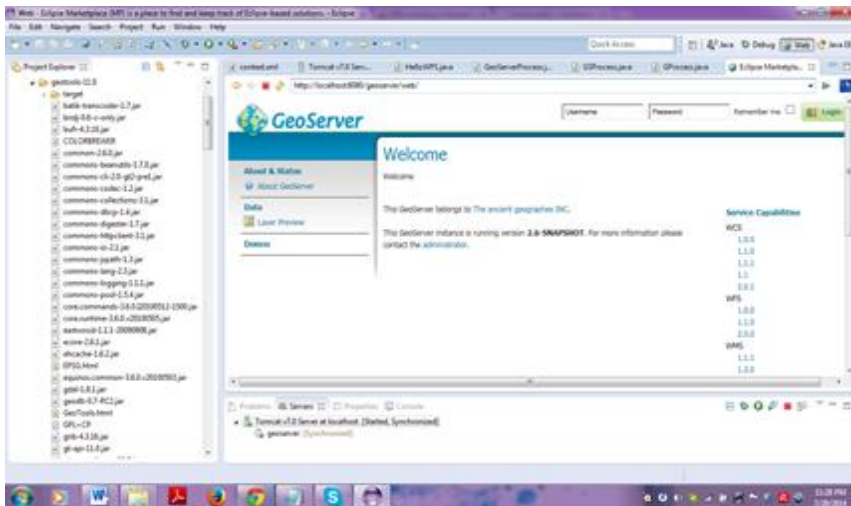


Fig: 5.12: Verification of successful installation of GeoServer on eclipse

5.1.4. Develop a Custom Process

Once GeoServer is successfully installed, our next step is to deploy MaxEnt species distribution modelling technique as process into GeoServer. For this, we first tried with a sample process referring to the document [46]. Thus for even more capability, GeoServer is extended by adding new WPS processes. Hence, the section below describes how a sample WPS process is created in Java and deployed into GeoServer.

5.1.4.1. Creating a java process class

A process is a Java class that provides a method which accepts parameters corresponding to the WPS and it returns a value which becomes the output of the process. As we discussed earlier, Maven tool has been used to build the project and manage its dependencies and we used GeoTools library as it offers a convenient framework for building process classes with

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

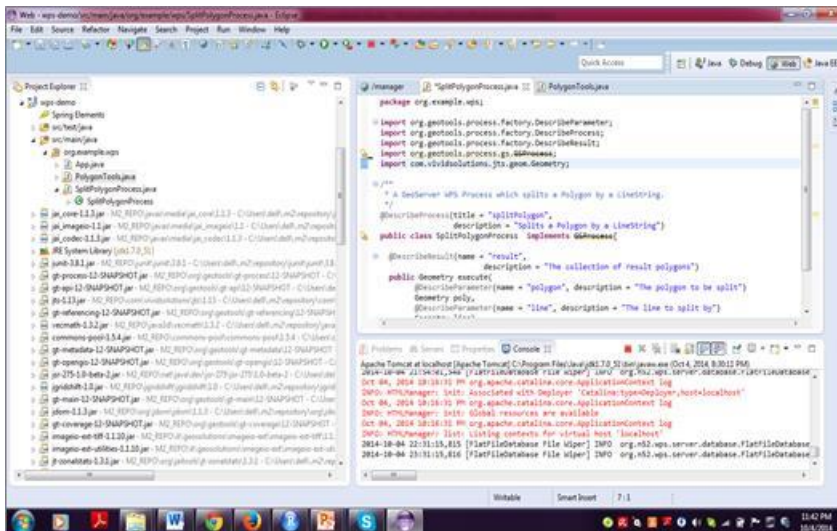


Fig: 5.13: Creating a Java process on Eclipse IDE

minimal coding. The GeoTools Process API allows GeoServer takes in many powerful libraries that could be used in building a process. When we needed external libraries, we have deployed them with in the process along with the process. The figure (5.14) shows that, a new Java project was created, using eclipse IDE. Archetypes are used by maven to build the project and POM files to explain the project dependencies.

5.1.4.2. Register the process with Spring Framework, Building the project and Deploying into publishing software

Maven builds the project and manages the dependencies whereas spring framework is used by GeoServer to manage components. To discover the new process, we need to configure it as a Spring Bean by providing a unique identification and register the process class. The following screenshot shows how the sample process is deployed in GeoServer. Once the registration is done, the project has to be packaged into a Java Archive (JAR) file by Maven. Hence the following command builds the project “**mvn clean install**”. This cleans up artifacts from previous builds, compiles the code, and executes available unit tests and the resultant process JAR file is created in the target directory. The resultant JAR file is deployed into Tomcat server on which GeoServer is running.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN" "http://www.springframework.org/dtd/spring-beans.dtd">
<beans>
  <bean id="SplitPolygon" class="org.example.wps.SplitPolygon"/>
</beans>
```

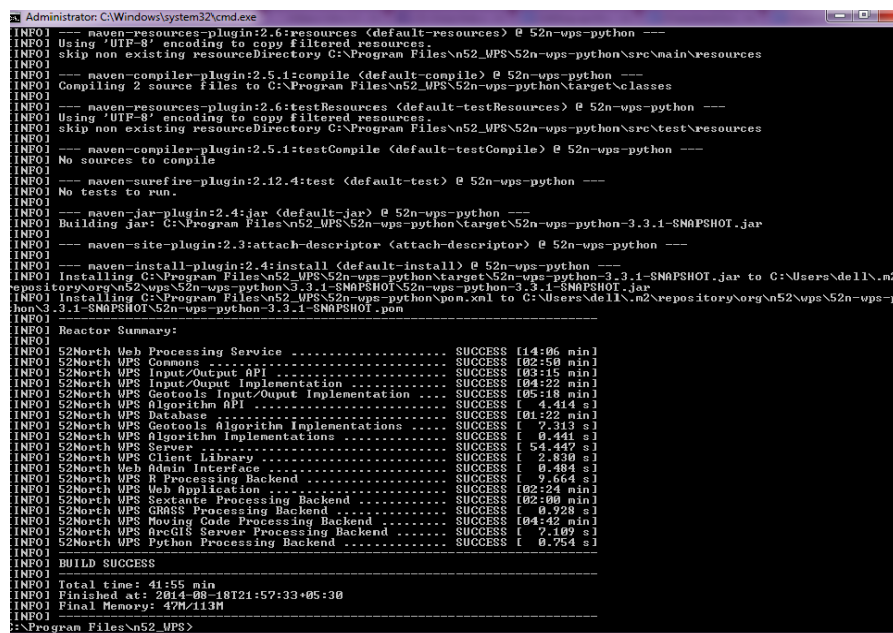
5.1.4.3.. Checking the availability of customized sample process being deployed into GeoServer

To verify whether the process is successfully registered with the GeoServer, the server needs to be restarted, and check the availability of processes using “GetCapabilities” request or using WPSRequestBuilder which displays all the registered processes in the drop-down box to choose. This is covered in the first session of sixth chapter (Results and Discussions).

5.2. Second Approach - 52° North WPS

5.2.1. Configuring 52° North WPS

As mentioned in the section 5.1, the source code of 52° North WPS was accessed through Git (Refer 5.1.1), and Maven tool (5.1.2) was used to build the 52° North WPS project as well.



```
Administrator: C:\Windows\system32\cmd.exe
INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ 52n-wps-python ---
INFO] Using 'UTF-8' encoding to copy filtered resources.
INFO] skip non existing resourceDirectory C:\Program Files\52_WPS\52n-wps-python\src\main\resources
INFO]
INFO] --- maven-compiler-plugin:2.5.1:compile (default-compile) @ 52n-wps-python ---
INFO] Compiling 2 source files to C:\Program Files\52_WPS\52n-wps-python\target\classes
INFO]
INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ 52n-wps-python ---
INFO] Using 'UTF-8' encoding to copy filtered resources.
INFO] skip non existing resourceDirectory C:\Program Files\52_WPS\52n-wps-python\src\test\resources
INFO]
INFO] --- maven-compiler-plugin:2.5.1:testCompile (default-testCompile) @ 52n-wps-python ---
INFO] No sources to compile
INFO]
INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ 52n-wps-python ---
INFO] No tests to run.
INFO]
INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ 52n-wps-python ---
INFO] Building jar: C:\Program Files\52_WPS\52n-wps-python\target\52n-wps-python-3.3.1-SNAPSHOT.jar
INFO]
INFO] --- maven-site-plugin:2.3:attach-descriptor (attach-descriptor) @ 52n-wps-python ---
INFO]
INFO] --- maven-install-plugin:2.4:install (default-install) @ 52n-wps-python ---
INFO] Installing C:\Program Files\52_WPS\52n-wps-python\target\52n-wps-python-3.3.1-SNAPSHOT.jar to C:\Users\dell\.m2\repository\org\52n-wps\52n-wps-python\3.3.1-SNAPSHOT\52n-wps-python-3.3.1-SNAPSHOT.jar
INFO] Installing C:\Program Files\52_WPS\52n-wps-python\pom.xml to C:\Users\dell\.m2\repository\org\52n-wps\52n-wps-python\3.3.1-SNAPSHOT\52n-wps-python-3.3.1-SNAPSHOT.pom
INFO]
INFO] Reactor Summary:
INFO] 52North Web Processing Service ..... SUCCESS [14:06 min]
INFO] 52North WPS Commons ..... SUCCESS [02:58 min]
INFO] 52North WPS Input/Output API ..... SUCCESS [03:15 min]
INFO] 52North WPS Input/Output Implementation ..... SUCCESS [04:22 min]
INFO] 52North WPS Geotools Input/Output Implementation ..... SUCCESS [05:18 min]
INFO] 52North WPS Algorithm API ..... SUCCESS [ 4.414 s]
INFO] 52North WPS Database ..... SUCCESS [01:22 min]
INFO] 52North WPS Geotools Algorithm Implementations ..... SUCCESS [ 7.313 s]
INFO] 52North WPS Algorithm Implementations ..... SUCCESS [ 0.441 s]
INFO] 52North WPS Server ..... SUCCESS [ 54.447 s]
INFO] 52North WPS Client Library ..... SUCCESS [ 2.838 s]
INFO] 52North Web Admin Interface ..... SUCCESS [ 0.484 s]
INFO] 52North WPS R Processing Backend ..... SUCCESS [ 9.664 s]
INFO] 52North WPS Web Application ..... SUCCESS [02:24 min]
INFO] 52North WPS Sextante Processing Backend ..... SUCCESS [02:00 min]
INFO] 52North WPS GRASS Processing Backend ..... SUCCESS [ 0.928 s]
INFO] 52North WPS Moving Code Processing Backend ..... SUCCESS [04:42 min]
INFO] 52North WPS ArcGIS Server Processing Backend ..... SUCCESS [ 7.109 s]
INFO] 52North WPS Python Processing Backend ..... SUCCESS [ 0.754 s]
INFO]
INFO] BUILD SUCCESS
INFO]
INFO] Total time: 41:55 min
INFO] Finished at: 2014-08-18T21:57:33+05:30
INFO] Final Memory: 47M/113M
INFO]
C:\Program Files\52_WPS>
```

Fig: 5.14: Successful build of 52° North WPS

The figure (5.14) shows that Maven has built 52° North WPS successfully on local system and subsequently, this has to be integrated with Eclipse. Thus it is integrated with Eclipse and built successfully as shown in the figure (5.15).

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

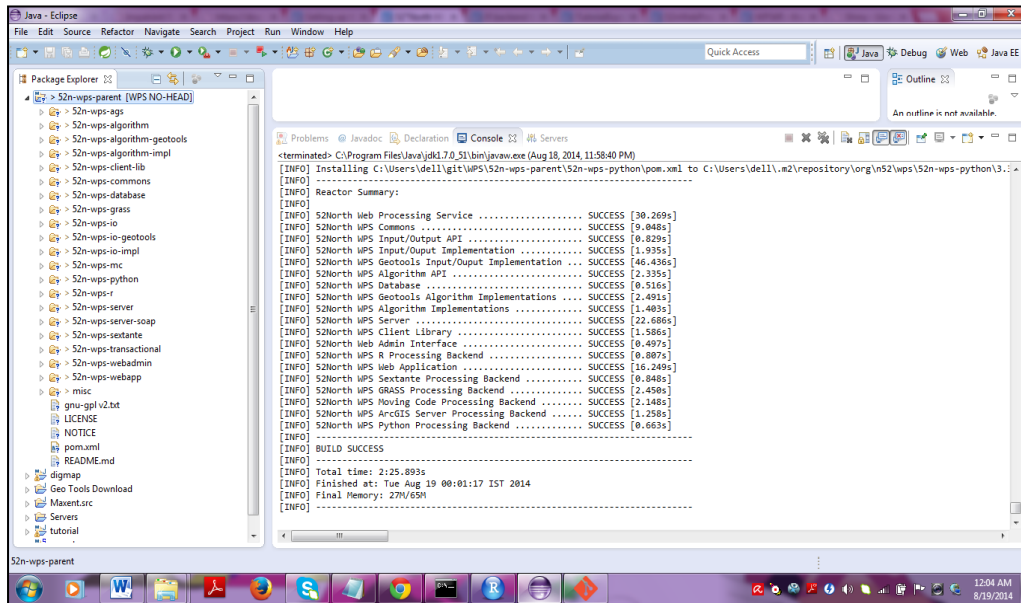


Fig: 5.15: 52⁰North WPS successfully built on Eclipse

Once the m2e plugin had successfully built the project, it is verified by running. As it runs the following figure (5.15) showing the repositories initialized and algorithms registered along with other details about web path set and context information.

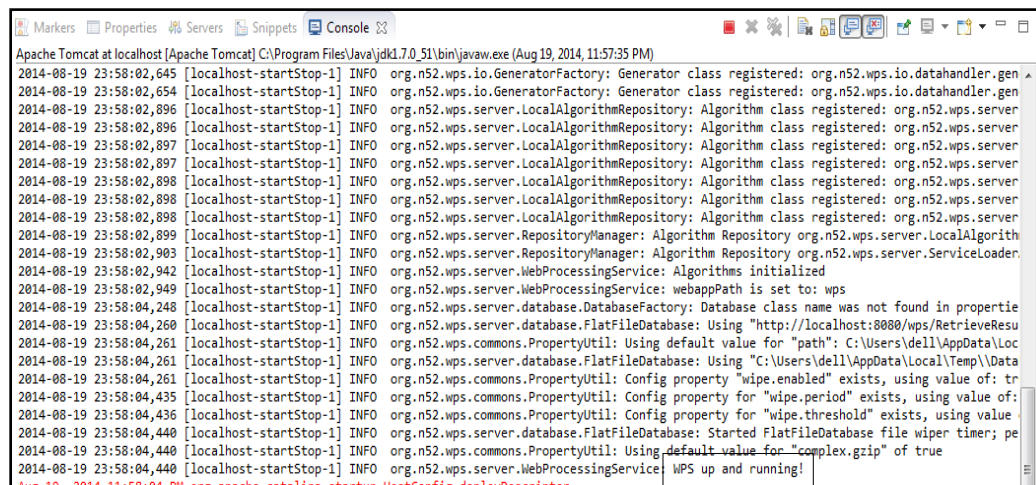


Fig: 5.16: 52⁰ North WPS Running

As a result, the main page appears as shown in the figure (5.16). If the configurations of new repositories are to be initialized, webAdmin console must be accessed and updated according to the requirement. Access of webAdmin console and deploying of script is explained in the Results session of chapter 6.

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

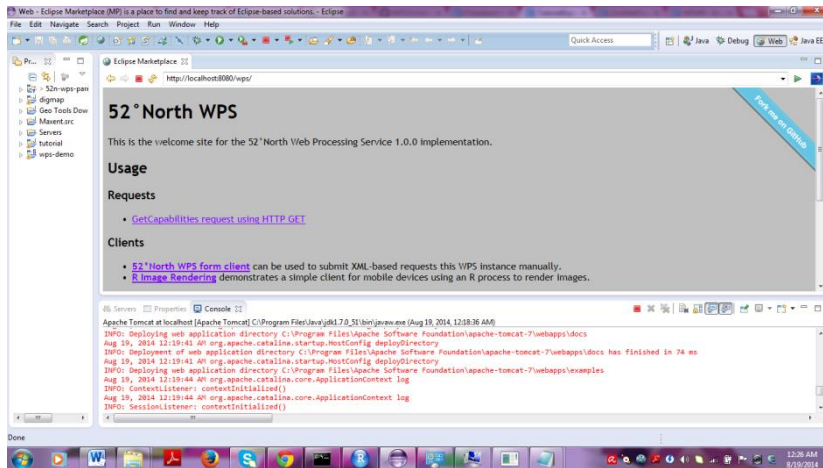


Fig: 5.17: Verification of successful installation of 52° North WPS

5.2.2. Implementation of Species Distribution Model Using Maximum Entropy Modelling Technique for *Hippophae Rhamnoides* in Lahaul-Spiti

The main purpose of the project was to expose a MaxEnt species distribution modelling technique as a Web Processing Service. With a set of packages anchored by the package ‘dismo’ in R, we use R statistical programming and WorldClim environmental data to determine the environmental conditions of the species. Species distribution models (SDMs) explains in general where a species might be based on where you know it actually is. The basic idea is, a list of locations where we know we can find a species and identify the environmental conditions at those locations; subsequently build a statistical model that differentiates the climate at the points where the species is found from other points where species isn’t found; and apply the model to climate from some other time or place to estimate a probability that the species growth would be suitable. ‘Dismo’ is a package exclusively for performing species distribution modelling in R. we need environmental data and species occurrence points for predicting the distribution of species using MaxEnt technique, so the environmental data was collected from WorldClim, all bioclim variables (0-19) as mentioned in the table no (5.1) at a resolution of 30 arc seconds, minimum, maximum and mean temperature and minimum, maximum and mean precipitation of October month since the amount of vegetation is expected to be high in the area Lahaul-Spiti located at Himachal Pradesh, India. The two valleys are quite different in character. Spiti is more barren and with an average elevation of the valley floor of 4,270 m (14,009 feet). It is enclosed between lofty ranges, with the Spiti River rushing out of a valley in the southeast to meet the Sutlej River. It is a typical mountain desert area with an average annual rainfall of only 170 mm (6.7 inches). So the impact of amount of temperature is high on the species growing in this region.

Table: 5.1: A tabular column describing bioclim variables and its significance

BIO1	Annual Mean Temperature
BIO2	Mean Diurnal Range (Mean of monthly (max temp - min temp))
BIO3	Isothermality (BIO2/BIO7) (* 100)
BIO4	Temperature Seasonality (standard deviation *100)

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

BIO5	Max Temperature of Warmest Month
BIO6	Min Temperature of Coldest Month
BIO7	Temperature Annual Range (BIO5-BIO6)
BIO8	Mean Temperature of Wettest Quarter
BIO9	Mean Temperature of Driest Quarter
BIO10	Mean Temperature of Warmest Quarter
BIO11	Mean Temperature of Coldest Quarter
BIO12	Annual Precipitation
BIO13	Precipitation of Wettest Month
BIO14	Precipitation of Driest Month
BIO15	Precipitation Seasonality (Coefficient of Variation)
BIO16	Precipitation of Wettest Quarter
BIO17	Precipitation of Driest Quarter
BIO18	Precipitation of Warmest Quarter
BIO19	Precipitation of Coldest Quarter

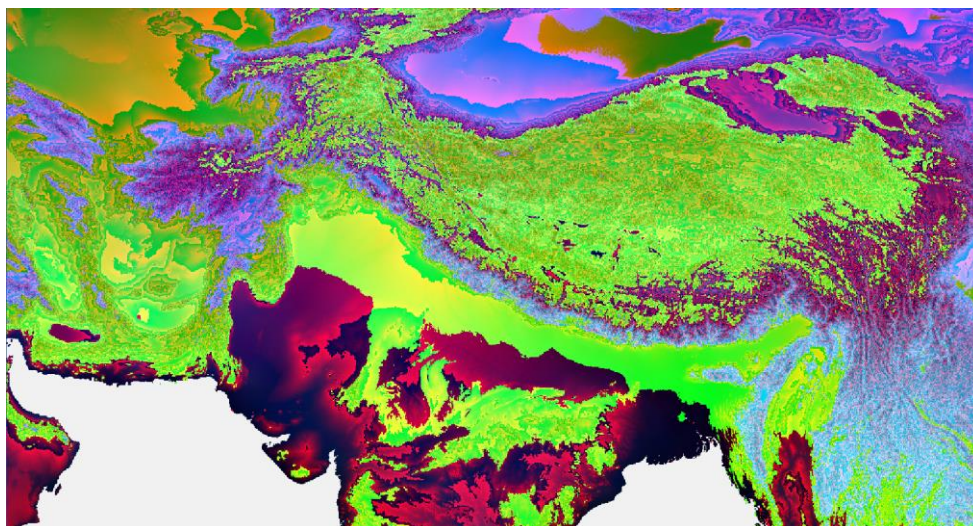


Fig: 5.18: Stack of Environmental variables used in the project

All the environmental layers collected were stacked and was subset to our study area Lahaul Spiti using ERDAS imagine and projected with respect to the reference data collected from GLCF with “WGS-84” datum 43N region. Since MaxEnt takes environmental layers only in ASCII format the layers are converted to ASCII format using ArcGIS.

The figure below (5.20) shows the stacked environmental data with respect to GLCF data

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

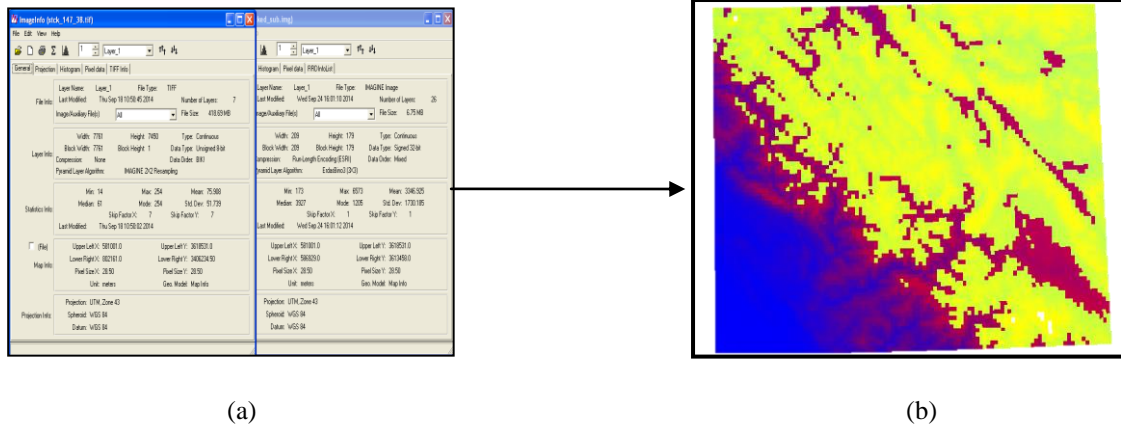


Fig: 5.19:
 (a): Re-projection with respect to GLCF data (reference data)
 (b): Resultant image after re-projection

After sub-setting the reprojected data, it is now clipped with the shape file (as shown in the figure (5.19)) of our study area and consequently, the clipped stacked reprojected environmental layers would look as shown in the figure below (5.20(b)).

To perform species distribution modelling, Fifty seven *Hippophae Rhamnoides*' occurrence points were collected from GBIF and Biodiversity project of National Remote Sensing Centre. These points were divided into two sets, training and testing sets, out of which, forty were divided for training and the rest testing.

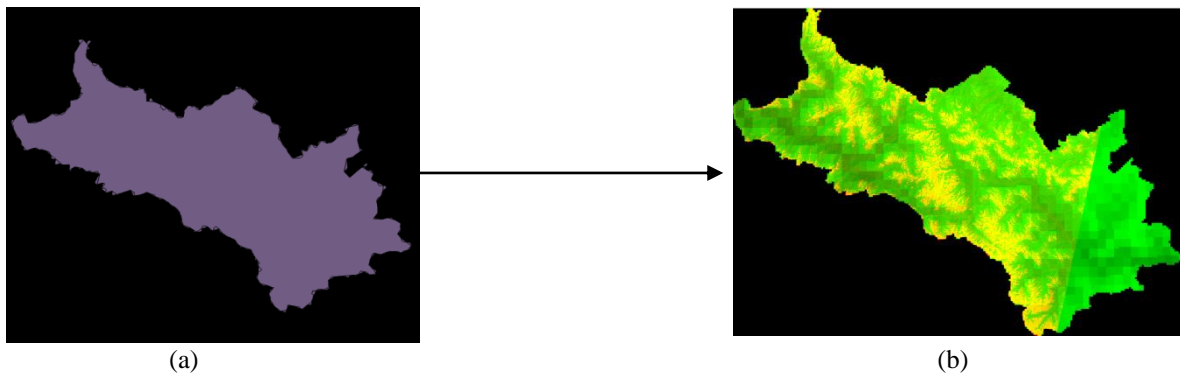


Fig: 5.20:
 (a) : Shape file of the study area – Lahaul Spiti
 (b) : Clipping of Environmental data with respect to study area

These points were spread on the study area so as to verify as shown in the figure (5.21(a)) and the figure (5.21(b)) shows the species spread per pixel.

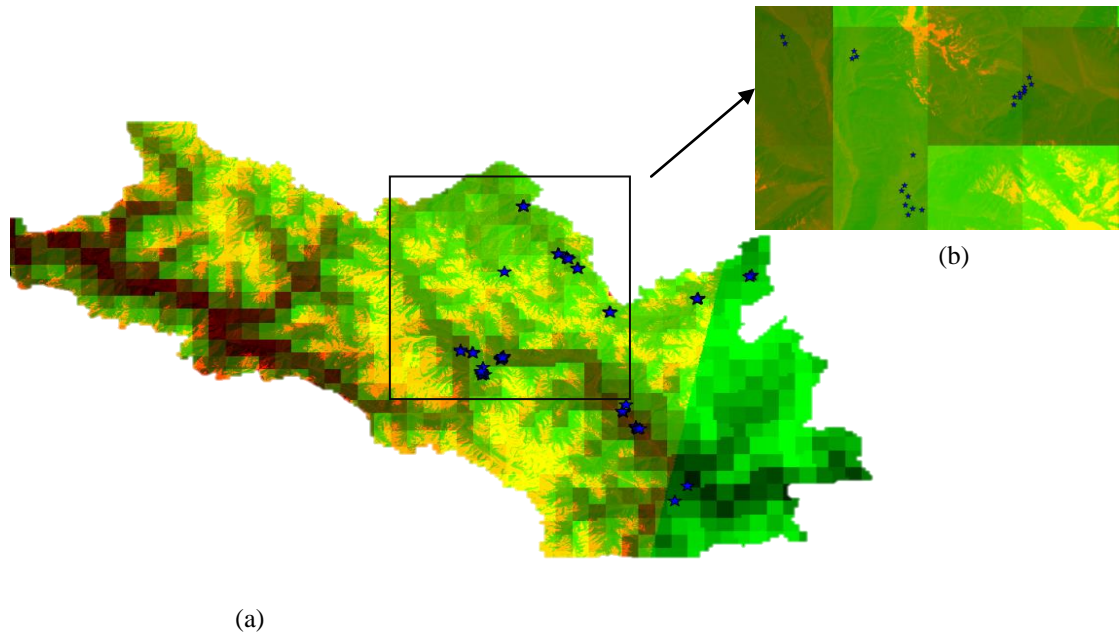


Fig: 5.21:

(a) Points spread on study area

(b) Species points per pixel

The results and discussions session shows the final results performed by MaxEnt species distribution model using R. We have called the MaxEnt utility through a dismo package in R, and scripted a program that runs the MaxEnt software, which takes environmental data as input, evaluates the model, and distributes the species and estimate the species distribution based on the principle of maximum entropy.

6. RESULTS AND DISCUSSIONS

6.1. Deployment of a sample process class using GeoServer

GeoServer has some processes by default that can help us test the way WPS works. This gives us a basic idea of how a process can be developed. The fig (6.1(a)) shows the default processes provided by GeoServer. In order to develop a customized process, the source code must be registered with spring beans, which uniquely identifies each process. Once the process is developed, registered to server and successfully deployed as explained in section 5.1, the deployed process will now be available as a service for the clients along with the default services provided by GeoServer as shown in fig (6.1(b)). Now, the process that serves the purpose can be selected in the Process selection session, where the required process can be selected by checking-in the box and applied.

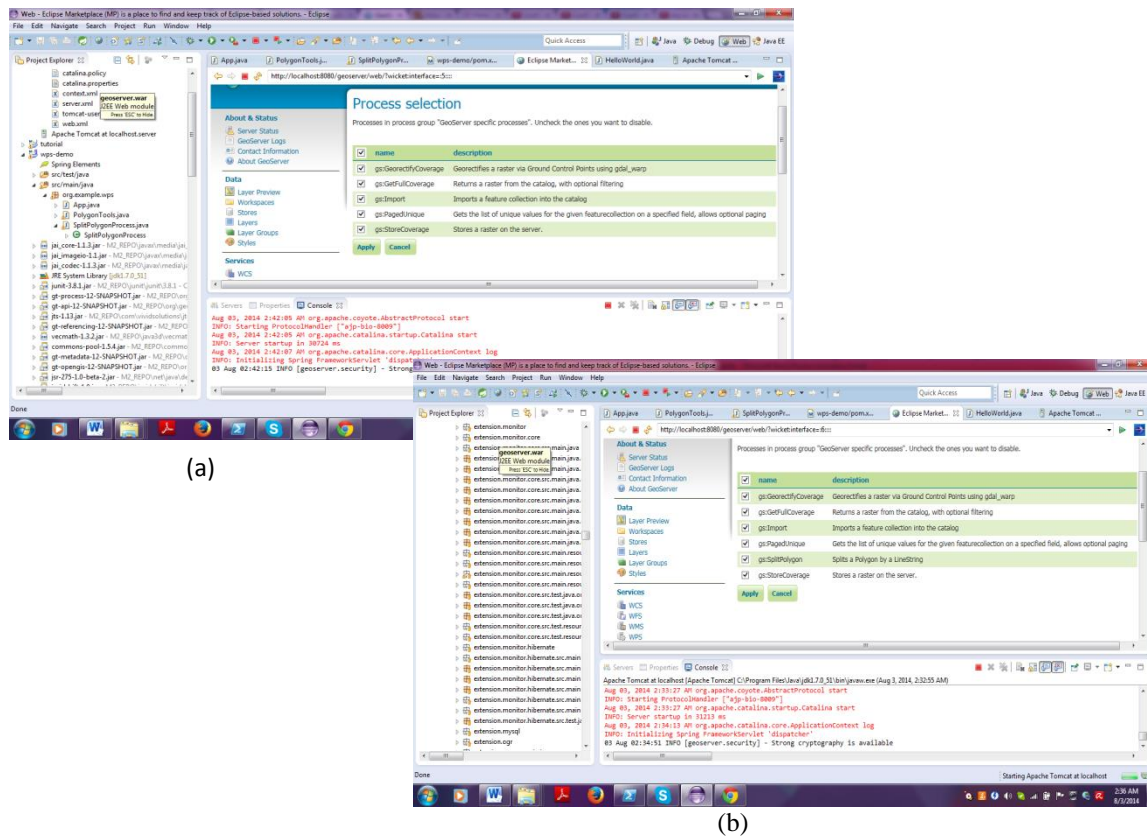


Fig: 6.1:

- (a): List of Processes available before deploying the customized process
- (b): List of Processes available after deploying the customized process

Now the selected process will be available in the drop-down box when the capabilities of the servers are requested by the client, as shown in fig (6.2), once the process is selected, the next phase being “describe process”, the process will be described as shown in fig (6.3), and finally executed as shown in the figure (6.4).

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

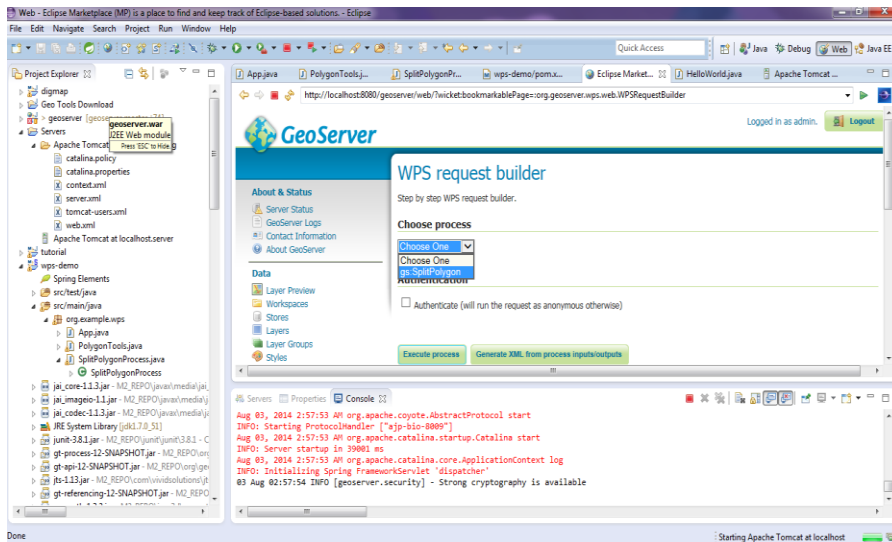


Fig. 6.2: Customized process available for selection

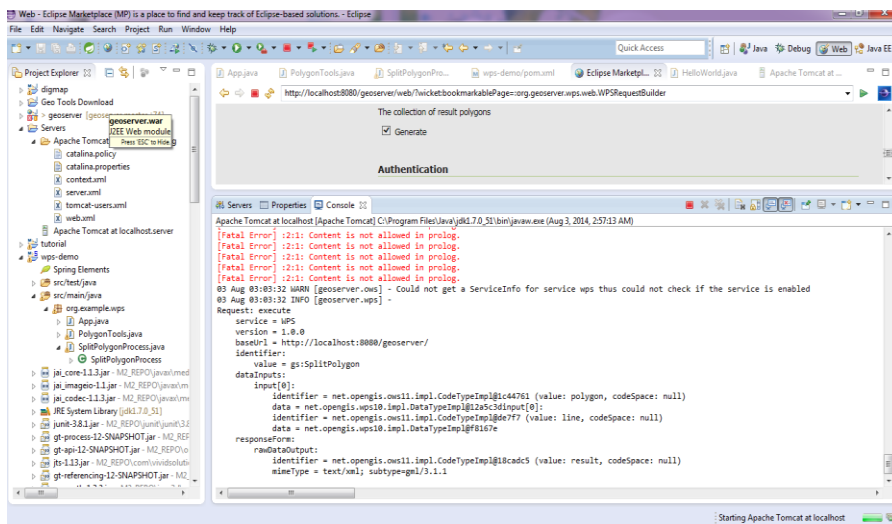


Fig. 6.3: After selection of process, Request is sent to the server internally

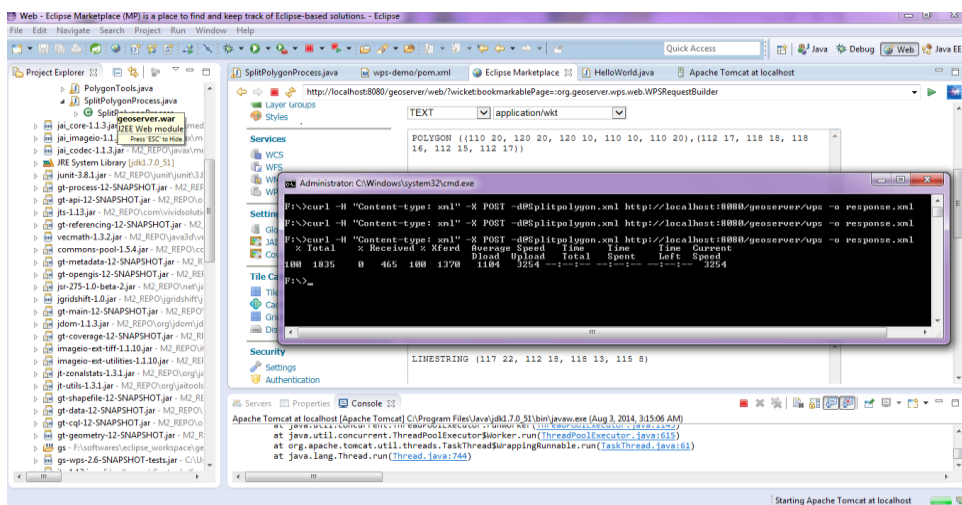


Fig. 6.4: Description and Execution of process

6.2. Limitations with GeoServer with respect to Web Processing Services

In order to provide MaxEnt species distribution modelling as a service, MaxEnt code should be registered to the GeoServer through spring beans. MaxEnt, being neither a proprietary

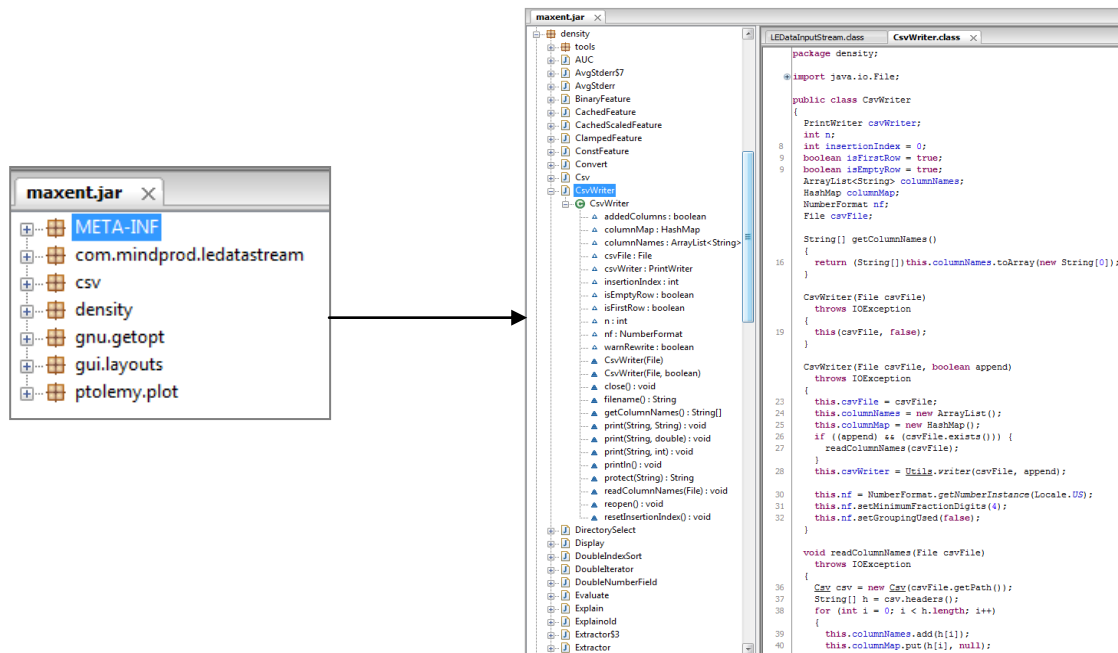


Fig: 6.5: Contents in the jar file (on left) shows no MaxEnt source files are accessible

software, nor open source project. It is just free desktop software, where the complete species distribution modelling is packaged to a Java archive, which runs on any system if java is pre-installed. We need the source code of MaxEnt in order to register the process with GeoServer but MaxEnt jar file doesn't provide any source code and only few packages along with class files are accessible as shown in the figure (6.5).

The above figure shows the available class files that are accessible among which, there are no source files. Thus, in order to produce MaxEnt as a process, either the entire software should be re-coded or the source files must be accessed. We requested the MaxEnt group for source files, but they expressed that it was a combined project and no single person is authorized to re-distribute the source code even if it was for an educational purpose and even after ensuring no commercial benefits will be claimed. May be it would take a while to convert this software into an open source project and make the source code available. When a process is to be deployed into GeoServer, until and unless the source code is registered with the GeoServer, the process cannot be available for the clients. We then explored alternatives to perform MaxEnt other than running the software alone. Thus, we learnt that MaxEnt software can be called through R interface using a dedicated package 'dismo' for species distribution modelling. But GeoServer hasn't got any backend supporting R; it can rather support Python and GRASS backend. So as to achieve that, we need an open source that can connect R programming language with a web service. As mentioned earlier, we have few more open source projects too that can serve the web service functionalities other than GeoServer, like 52⁰North WPS, PyWPS, Zoo and degree projects. All of these projects too have similar functionalities similar to GeoServer, with certain pros and cons. Among these, R statistical

programming can be connected to 52North WPS through an extended module. Thus our next phase of the research deals with exploration of 52⁰ North WPS; develop a process that can run MaxEnt software as a service. Accordingly, 52⁰North WPS with an R extension module was configured on a local system and is described in next session in detail.

6.3. Configuring 52⁰North WPS

On successful configuration of 52⁰North WPS (refer fig: 5.18), the default login page is accessed through eclipse. The user name and password should to be submitted to the server so as to allow the access to the web admin console as shown in the figure (6.6)

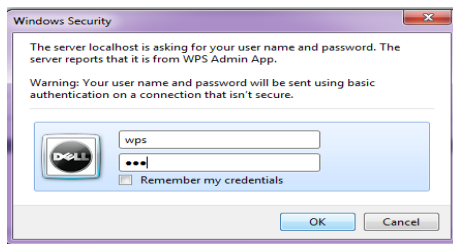


Fig:6.6: Submitting credentials to 52⁰North WPS in order to access the webAdmin console

The default repository doesnot contain any extended modules as shown in the figure (6.7) and contains only few default processes as shown in figure (6.8).

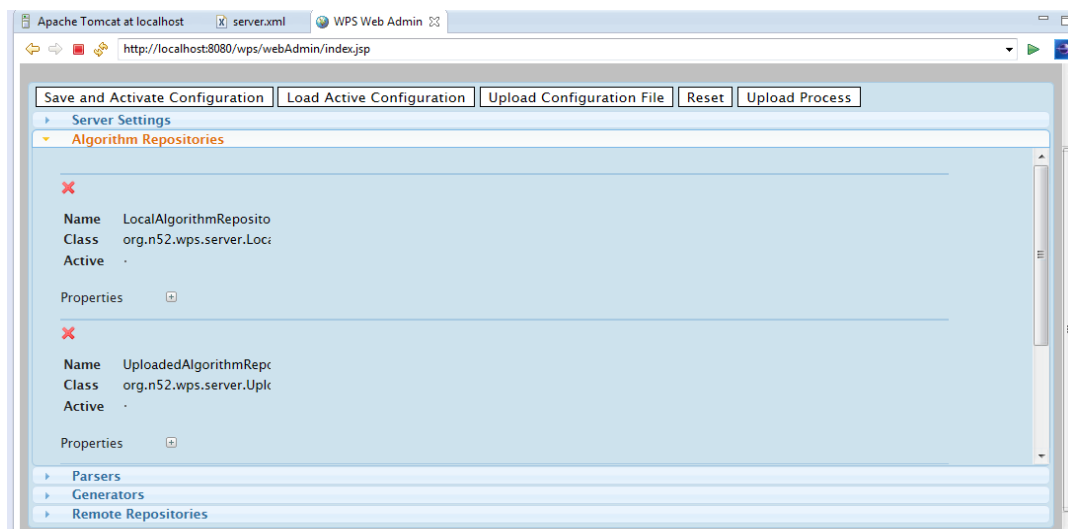


Fig:6.7: Default repositories in 52⁰North WPS

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

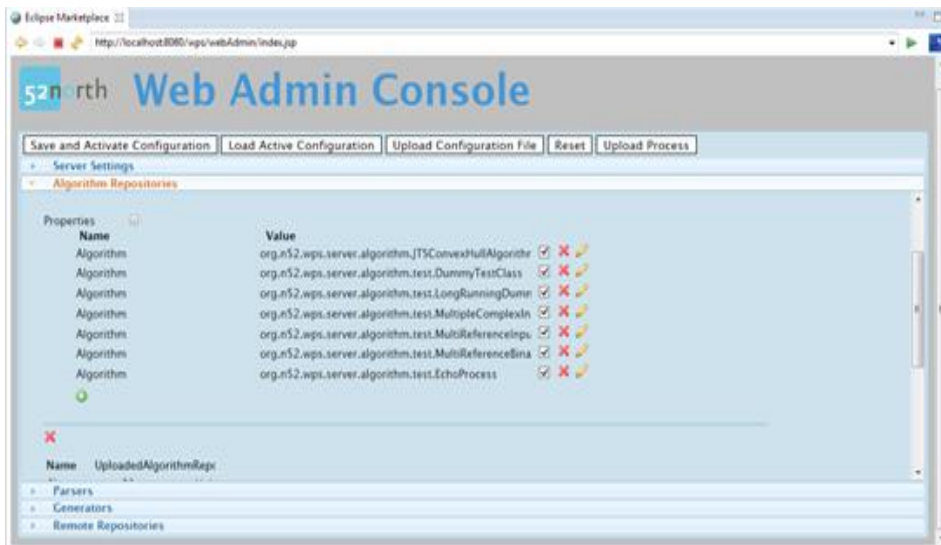


Fig: 6.8: Default processes available in LocalAlgorithm repository in 52⁰North WPS

Our purpose was to execute the process using R as an interface, for this, we need to activate R repository that connects R software and 52⁰North WPS through an interface Rserve. Rserve acts as a socket server (TCP/IP sockets) which allows binary requests to be sent to R. This helps clients working on Java to use the facilities of R without the need of linking to R code. Rserve supports remote connection, user authentication and file transfer. To run Rserve (), initially path should be set up and check if the configuration is successful.

Testing Rserve connection using telnet connection, shown in figure (6.9(a)) shows Rserve is connected and is tested whether it is ready to answer the queries (fig: 6.9(b)).

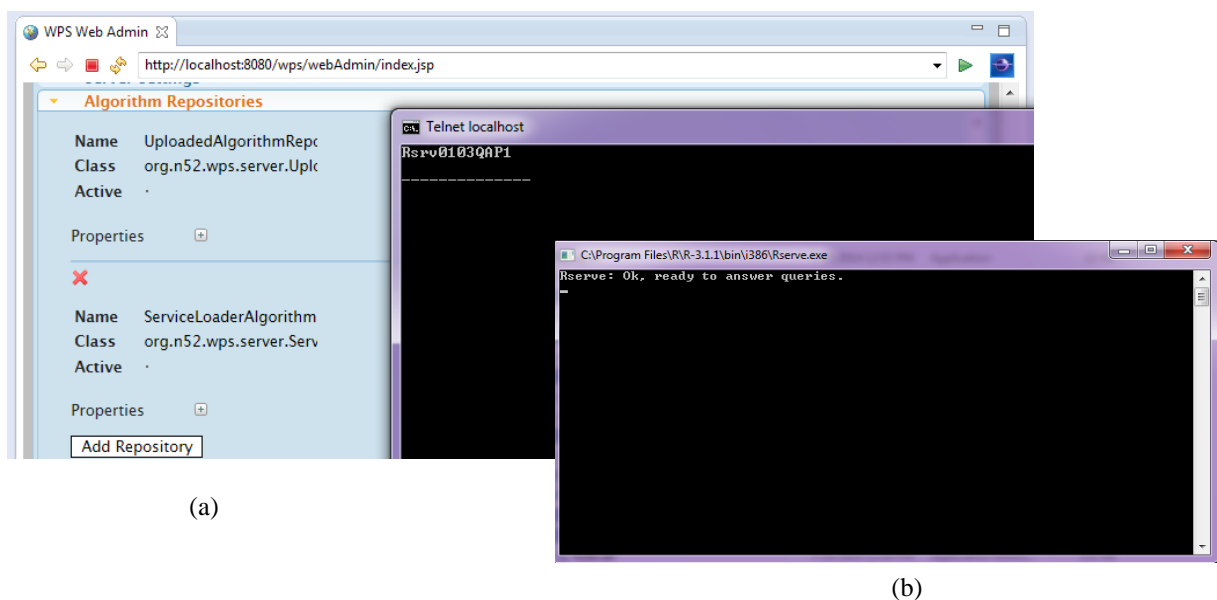


Fig: 6.9:
 (a): Testing the connection of Rserve to 52⁰North WPS with Telnet
 (b): Successful installation of Rserve

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

```
Apache Tomcat at localhost [Apache Tomcat] C:\Program Files\Java\jdk1.7.0_51\bin\javaw.exe (Aug 20, 2014, 12:20:37 AM)
014-08-20 00:26:08,697 [http-bio-8080-exec-1] INFO org.n52.wps.server.LocalAlgorithmRepository: Algorithm class registered: org.n52.wps.server.algorithm.test.MultiRefe
014-08-20 00:26:08,697 [http-bio-8080-exec-1] INFO org.n52.wps.server.LocalAlgorithmRepository: Algorithm class registered: org.n52.wps.server.algorithm.test.MultipleC
014-08-20 00:26:08,697 [http-bio-8080-exec-1] INFO org.n52.wps.server.RepositoryManager: Algorithm Repository org.n52.wps.server.LocalAlgorithmRepository initialized
014-08-20 00:26:08,698 [http-bio-8080-exec-1] INFO org.n52.wps.server.RepositoryManager: Algorithm Repository org.n52.wps.server.ServiceLoaderAlgorithmRepository initi
014-08-20 00:26:08,699 [http-bio-8080-exec-1] WARN org.n52.wps.server.RepositoryManager: An error ocured while registering AlgorithmRepository: org.n52.wps.server.r.l
014-08-20 00:26:08,699 [http-bio-8080-exec-1] INFO org.n52.wps.io.GeneratorFactory: org.n52.wps.io.GeneratorFactory$1: Received Property Change Event: WPSConfigUpdate
014-08-20 00:26:08,700 [http-bio-8080-exec-1] INFO org.n52.wps.io.GeneratorFactory: Generator class registered: org.n52.wps.io.datahandler.generator.NKTPGenerator
014-08-20 00:26:08,700 [http-bio-8080-exec-1] INFO org.n52.wps.io.GeneratorFactory: Generator class registered: org.n52.wps.io.datahandler.generator.GenericXMLDataGene
014-08-20 00:26:08,701 [http-bio-8080-exec-1] INFO org.n52.wps.io.GeneratorFactory: Generator class registered: org.n52.wps.io.datahandler.generator.GenericFileGenerat
014-08-20 00:26:08,701 [http-bio-8080-exec-1] INFO org.n52.wps.io.ParserFactory: org.n52.wps.io.ParserFactory$1: Received Property Change Event: WPSConfigUpdate
014-08-20 00:26:08,702 [http-bio-8080-exec-1] INFO org.n52.wps.io.ParserFactory: Parser class registered: org.n52.wps.io.datahandler.parser.NKTPParser
014-08-20 00:26:08,702 [http-bio-8080-exec-1] INFO org.n52.wps.io.ParserFactory: Parser class registered: org.n52.wps.io.datahandler.parser.GenericXMLDataParser
014-08-20 00:26:08,703 [http-bio-8080-exec-1] INFO org.n52.wps.io.ParserFactory: Parser class registered: org.n52.wps.io.datahandler.parser.GenericFileParser
014-08-20 00:26:08,703 [http-bio-8080-exec-1] INFO org.n52.wps.common.WPSConfig: Configuration Reloaded, Listeners informed
014-08-20 00:26:08,703 [http-bio-8080-exec-1] INFO org.n52.wps.webadmin.ChangeConfigurationBean: Saved and Activated new configuration!
```

Fig: 6.10: Successful activation of new configuration

The figure (6.11) shows that R repository is configured with 52North WPS.

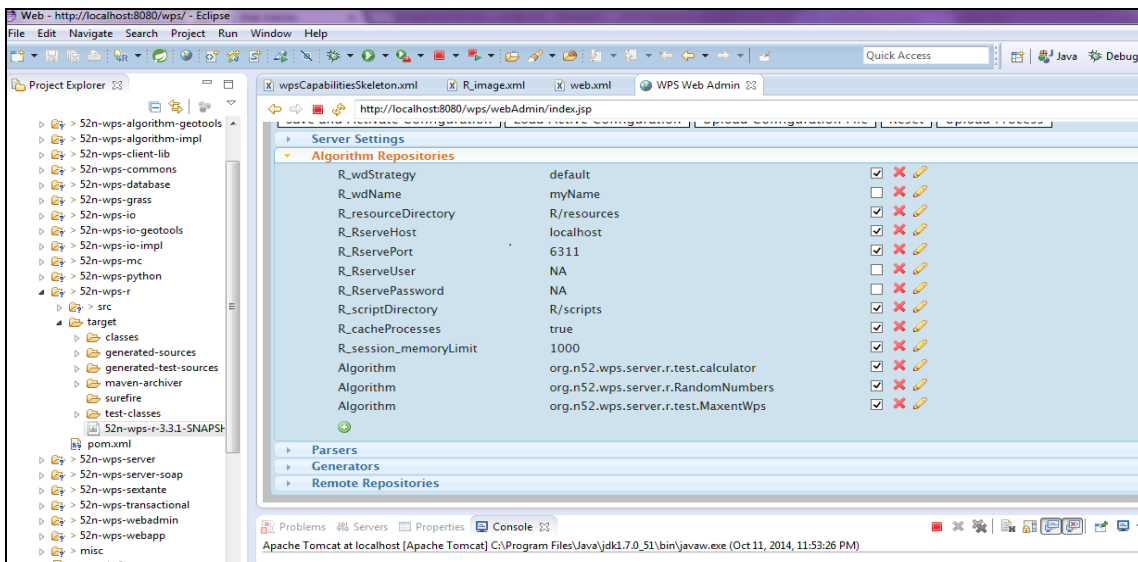


Fig: 6.11: Configuration of R repository into 52⁰North WPS and deploying the MaxEnt process

On successful configuration, the console window shows up, as shown in the figure (6.10), we get a notification that the configuration is successfully saved and activated.

In order to check the processes registered on server, we request for capabilities of using 'GetCapabilities' request. When a GetCapabilities request is sent, all the processes registered would be displayed as shown in the figure (6.12). This allows user to select a process from available processes.

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

```
<wps:ProcessDescriptions xmlns:wps="http://www.opengis.net/wps/1.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ows="http://www.opengis.net/ows/1.1"
xsi:schemaLocation="http://www.opengis.net/wps/1.0.0 http://schemas.opengis.net/wps/1.0.0/wpsDescribeProcess_response.xsd" xml:lang="en-US" service="WPS" version="1.0.0">
  <ProcessDescription statusSupported="true" storeSupported="true" wps:processVersion="1.1.0">
    <ows:Identifier>
      org.n52.wps.server.algorithm.JTSConvexHullAlgorithm
    </ows:Identifier>
    <ows:Title>
      org.n52.wps.server.algorithm.JTSConvexHullAlgorithm
    </ows:Title>
    <DataInputs>
      <Input minOccurs="1" maxOccurs="1">
        <ows:Identifier>data</ows:Identifier>
        <ows:Title>data</ows:Title>
        <ComplexData>
          <Default>
            <Format>
              <MimeType>application/wkt</MimeType>
            </Format>
          </Default>
          <Supported>
            <Format>
              <MimeType>application/wkt</MimeType>
            </Format>
          </Supported>
        </ComplexData>
      </Input>
    </DataInputs>
    <ProcessOutputs>
      <Output>
        <ows:Identifier>result</ows:Identifier>
        <ows:Title>result</ows:Title>
        <ComplexOutput>
          <Default>
            <Format>
              <MimeType>application/wkt</MimeType>
            </Format>
          </Default>
          <Supported>
            <Format>
              <MimeType>application/wkt</MimeType>
            </Format>
          </Supported>
        </ComplexOutput>
      </Output>
    </ProcessOutputs>
  </ProcessDescription>
</wps:ProcessDescriptions>
```

Fig: 6.12: Displaying the capabilities of 52⁰North WPS server through GetCapabilities request

After checking the availability of processes, client can send request through WPS_TestClient, and can select one from the drop down box (fig: 6.13).

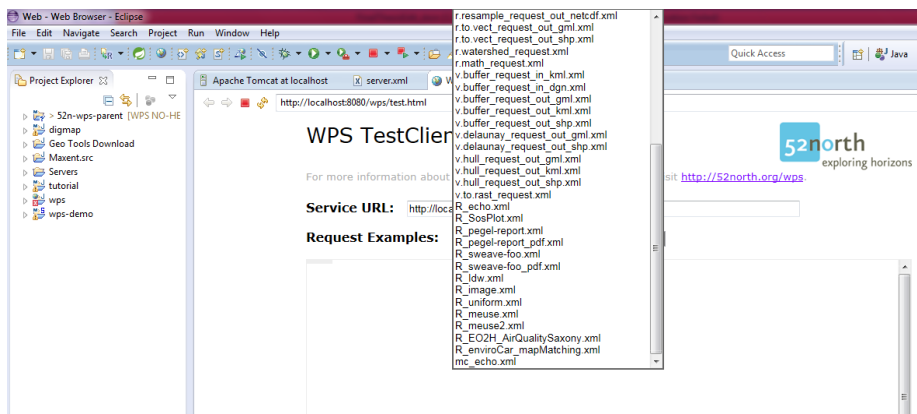


Fig: 6.13: WPS Test Client displaying the list of registered processes, available for users to select

When the process is selected request is sent to describe the process and a notification of its success will be shown.

6.4. R-Script Calling MaxEnt Utility

Since the MaxEnt software is packaged as a jar file, we are utilizing the MaxEnt software functionality by calling it into R.

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

```
#MAXENT
# checking if the jar file is present. If not, skip this bit
jar <- paste(system.file(package="dismo"), "/java/maxent.jar", sep='')
if (file.exists(jar))
{
  xm <- maxent(pred, pres_train, factors='l15')
  plot(xm)
} else {
  cat('cannot run this example because maxent is not available')
  plot(1)
}

# RESPONSE PLOT
if (file.exists(jar)) {
  response(xm)
} else {
  cat('cannot run this example because maxent is not available')
  plot(1)
}
if (file.exists(jar)) {
  e <- evaluate(pres_test, backg_test, xm, pred)
  e
  px = predict(pred, xm, progress='')
  par(mfrow=c(1,2))
  plot(px, main='Maxent, raw values')
  #plot(wrld_simpl, add=TRUE, border='dark grey')
  tr <- threshold(e, 'spec_sens')
  plot(px > tr, main='presence/absence')
  #plot(wrld_simpl, add=TRUE, border='dark grey')
  points(pres_train, pch='+')
} else {
  plot(1)
}
```

Modifying the script with respect to 52⁰North WPS

Scripts are a typical way for analyzing data in many different scientific domains, a generic and seamless solution to deploy scripts on web services opens up the possibility to share and re-use scripts. These scripts are encapsulated with few comment lines, the goal of these comments is to make coding for a specific server runtime environment obsolete and to reduce the workload for deploying scripts to uploading a single file the annotated script[5]. The first annotation type provides a general *process description* and occurs once in a script. It is used as followings:

```
# des: id = sdm, title= "MaxEnt", # abstract = "analyzing species distribution modeling";
```

The mandatory *id*-field sets the identifier of the script and must be used by the web service interface to identify the process provided by the script. The fields *title* and *abstract* are self-explaining and provide a brief documentary description of the process. The annotations *in* and *out* can occur multiple times in a script. The former is used to declare the inputs that should be loaded into the script environment before the script is executed; the latter is used to declare outputs that the execution environment should retrieve from the script workspace after the script has been executed and provide to the requesting web client. Apart from the descriptive and optional fields *abstract* and *title*, they contain the mandatory fields *id* and *type*. *Id* in this case specifies the name of the variable within the script's workspace that contains the input or output value. In case the input or output is a file, the corresponding variable contains the name of the file. The *type* argument declares the data type. Its value is resembles either a primitive data type like an integer, double, Boolean/character string, or complex data types and formats, like object and file. The *input* annotation also contains an optional field *value* to declare literal default values. In order to state how many times an input or output must occur we use optional fields *minOccurs* and *maxOccurs*. Dividing into training and testing sets as shown on the script on the right hand side in figure(6.14) that is displaying the environmental layers in ascii format in R.

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables



Fig. 6.14: Graphical representation of Environmental variables contributing to the growth of *Hippophae Rhamnoides* (on left) & a snippet of script providing result image

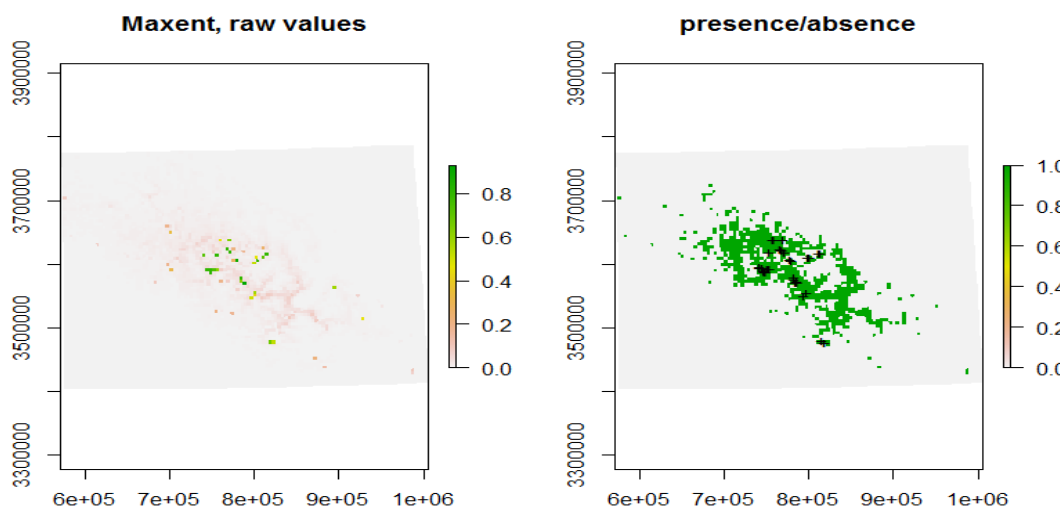


Fig: 6.15: Prediction of distribution of *Hippophae Rhamnoides* using Maximum Entropy modelling technique

Figure 6.15 shows the final prediction of *Hippophae Rhamnoides* distribution using Maximum Entropy modelling. The area covered is the study area and is showing the maximum occurrences on the Spiti river basin, since the area around it is the wettest among other regions and as shows in the figure below, the wettest quarter shows the maximum contribution for growth of the species. Hence this species is also called as cold desert crop.

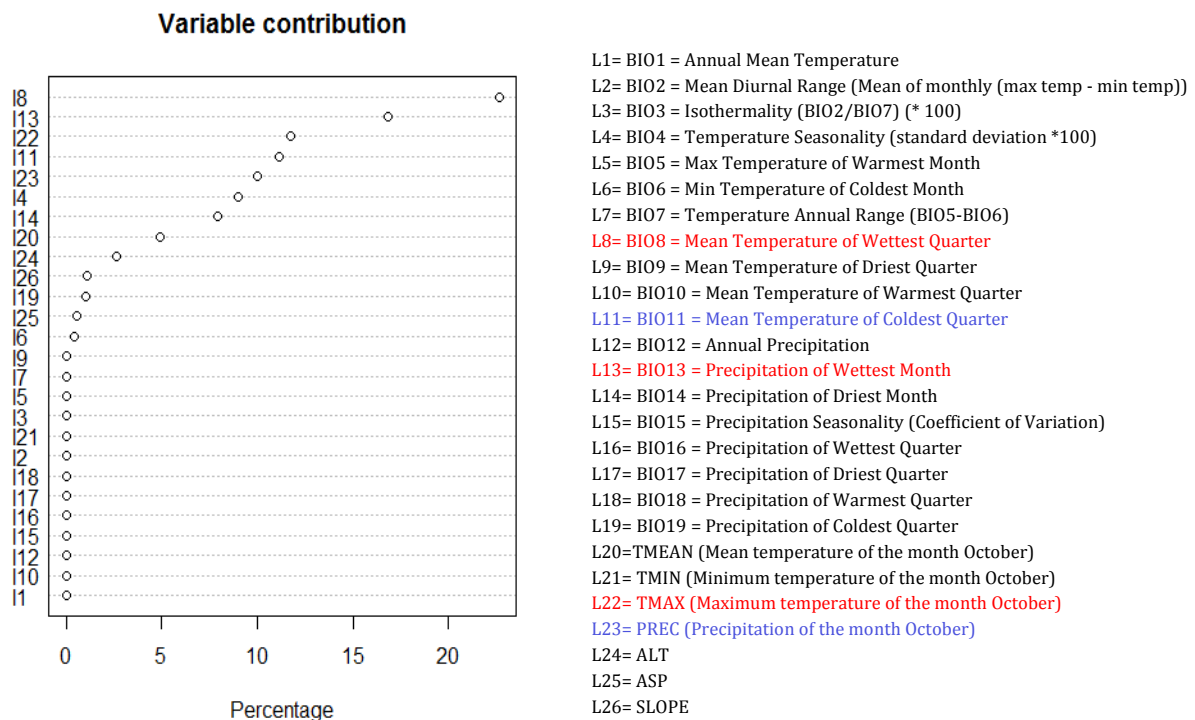


Fig: 6.16: Individual variable contribution graph and each Variable's denotation

We have taken all bioclim variables and maximum, minimum and mean temperatures as well as precipitations data of the month October. The variables slope and aspect were derived from the variable altitude. The figure (6.16) shows that the maximum contributing variable is 18 for the growth of *Hippophae Rhamnoides* which signifies, the species niche distribution is more suitable when the temperature and precipitation are extreme wet to be precise during the month of October. The next most contributing variable apart from wettest quarter of temperature and wettest month of precipitation is mean temperature of the coldest quarter. This explains that the species tends to grow and is more suitable when the temperature is coldest.

6.5: Testing the distribution using MaxEnt software

The data was later tested with the MaxEnt species distribution modelling software so as to evaluate the results performed by R. the similar results were obtained as shown in figure (6.17).

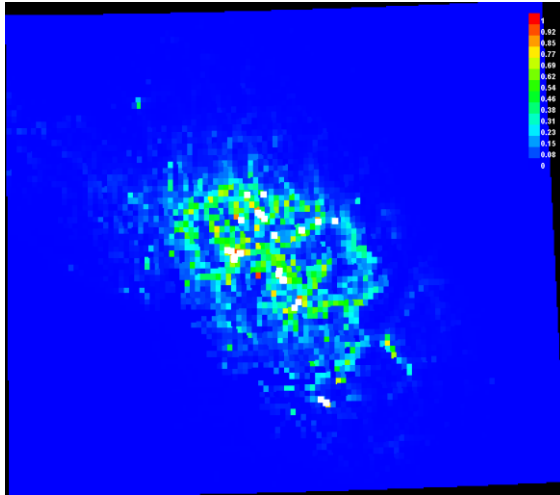


Fig: 6.17: (a) Output produced by MaxEnt software



(b) Species points used in R was collected from this area for modelling and this resembles the spread of the points shown by software

Fig 6.17(a) explains that warmer colors show areas with better predicted conditions. White dots show the presence locations used for training, while violet dots show test locations. The points were collected with reference to the area shown in 6.17(b) and thus, the points spread in 6.17(a) resemble the area in (b).

7. CONCLUSIONS AND RECOMMENDATIONS

The primary objective of the research was to build an Open Geospatial Consortium's Web Processing Service for Species Niche modelling using environmental variables, and analyse the results envisaged by the MaxEnt species distribution modelling technique. A distribution model of the species *Hippophae Rhamnoides.L* was performed using MaxEnt modelling technique and produced satisfactory results by predicting the presence of species using the environmental variables. So as to make this available as a web processing service, it was integrated to 52⁰North Web Processing Service with an extended R-module, where the MaxEnt utility was called using R. Though the textual detail has been kept minimal but still each step has been explained well using sufficient screen shots. The script calls the MaxEnt utility into R, performs a distribution model and exposes the script on web using 52⁰ North WPS. This involves various platforms, connection of different softwares on a common platform. Many modelling techniques exist for species distribution modelling but this provides a web interface to perform the MaxEnt modelling. The research work undertaken had answered all the research questions.

7.1. Research Questions

1. How to integrate the chosen model within WPS?

The Maximum Entropy modelling technique was chosen for performing species distribution model for the species *Hippophae Rhamnoides*. We called the MaxEnt utility into R through a script which performed similar results as that of the original MaxEnt software. This was integrated to 52⁰North WPS by activating the R- repository over that of a default one using Rserve which acts as a socket server (TCP/IP socket) and allows binary requests to be sent to R. Since 52⁰North WPS is basically a Java based open source, with an option to use R through an extended module. Through which clients working on Java can use the facilities of R without the need of linking to R code entirely.

2. What services of the modelling technique are generated as result?

The MaxEnt model utilized through R performs similar results like that of the original software, provided few advanced results may not be available. We have focused on the basic functionality of MaxEnt model, and our script results each variable's individual contribution along with response curves which is similar to that of jackknife test of the original software. Apart from this, a distribution map is generated predicting the presence of species based on the Maximum Entropy principle which shows the maximum probability of occurrence of a species per pixel along with its response to environmental variables. This is akin with the results performed by the software when evaluated and compared.

3. How effective is the OGC WPS service for the implementation of Species Distribution Modelling?

Since the species data deals with sensitive data, to avoid any sort of privacy issues, we designed such that the data is statically given as an input by wrapping up within the script, whereas there are other possibilities too, either the data can be dynamically defined or the data can be stored in GeoServer and can be accessed through web coverage service. But when the data is stored on cloud, proper measures should be taken to assure privacy since the species data is assumed to be confidential to make it accessible to all. So when the data is given through the script, the data can be safe, thus making OGC WPS an effective option for the implementation of species distribution modelling.

7.2. Recommendations

Few recommendations to improve and extend the research on this domain are; as MaxEnt software is called into R using a specific package and the results perform just as the original MaxEnt modelling software which serves a simple distribution prediction map along with the variable contribution and response curves of each variable. Advanced options provided by the original MaxEnt software like L-regularization is not available since we have focussed on basic requirements as of now. So in the future, maybe few more functions similar to that of the advanced section in the original MaxEnt software can be integrated to make good analysis of the resultant prediction.

Another improvement can be change in the script with dynamic inputs. Since the current model takes input that is declared statically and sent through the script, this can be upgraded to dynamic inputs. In addition to this, we have installed Rserve on Windows for the current research, so installing Rserve on Linux or Debian would give better results as per the 52North documentation, hence that can also be tried and the results can be compared with respect to speed.

The research can be extended by connecting 52North WPS to GeoServer and view the processes on GeoServer. And the data can even be stored in GeoServer using web map services and web coverage services and can be programmed to access the data within the GeoServer itself, process it and display the result.

We can even call R script from python using 'RPy' which acts as an interface between python and R; Python can be used instead of Java, but when using GeoServer with python backend it has to be activated and when 52⁰North WPS is used, if the code is developed in python by calling R through Python interface, then "Python process repository" should be activated to the process registration if WPS is accessed through 52⁰North WPS. This is another method recommended.

Apart from these, there is another approach; involving an open source desktop tool Quantum GIS, can be used for calling online web processing services through Sextante toolbox which connects various geo-processing scripts and processes of open source softwares like GeoServer and 52⁰North WPS. QGIS can easily connect to 52North WPS or GeoServer's online services, activating the Sextante repository.

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

Another recommendation would be, if the MaxEnt source code is available, or is made open source, then the entire development can be done on GeoServer instead of these interconnections.

REFERENCES:

- [1] J. Elith and J. R. Leathwick, "Species Distribution Models: Ecological Explanation and Prediction Across Space and Time," *Annu. Rev. Ecol. Evol. Syst.*, vol. 40, no. 1, pp. 677–697, Dec. 2009.
- [2] R. G. Pearson, "Species' distribution modeling for conservation educators and practitioners," *Lessons Conserv*, vol. 3, pp. 54–89, 2010.
- [3] J. T. Rotenberry, K. L. Preston, and S. T. Knick, "GIS-BASED NICHE MODELING FOR MAPPING SPECIES'HABITAT," *Ecology*, vol. 87, no. 6, pp. 1458–1464, 2006.
- [4] S. J. Phillips, M. Dudík, and R. E. Schapire, "A maximum entropy approach to species distribution modeling," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 83.
- [5] M. Hinz, D. Nüst, B. Proß, and E. Pebesma, "Spatial Statistics on the Geospatial Web."
- [6] K. Evangelidis, K. Ntouros, S. Makridis, and C. Papatheodorou, "Geospatial services in the Cloud," *Comput. Geosci.*, vol. 63, pp. 116–122, Feb. 2014.
- [7] A. M. V. Monteiro and G. Câmara, "WBCMS—A SERVICE ORIENTED WEB ARCHITECTURE FOR ENHANCING COLLABORATION IN BIODIVERSITY: THE CASE OF SPECIES DISTRIBUTION MODELLING COMMUNITY," *Month*, vol. 20, p. 200X, 2009.
- [8] "Species Distribution Modelling - Microsoft Research." [Online]. Available: <http://research.microsoft.com/en-us/projects/sppdistmod/>. [Accessed: 27-May-2013].
- [9] J. Elith and J. Franklin, "Species Distribution Modeling," in *Encyclopedia of Biodiversity (Second Edition)*, Editor-in-Chief: Simon A. Levin, Ed. Waltham: Academic Press, 2013, pp. 692–705.
- [10] R. Jones, D. Cornford, and L. Bastin, "UncertWeb processing service: making models easier to access on the web," *Trans. GIS*, vol. 16, no. 6, pp. 921–939, 2012.
- [11] A. M. Castronova, J. L. Goodall, and M. M. Elag, "Models as web services using the Open Geospatial Consortium (OGC) Web Processing Service (WPS) standard," *Environ. Model. Softw.*, vol. 41, pp. 72–83, Mar. 2013.
- [12] G. Dubois, M. Schulz, J. Skøien, L. Bastin, and S. Peedell, "eHabitat, a multi-purpose Web Processing Service for ecological modeling," *Environ. Model. Softw.*, vol. 41, pp. 123–133, Mar. 2013.
- [13] S. Nativi, P. Mazzetti, and G. N. Geller, "Environmental model access and interoperability: The GEO Model Web initiative," *Environ. Model. Softw.*, vol. 39, pp. 214–228, Jan. 2013.
- [14] "Standards - W3C." [Online]. Available: <http://www.w3.org/standards/>. [Accessed: 07-Dec-2014].

- [15] "Web Processing Service | OGC." [Online]. Available: <http://www.opengeospatial.org/standards/wps>. [Accessed: 07-Dec-2014].
- [16] "Species distribution modelling: Data analysis in ecology | School of Mathematics and Statistics." [Online]. Available: <https://www.maths.unsw.edu.au/about/species-distribution-modelling-data-analysis-ecology>. [Accessed: 22-Sep-2014].
- [17] M. Aguilar and C. Lado, "Ecological niche models reveal the importance of climate variability for the biogeography of protosteloid amoebae," *ISME J.*, vol. 6, no. 8, pp. 1506–1514, 2012.
- [18] S. J. Sinclair, M. D. White, and G. R. Newell, "How useful are species distribution models for managing biodiversity under future climates," *Ecol. Soc.*, vol. 15, no. 8, 2010.
- [19] C. Merow, M. J. Smith, and J. A. Silander, "A practical guide to MaxEnt for modeling species' distributions: what it does, and why inputs and settings matter," *Ecography*, vol. 36, no. 10, pp. 1058–1069, Oct. 2013.
- [20] D. L. Warren and S. N. Seifert, "Ecological niche modeling in Maxent: the importance of model complexity and the performance of model selection criteria," *Ecol. Appl.*, vol. 21, no. 2, pp. 335–342, 2011.
- [21] J. Elith, S. J. Phillips, T. Hastie, M. Dudík, Y. E. Chee, and C. J. Yates, "A statistical explanation of MaxEnt for ecologists: Statistical explanation of MaxEnt," *Divers. Distrib.*, vol. 17, no. 1, pp. 43–57, Jan. 2011.
- [22] S. J. Phillips, R. P. Anderson, and R. E. Schapire, "Maximum entropy modeling of species geographic distributions," *Ecol. Model.*, vol. 190, no. 3–4, pp. 231–259, Jan. 2006.
- [23] N. Young, L. Carter, and P. Evangelista, "A MaxEnt Model v3. 3.3 e Tutorial (ArcGIS v10)," *Nat. Resour. Ecol. Lab. Colo. State Univ. Natl. Inst. Invasive Species Sci. Colo. USA*, 2011.
- [24] M. ARMINJON and D. IMBAULT, "1 The principle of maximum statistical entropy (MAXENT)," *Z. Für Angew. Math. Mech.*, vol. 80, no. 1, pp. 13–16, 2000.
- [25] S. Phillips, "A brief tutorial on Maxent," *ATT Res.*, 2005.
- [26] A. Sayar, M. Pierce, and G. Fox, "OGC compatible geographical information systems web services," *Indiana Comput. Sci. Rep. TR610*, 2005.
- [27] "Open Geospatial Consortium | OGC." [Online]. Available: <http://www.opengeospatial.org/>. [Accessed: 07-Dec-2014].
- [28] "Web Coverage Service | OGC." [Online]. Available: <http://www.opengeospatial.org/standards/wcs>. [Accessed: 07-Dec-2014].
- [29] "Web Coverage Processing Service (WCPS) Standard | OGC." [Online]. Available: <http://www.opengeospatial.org/standards/wcps>. [Accessed: 07-Dec-2014].

- [30] "The OGC's Role in Geosciences and the Environment | OGC." [Online]. Available: http://www.opengeospatial.org/domain/geosciences_and_environment. [Accessed: 07-Dec-2014].
- [31] "GeoServer User Manual — GeoServer 2.6.x User Manual." [Online]. Available: <http://docs.geoserver.org/stable/en/user/>. [Accessed: 07-Dec-2014].
- [32] "Implementing Geospatial Web Services: A Resource Webliography." [Online]. Available: <http://www.istl.org/10-spring/internet2.html>. [Accessed: 08-Oct-2014].
- [33] "Introduction — ZOO Project 1.4 documentation." [Online]. Available: <http://www.zoo-project.org/docs/services/introduction.html>. [Accessed: 07-Dec-2014].
- [34] B. Anupkumar, T. S. Rao, and K. K. Satpathy, "Mapping of Hippophae rhamnoides Linn. in the adjoining areas of Kaza in Lahul and Spiti using remote sensing and GIS," *Curr. Sci.*, vol. 80, no. 9, p. 1107, 2001.
- [35] A. Ranjith, "Phytochemical investigations on sea buckthorn (hippophae rhamnoides) berries," 2011.
- [36] "Species Distribution Modeling | California Climate Commons." [Online]. Available: <http://climate.calcommons.org/article/species-distribution-modeling>. [Accessed: 09-Oct-2014].
- [37] T. Mitchell, "Our New Publication," *OSGeo J.*, vol. 1, no. 1, 2007.
- [38] "The R Project for Statistical Computing." [Online]. Available: <http://www.r-project.org/>. [Accessed: 07-Dec-2014].
- [39] "GeoServer Developer Manual — GeoServer 2.5.x Developer Manual." [Online]. Available: <http://docs.geoserver.org/2.5.x/en/developer/>. [Accessed: 04-Nov-2014].
- [40] "geoserver/geoserver · GitHub." [Online]. Available: <https://github.com/geoserver/geoserver>. [Accessed: 04-Nov-2014].
- [41] "Tools — GeoServer 2.5.x Developer Manual." [Online]. Available: <http://docs.geoserver.org/2.5.x/en/developer/tools.html#maven>. [Accessed: 04-Nov-2014].
- [42] "Java SE Development Kit 7 - Downloads | Oracle Technology Network | Oracle." [Online]. Available: <http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>. [Accessed: 10-Oct-2014].
- [43] "Apache Tomcat 7 (7.0.56) - Tomcat Setup." [Online]. Available: <http://tomcat.apache.org/tomcat-7.0-doc/setup.html>. [Accessed: 10-Oct-2014].
- [44] "Eclipse Guide — GeoServer 2.6.x Developer Manual." [Online]. Available: <http://docs.geoserver.org/stable/en/developer/eclipse-guide/index.html>. [Accessed: 10-Oct-2014].
- [45] "Maven – Download Apache Maven." [Online]. Available: <http://maven.apache.org/download.cgi>. [Accessed: 09-Nov-2014].

- [46] "Implementing a WPS Process — GeoServer 2.5.x Developer Manual." [Online]. Available: <http://docs.geoserver.org/2.5.x/en/developer/programming-guide/wps-services/implementing.html>. [Accessed: 04-Nov-2014].

USER GUIDE

This project deals with modeling the distribution of species using a web processing service. Using this, user can model the distribution of species, provided the data required for modeling (in our case; since we use MaxEnt modelling technique only presence occurrence points are sufficient along with the background data/ climatic data).

Pre-requisites:

User is expected to have the following softwares configured/installed in the system.

- Apache tomcat
- 52North WPS
- R
- Eclipse IDE
- An idea of Species distribution modelling in R

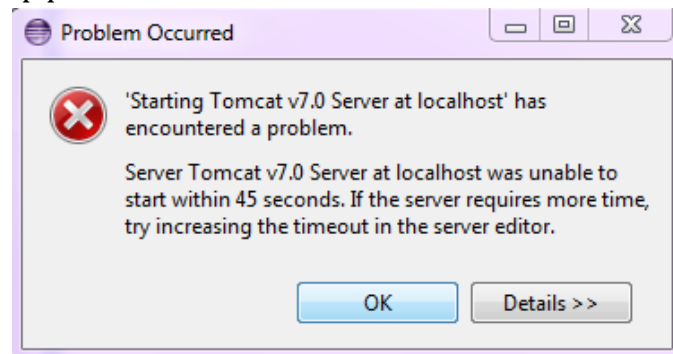
Note: It is suggested to install and configure all the softwares on a common platform; say Eclipse

Steps:

1. Assuming 52North WPS is installed in default mode, make sure the R-repository is activated (i.e. one of an extension module of WPS, WPS4R)
2. Now, R is connected to 52⁰North WPS using Rserve, so Rserve should be installed and activated.
3. Now, after Rserve is installed, restart the 52⁰North WPS and this connects, R and 52⁰North WPS.
4. A script which calls MaxEnt utility should be supplemented by species data and climatic data as well.
5. Now, check if the script is performing distribution model when it is run.
6. Then, the script should be little modified with respect to 52⁰North WPS terms, that is the input variables and output variables should be declared with the fields 'in' and 'out' respectively, and the script should be given a unique identification in the field 'id' and this should be the name of the script it is saved as and other optional fields about the description of the process can also be added as per your choice.
7. This script should be registered to 52⁰North WPS and deployed.
8. Restart the server now and if the script is registered, you will be able to view it in your capabilities list.
9. Using WPS_Client, the process can be selected and request is sent and executed.
10. Now, the script performs the distribution model of the species input passed through the R-script.

EXPECTED ERRORS WHILE INSTALLATION OF SOFTWARES AND HOW TO SOLVE THEM:

1. Server startup problem



This is the error occurred during integrating Apache Tomcat with Eclipse.

In order to solve this, the startup time should be increased, since projects like GeoServer and 52⁰North doesn't startup in 45seconds.

2. Memory pool of the Apache Tomcat server should be increased so as to avoid memory issues while installing softwares like GeoServer and 52⁰North WPS.

OutOfMemory issues:

```
Files\Apache Software Foundation\Tomcat 7.0\wtpwebapps\wps\WEB-INF\classes\users
.xml
Aug 22, 2014 4:12:02 PM org.apache.jasper.compiler.TldLocationsCache tldScanJar
INFO: At least one JAR was scanned for TLDs yet contained no TLDs. Enable debug
logging for this logger for a complete list of JARs that were scanned but no TLD
s were found in them. Skipping unneeded JARs during scanning can improve startup
time and JSP compilation time.
Exception in thread "http-apr-9090-exec-6"
Exception: java.lang.OutOfMemoryError thrown from the UncaughtExceptionHandler i
n thread "http-apr-9090-exec-6"
Aug 22, 2014 4:22:03 PM org.apache.catalina.core.StandardContext reload
INFO: Reloading Context with name [/wps] has started
Aug 22, 2014 4:22:44 PM org.apache.catalina.core.StandardWrapper unload
INFO: Waiting for 1 instance(s) to be deallocated for Servlet [jsp]
Aug 22, 2014 4:23:20 PM org.apache.catalina.core.StandardWrapper unload
INFO: Waiting for 1 instance(s) to be deallocated for Servlet [jsp]
Aug 22, 2014 4:23:54 PM org.apache.catalina.core.StandardWrapper unload
INFO: Waiting for 1 instance(s) to be deallocated for Servlet [jsp]
2014-08-22 16:24:41.328 [ContainerBackgroundProcessor[StandardEngine[Catalina]]
INFO org.n52.security.service.config.support.SecurityConfigContextListener: Se
curityConfiguration successfull removed from ServletContext
Exception in thread "http-apr-9090-exec-3"
Exception: java.lang.OutOfMemoryError thrown from the UncaughtExceptionHandler i
n thread "http-apr-9090-exec-3"
Aug 22, 2014 4:30:23 PM org.apache.catalina.core.ContainerBase$ContainerBackgrou
ndProcessor run
SEVERE: Unexpected death of background thread ContainerBackgroundProcessor[Stand
ardEngine[Catalina]]
java.lang.OutOfMemoryError: PermGen space
```

3. When building the configuration, if a system is being accessed through a proxy server, then to make Maven work properly, proxy address should be given in the configuration file so that the archetypes are downloaded through proxy.

4. Context of the open source software should be registered with the Apache Tomcat, though it works without registering the context, ignoring it might result in improper functioning of software.

5. Multiple softwares can run on same port numbers, unless and until memory is an issue. Though GeoServer has an embedded server, both GeoServer and 52North can work on same port number as well making us easily access processes in 52North via GeoServer.

Development of Open Geospatial Consortium's Web Processing Service for species niche modelling using environmental variables

6. In order to use MaxEnt software on system Java heap memory should be increased to avoid data size issues.
7. It is safe to store the data in a dismo package's folder and address it through that instead of local system's destination.
8. Make sure that the environmental data used is in ASCII format only.
9. While embedding the Apache Tomcat with Eclipse, make sure Eclipse isn't running externally. It might be conflicting if it is working outside eclipse and apache tomcat wouldn't start within eclipse.
10. Any project after forking GitHub will not provide web archive file, so in order to run your open source on a local system as a developer end, web archive should be self-generated.
11. Make sure Rserve is working and the connection is established in order to run the process on the web.
12. Improper installation or configuration of 52North WPS might not provide the access to the WebAdminConsole and the following error will be produced.

HTTP Status 500 - javax.xml.parsers.FactoryConfigurationException: Provider org.apache.xerces.jaxp.DocumentBuilderFactoryImpl not found

type Exception report

message javax.xml.parsers.FactoryConfigurationException: Provider org.apache.xerces.jaxp.DocumentBuilderFactoryImpl not found

description The server encountered an internal error that prevented it from fulfilling this request.

exception

```
javax.servlet.ServletException: javax.xml.parsers.FactoryConfigurationException: Provider org.apache.xerces.jaxp.DocumentBuilderFactoryImpl not found
org.apache.jasper.servlet.JspServlet.service(JspServlet.java:348)
javax.servlet.http.HttpServlet.service(HttpServlet.java:727)
org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
org.n52.security.service.authentication.servlet.AuthenticationChainFilter.doFilter(AuthenticationChainFilter.java:80)
org.n52.security.service.config.support.AbstractSecurityConfigServletFilter.doFilter(AbstractSecurityConfigServletFilter.java:95)
org.n52.security.service.config.support.SecurityConfigDelegatingServletFilter.doFilter(SecurityConfigDelegatingServletFilter.java:62)
org.n52.security.service.config.support.AbstractSecurityConfigServletFilter.doFilter(AbstractSecurityConfigServletFilter.java:95)
com.thetransactioncompany.cors.CORSFilter.doFilter(CORSFilter.java:169)
com.thetransactioncompany.cors.CORSFilter.doFilter(CORSFilter.java:232)
```

root cause

```
javax.xml.parsers.FactoryConfigurationException: Provider org.apache.xerces.jaxp.DocumentBuilderFactoryImpl not found
org.apache.jasper.xmlparser.ParserUtils.parseXMLDocument(ParserUtils.java:115)
org.apache.jasper.compiler.JspConfig.processWebBotXml(JspConfig.java:95)
org.apache.jasper.compiler.JspConfig.init(JspConfig.java:243)
org.apache.jasper.compiler.JspConfig.findJspProperty(JspConfig.java:302)
org.apache.jasper.compiler.Compiler.generateJava(Compiler.java:115)
org.apache.jasper.compiler.Compiler.compile(Compiler.java:374)
org.apache.jasper.compiler.Compiler.compile(Compiler.java:354)
org.apache.jasper.compiler.Compiler.compile(Compiler.java:341)
org.apache.jasper.JspCompilationContext.compile(JspCompilationContext.java:657)
org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:357)
org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:395)
org.apache.jasper.servlet.JspServlet.service(JspServlet.java:339)
javax.servlet.http.HttpServlet.service(HttpServlet.java:727)
org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
org.n52.security.service.authentication.servlet.AuthenticationChainFilter.doFilter(AuthenticationChainFilter.java:80)
org.n52.security.service.config.support.AbstractSecurityConfigServletFilter.doFilter(AbstractSecurityConfigServletFilter.java:95)
org.n52.security.service.config.support.SecurityConfigDelegatingServletFilter.doFilter(SecurityConfigDelegatingServletFilter.java:62)
org.n52.security.service.config.support.AbstractSecurityConfigServletFilter.doFilter(AbstractSecurityConfigServletFilter.java:95)
com.thetransactioncompany.cors.CORSFilter.doFilter(CORSFilter.java:169)
com.thetransactioncompany.cors.CORSFilter.doFilter(CORSFilter.java:232)
```

root cause

```
java.lang.ClassNotFoundException: org/apache/xerces/jaxp/DocumentBuilderFactoryImpl
java.lang.Class.forName0(Native Method)
java.lang.Class.forName(Class.java:270)
javax.xml.parsers.FactoryFinder.getProviderClass(FactoryFinder.java:123)
javax.xml.parsers.FactoryFinder.newInstance(FactoryFinder.java:178)
javax.xml.parsers.FactoryFinder.newInstance(FactoryFinder.java:147)
javax.xml.parsers.FactoryFinder.find(FactoryFinder.java:219)
javax.xml.parsers.DocumentBuilderFactory.newInstance(DocumentBuilderFactory.java:121)
org.apache.jasper.xmlparser.ParserUtils.parseXMLDocument(ParserUtils.java:115)
```