DecSpace Framework: Improvements to the Catalog of Methods

Francisco Todo Bom
Instituto Superior Técnico
franciscobom@gmail.com

ABSTRACT

Multi-criteria decision analysis (MCDA) is a domain of knowledge that explicitly evaluates multiple criteria in decision making problems. It is an area of growing interest with a wide-array of uses in various fields. The work for this project was done on the DecSpace platform. DecSpace is a web-based client application that features a number of different implemented MCDA methods and a user-friendly interface for non-specialists. It was developed in a project prior to this one and improved through the work done here.

One of the more important tools related to MCDA and this project is the diviz framework, which is a software tool for constructing complex MCDA workflows through a visual interface. The diviz platform also has many methods available for anyone to use through their webservices. The main new functionality developed in this project was the integration of DecSpace with the diviz webservices in an effort to increase the number of MCDA methods available through DecSpace.

Additionally, a new local method was added, and in order to improve performance and security the database solution was changed. Finally, some small changes to the interface were made to improve usability and user-experience. A user-evaluation was done on the platform's usability and it managed to achieve the expected usability scores. As a result of this evaluation, some small improvements were made to the platform's interface.

KEYWORDS

Multi-criteria, decision, software, interface, application

1 INTRODUCTION

MCDA is a domain of knowledge for evaluating problems with multiple and sometimes conflicting criteria that serves to aid in the decision making process. It is a growing area of knowledge with a vast applicability in many different fields and large number of different methods.

Some of these methods can be complicated to use even for professionals that are not familiar with the field, thus the need for software solutions often called Decision Support Systems (DSS). These are then used to support the Decision Maker (DM) during the decision process. These systems aim to facilitate the use of MCDA methods and provide different MCDA methods that can then be applied to the user's specific problem.

The solution proposed by DecSpace is a unified online framework that MCDA methods can be developed for, under a single interface, and standardized input methods. By providing a single interface it can become trivial to apply various MCDA problems to the same problem. The framework also aims to be user-friendly enough so that it is not restricted to advanced users. The user should not need to have knowledge in MCDA methods to use the framework.

With the DecSpace prototype having been developed in another project, the objective of this thesis became to extend the platform in several aspects. The main functionality added to the platform was the interoperability with the diviz web-services, which considerably extended the number of available methods on the platform. Additionally, the extensibility of the platform was tested by implementing a new local method, the database solution was changed and several other small usability changes were made to the platform.

2 MCDA OVERVIEW

MCDA is a branch of Operational Research that uses mathematical techniques to aid in decision problems. These methods can be applied to various domains of knowledge such as healthcare, environmental management, risk management, governance, finance, among several others.

The decision aiding process usually involves two main actors. The first is the decision maker who is the one the aid must be provided. The second is the analyst, who is in charge of facilitating the modeling of the decision problem. This process is done with the decision maker in order to include his personal preferences and objectives for the given problem.

On any given day people face a several different decision problems. However, there have been four main types of decision problems identified:

- **The choice problem:** Where the goal is to select the single best option or reduce the options to a subset comprised of "good" options.
- **The sorting problem:** Where the goal is to sort the various options into pre-defined ordered groups called categories. The aim of these problems is to aggregate options that share similar characteristics for descriptive, organizational or predictive reasons.
- **The ranking problem:** Where the aim is to sort the various options form best to worst by scoring them or comparing them to each other in pairs. These rankings can vary with the different criteria.
- **The description problem:** Where the goal is to describe the options and their consequences. The objective being that describing the options is the first step to understanding them more fully to solve the decision problem.

Choosing the appropriate MCDA method can often be an arduous task. None of the methods are perfect nor can they be applied to all the problems. The three major types of MCDA methods are described below:

- **Full aggregation approach:** Each criterion is given a score and these are then combined into a single global score.
- **Outranking approach:** In this approach a bad score cannot be compensated by a good score. It also accepts the notion of incomparability between options, which results in a partial order of options.
- **Goal, aspiration or reference level approach:** With this approach a goal is defined for each criterion. The option that comes closest to completing all the selected goals is then chosen.

The decision aiding activity is based on three fundamental pillars :

- **Actions:** Formal definitions of the possible actions or alternatives for a given problem.
- **Consequences:** Aspects, attributes or characteristics of the actions that allow them to be comparable to each other.
- **Modeling of a preference system:** This consists of an implicit or explicit process, that for each pair of actions envisioned, assigns one of the following possibilities: Indifference, preference, or incomparability.

Based on these pillars the analyst should attempt to obtain a coherent structured set of results to guide the decision process and follow an approach that aims to produce knowledge from a certain hypotheses. This approach should be based on models discussed with the decision maker.

3 RELATED WORK

In this section we will briefly describe some of the current MCDA software solutions related to this project. These tools were chosen based on features that would be interesting to implement in the DecSpace project and their relevance in the area.

3.1 OrclassWeb

The ORCLASS method is focused in classifying multi-criteria alternatives, which are the options for solving the problem, and can be characterized according to the criteria provided. The method differs from other verbal decision analysis methods because it does not aim at ordering alternatives but at classifying all possible alternatives, which are given by the Cartesian product of the criteria values defined for the problem. The method aims at categorizing the alternatives into a small number of decision classes or groups, which are pre-ordered according to the decision maker's preferences.

The OrclassWeb Tool is a web-based program built in a web environment in platform Java 1.6 using JSF 2 and runs in server Tomcat 6. It aims to provide an automated way to perform the ORCLASS comparison process of alternatives. OrclassWeb was developed divided into four stages:

1. Criteria and criteria values definition
2. Alternatives definition
3. Construction of the classification rule
4. Presentation of results obtained

The ORCLASS system increases its complexity exponentially with the number of criteria and criteria values for each. With the OrclassWeb tool, the user can apply ORCLASS with a greater number of criteria and criteria values.

When using the OrclassWeb tool, the user will go through three stages. At the beginning of the application the user will need to define the criteria and criteria values associated that the alternatives will be compared against. This stage is called "Problem Formulation". The second stage, called "Construction of the Classification Rule", involves defining the alternatives and characterizing them according to the criteria values. The next step is the elicitation of preferences stage where the decision maker will answer some questions in order to obtain the scale of preferences. Given these answer, OrclassWeb will divide the alternatives into the classes/groups. As the final step, OrclassWeb presents an interface composed by the groups and the related alternatives preferred for each one in order to allow a complete and more detailed analysis of the results.

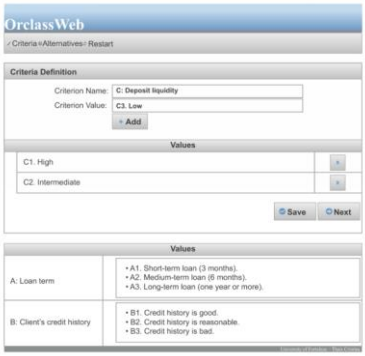

Figure 3.1: OrclassWeb Criteria Definition Interface

3.2 Decerns

Decerns is a framework for MCDA or spatial and non-spatial alternatives that uses scalable decision support systems. There are two application described in this project, DecernsSDSS and DecernsMCDA.

The main objective of this project was to to develop models and computer tools for aiding in decision making for land-use planning problems. This platform contains different MCDA methods and DSSs.

DecernsSDSS is a web application built according to Java EE 6 specifications with a three-tier architecture: presentation, application processing, and data management.

DecernsMCDA is a scalable DSS for analysis of multi-criteria problems. The main differentiator between DecernsMCDA and other similar systems is the inclusion of several popular MADM

methods in a single framework which is useful for uncertainty treatment.

It has been shown to be an effective tool for decision support regarding environmental problems, remediation of contaminated sites and risk based land-use planning.

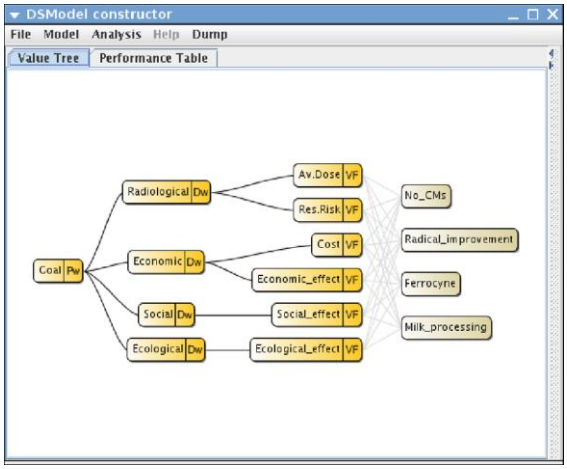

Figure 3.2: Decerns Model Construction

3.3 MCDA-ULaval

MCDA-ULaval is a free single-layered desktop application tool programmed in Java that implements multiple criteria decision analysis algorithms developed by the University Laval. It is a stand-alone desktop application with no connection to other systems or services. Supported decision methods are Electre II, III, Tri-B, Tri-C, Tri-rC and Tri-nC.

The system is based on the concept of "project". Each project has a set of actions and criteria from which subsets are derived. It can also contain many performance tables and decision configurations. It also features the option to choose between cardinal and ordinal levels of measurement for each criterion, as well as maximization or minimization of criteria. Also included is sensitivity analysis of a decision's parameter giving the lower and upper bond of the interval more stability. Performance tables are imported and exported via CSV files and there is also the option to export as CSV the concordance, discordance, credibility and outranking matrices of a result.

In terms of visuals output, the performance tables an also be displayed through spider charts and rankings in Electre II and III can also be represented in graphs. Other general output such as results and rankings can be displayed in graphs as well, e.g. pie charts.

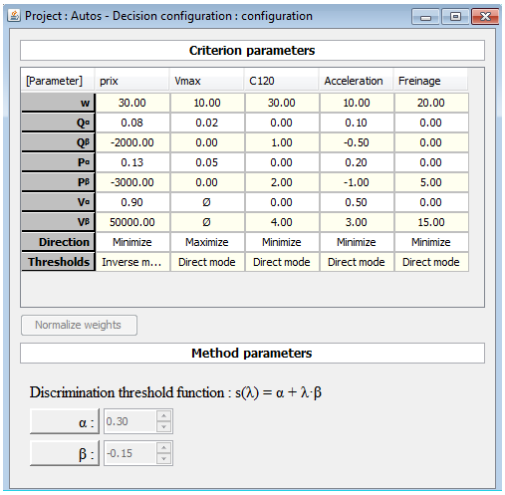

Figure 3.3: MCDA-ULaval Criterion Parameters

3.4 Diviz

This The DecSpace project borrows heavily from the diviz framework and as such diviz will be described in detail in this section. The diviz software is a tool for modeling, processing, and sharing MCDA techniques. The target audience are users experienced in MCDA methods, ranging from researchers, teachers, and students. The main features of diviz are its modular nature and ease of comparison of results produced by different methods for the same problem.

The diviz tool is a classical 3-tier application made up of the client, a component that accesses the XMCDA web-services (described in detail later) and a server. The server side concerns itself with executing the submitted workflows, whereas the client side is where the workflows are designed by the user. The web-services are used to communicate between these two sides.

Diviz is an open-source software tool made to design complex workflows of MCDA algorithms in an intuitive manner, execute said methods, and share them. The diviz tool also makes use of two other outcomes of the Decision Deck Project:

- XMCDA: A standardized CML recommendation used to represent objects and data structures from the field of MCDA. It's used to allow different methods to interact with each other by sharing the same standard.
- XMCDA web-services: distributed open-source computational MCDA resources.

The diviz tool's purpose it to build, execute, and share complex MCDA algorithms in the form of workflows. It allows the construction of these workflows, also referred to as methods, by combining various modular calculation components. This methodology provides several advantages:

- Removes the black box effect of certain methods.
- Provides a better understanding of the methods and their similarities or lack thereof.

- Allows the user to easily test variations of methods by replacing certain components.

Succinctly, it allows users to better understand, interact, and alter know MCDA methods.

MCDA workflows are designed through an intuitive graphical interface. Different algorithms are represented by boxes which can be linked together or with other supplementary elements or data files through connectors. As such, diviz does not require any sort of programming knowledge, although it does require a basic understanding of the various modules and how to combine them effectively. The construction of workflows is done by dragging and dropping modules from a list of available calculation elements into the workplace. The relevant data files must also be included in the workflow and connected accordingly.

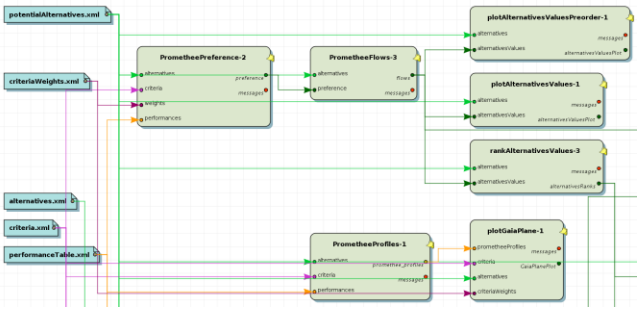

Figure 3.4: Diviz Workflow Example

MCDA methods are often constructed by combining a sequence of modular algorithms, as explained earlier, although this is not always clear. The XMCDA web-services provide the functionality for these elementary modules. This also works to eliminate the repetition of certain operations. If many methods can be created from a few elementary algorithms, then the same code for these algorithms can be reused for the various methods.

3.4 Conclusions

This section analyzed some of the software solution to MCDA problems available today. These platforms were all different to each other and each provided their own set of advantages and disadvantages in relation to each other.

MCDA-Ulaval has an interesting project management aspect which is useful for comparing results and has specific tools to help introduce data into the program. However, it's an offline application and has a strict interface that doesn't allow multiple methods to be run off each other's results.

Decerns has the advantage of having a wide array of visualization techniques and also provides many MCDA methods. It suffers by not being able to compare results from different projects directly, as well as not having a data importation feature.

The diviz software boasts many qualities such as its flexibility, usability and the inter-interoperability of its methods thanks to the way workflows are designed. These qualities are something Decspace aims to replicate and were the motivation behind the workspace.

4 PROBLEM ANALYSIS AND REQUIREMENTS

This section aims to explain what tasks were considered important to the platform and the reasoning behind those decisions. The first section will describe the DecSpace platform in detail in the state it was before development started on this thesis. From there, the requirements for improving the platform were lifted.

4.1 DecSpace Requirements and Use Cases

DecSpace is a web-based software solution that aims to provide a user-friendly interface for using various MCDA methods for non-specialists. It is a prototype developed in another project, which in turn was an improvement over another MCDA framework prototype. The work done on this prototype will serve as a foundation for the development done during this project. This chapter will describe the original DecSpace platform in detail.

The main requirement for DecSPace was that it needed to be a web-based platform. This aspect made the application platform independent and more easily accessible by users, since all you need is a device with a modern web browser and an Internet connection. There is no need to install the application and work can be done on separate devices since all your data will be stored in the database.

Another advantage of a web-based solution is more simplicity on the development side. The application becomes more scalable since different aspects of the program can be separated by server and client side. Development complexity should also be kept to a minimum, which is the reason behind the choices made for the technology used, explained in detail in section 5.

DecSpace should also be open-source, to allow other developers and researchers to eventually extend the platform through plugins.

DecSpace should provide MCDA methods with simple to use interfaces and be easily expandable with minimum coding.

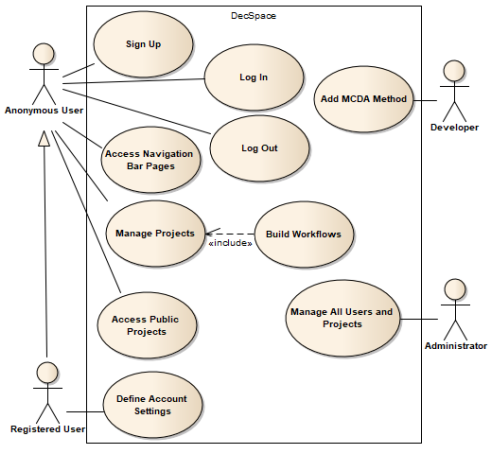

Figure 4.1: DecSpace Use Cases

DecSpace shares its basic architecture with other MCDA tools such as diviz, Decerns and the original MCDA framework it was originally based on. DecSpace is designed in a classic three-tier architecture comprised of a data tier, an application tier and a client tier. This three tiered solution has the advantage of compartmentalizing certain aspects of the software, such as keeping the method implementations separate from the interface and data sources. The three tiers of DecSpace are as follows:

- Client tier - This consists of the user interface that interacts with the system. It is through this tier that a user can send requests to the application tier. It is mostly made up of the HTML pages that serve as an interface and some client-side javascript code that controls the pages' behavior and is therefore located entirely in the web-browser.
- Application tier - This is where most of the computational work is done. It communicates with the users through the client tier and connects with the data tier and any other external services when needed. It is deployed in the application server.
- Data tier - This is where the information is stored. This tier communicates with the application tier when it needs to answer data requests.

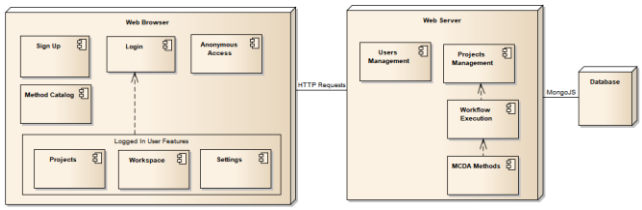

Figure 4.2: DecSpace Architecture

In terms of technology, the DecSpace solution is built on the MEAN stack approach. This stack defines itself as an "opinionated full stack Javascript framework" that simplifies and accelerates the development of web applications. The stack is comprised of the following tools:

- MongoDB - MongoDB is a free open-source cross-platform database document oriented database program.
- Node-js - Node.js is an open-source, cross-platform JavaScript run-time environment for executing JavaScript code server-side design to build scalable network applications.
- Express - Express is a web application framework for Node.js and is considered the standard server framework for Node.js
- AngularJS - AngularJS is the application framework for the front-end development of the application
-

Since Decspace is a web application, it also uses HTML for formatting and designing the structure of the web pages that make up the application. CSS is also used for visually styling the pages. In addition, JQuery is used to navigate the document and selecting the various elements to transform them client-side. To help design the web-pages, BootStrap, an open-source framework for building reactive web-pages, was also used.

The data used in the application, including user-data and data relating to the MCDA methods is stored in an online MongoDB service called mLab.

There are a few concepts that should be explained along with their relationships in order to better understand DecSpace.

A user is identified in the database by its email address (which must be unique) and a password, which are used to log in. They also have a name and privacy setting associated with their account. Other details such as the date they first registered and the date of the last login are also stored. A user can have multiple projects, defined by a unique identifier.

Projects also contain information such as their name, user, date of creation and date of the last update. A single project can have several workflows archived by the user and has a unique identifier. Projects are associated to the user that created them and can be either private or public. Private projects are only accessible to the original creator whereas public projects are accessible by anyone.

A workflow is composed of various modules such as input files, method modules, and output modules. These modules can be added to the DecSpace workspace and connected to each other to form executable workflows. There are different types of method modules, one for each available method on the platform.

These concepts are illustrated by figure 4.3.

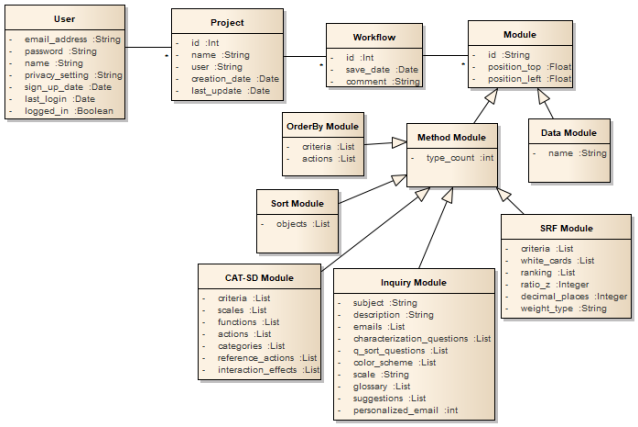

Figure 4.3: DecSpace Domain Model

4.2 Problem Requirements

The main objective of the work done throughout this thesis on the DecSpace platform was to expand the platform's usability and functionalities. The code base from DecSpace was mostly kept and changes were only implemented when necessary. The objective was to add new functionalities and have them seamlessly integrate into the existing framework.

Most of the changes made were mostly back-end with only a few minor changes made to the user-interface and a few bug-fixes. The main objectives can be summarized as:

- Add a new local method: This was done to test whether the platform was extensible when it comes to the local methods implemented.
- Import XML type files: he previous version only accepted CSV and JSON file types so this new functionality had to be added in order to have input files that follow the XMCDA standard.
- Implement a method calculated remotely: A method where all the calculations are done in a remote server was implemented to text the extensibility of the platform to use remote methods.
- Import the diviz method catalog: Once the remote method had been tested, a new functionality to import the diviz catalog in real time was implemented. The user can now browse all available methods through the diviz webservices.
- Implement the generic diviz method: Through this method the user can pick any of the remote methods available through the diviz webservices and call on them through DecSpace.
- Change the database to a local solution: With the growth of the platform it became important to have more control over the database. DecSpace used an online database service which was good enough for the prototype but having a local database increases the speed, security, and control over the data.
- Interface and usability changes: Various small changes were made to the user interface to deal with the new functionalities as well as some slight improvements suggested by users of the platform.

4.3 Conclusions

This section started by explaining the architecture, technology and the concepts behind the DecSpace platform. When developing the new functionalities, the code base was reused and an attempt was made to make them blend into the existing framework with minimal changes.

The main actors in the DecSpace platform remain the same: anonymous user, registered user, developer and administrator. Although the database solution was changed, the platform maintains the three-tier architecture of client, application server, and database.

This version served as the base for the development of this project and as such had a great influence over the technologies that were used and which functionalities were considered to be lacking. By analyzing this framework in detail it was decided which aspects needed improving and how the new functionalities could be implemented as seamlessly as possible into the existing framework without major changes.

From here, the requirements for improving the solution were lifted and described.

5. SOLUTION PROPOSAL AND DEVELOPMENT

The development for this thesis followed an "Evolutionary Prototyping" method where new functionalities were added iteratively.

Since the main changes to the platform were done on the server side, the changes in technology are limited to new NodeJS libraries. The libraries used were the following:

- request: Used to simplify http calls and used to make SOAP requests.
- xml2js: Simple XML to Javascript object converter used to parse downloaded XML files.
- html-entities: Used to encode and decode the SOAP envelopes as required.

5.1 New Local Method

The additive aggregation method was the first method implemented in DecSpace for this project for its simplicity and relevance.

This method takes in two sets of data, one which defines the criteria for evaluation and their respective weight, and another with the various options and their defined scores for each criterion. Both of these sets must be defined in standardized scales for the method to be performed correctly although the method does not impose any limitations on the scale used; that is left to the user. The data can be entered manually by the user through a modal that allows this or, alternatively, input files with the relevant data in json or CSV format can be provided. The method is run through a service like all other DecSpace implemented methods and returns a table with the results, ordered from best to worst. The data required for the method to run can be introduced either manually through the method module modal or through input files in the CSV or JSON formats.

This method was added to test the extensibility of the DecSpace platform when it comes to implementing new methods that are processed locally.

5.2 Retrieving Available Diviz Methods

In order to have up-to-date versions of what methods are currently available through the diviz servers, a new functionality for retrieving them was developed for Decspace. It works by automatically retrieving XML description files from the Diviz website which are available publicly and processing these files to retrieve the relevant information. These description files have a common internet address and a unique numerical identifier and are therefore straightforward to cycle through until all method descriptions have been downloaded. Once downloaded, Decspace will process the data, convert the XML descriptions into Javascript objects and store them as JSON files in the Decspace database. This way, it becomes much faster to access and read this data whenever the user needs it without the need to connect to the Diviz servers each time we need to access

This functionality was added in preparation for the Generic Diviz method, which is described later in this section.

5.2 Calling Remote Methods

One of the main features developed for the Decspace platform for this thesis was the ability to call remote methods from the diviz servers which are made available through public web-services. Adding this functionality was considered crucial because it would add a great number of methods to the platform, making it more versatile and robust as a result.

The communications with the Diviz server are all done through standard SOAP requests, where all the relevant information is contained within the envelope itself. There are no input or output files being transfered, the input files for any given method are opened and written into the SOAP envelope when requesting a method and the same is done for the results, where all the relevant data is contained within the response envelope. The solution was then to write an algorithm that builds custom SOAP envelopes with all the relevant data and then sends them to the appropriate web-service address. This is done in NodeJS and the data can come from either an input XML file or from data from within the application. With this solution we remove the need for the flask framework, the file-system, and the need to rely on the provided python script, allowing for a finer control of how the methods are called and what is being sent.

5.2.1 RankAlternativesValues Method

The first remote method implemented is called the "Rank Alternative Values" method that exists on the diviz servers. It's a simple method used for demonstration purposes that simply orders a set of "alternatives" based on their "overall values". It essentially constructs an ordered list based on the inputs.

This particular method works with raw XML data that follows the XMCDA format, provided by the user, and constructs a properly formatted SOAP envelope with the request to the Diviz server. The user can alter the data, but to do so they must work directly on the raw XML text, which can be confusing and not user-friendly. The recommended way to use this method is to import the data directly from a file.

This method was programmed as a first prototype to test the connection between the DecSpace platform and the diviz servers and as such has limited functionality and interactivity.

5.2.2 Generic Diviz Method

This new method allows the user to call any method currently available and listed in the Diviz webservices website. It can be used through the workspace like all other methods but has a few key differences. Since this method does not represent one single method and instead can be used to call upon a large number of them it has a dynamic interface modal for user input. When the input modal is open, the user can search for the method they want to use through the search bar and the relevant necessary inputs will appear once the user has selected a method. The user may then fill the form with the relevant data by hand but since the files necessary to run diviz methods have to follow the XMCDA standard a better approach would be import the files directly, which can be done through the import files button.

Once the user has input all the relevant data, they can run the method through the workspace like they would any other. The communication with the diviz server is then done through the NodeJS server and the output is produced in the output box in the workspace. The output contains a message from the server which informs the user if the method was run successfully, as well as the answer body. If the input data is incorrectly formed or if the server is busy the message will inform the user. The answer body will contain the relevant data, which varies by method, and also follow the XMCDA data format so it might not be easy to read for novice users. However, since the use of this generic method requires the user to have a basic understanding of the XMCDA format to submit the input data, it is assumed they will be able to interpret the results

5.3 Changing the Database Solution

The previous version of Decspace used the website mLab.com as a database service for all functionalities including user login data, project data, and data related to the Inquiry method. This service functions as a cloud-based MongoDB database.

This solution has the problem of relying on an external resource outside our control which comes with a few disadvantages. The main problem is relying on the current state of this external service. If the service is busy or offline then the application will not function. The second problem was the difficulty in having multiple instances of the database attributed to different developers. The solution was tied to a single account in mLab.com and a single database, meaning that every local distribution of Decspace was accessing the same database. A third problem was that the execution time for various functionalities was negatively influenced by having to wait on this service, sometimes for several seconds for even the most basic functionalities. Finally, having a remote solution also meant less control over the files and presented some security concerns although we won't address them for now.

The solution to this problem was to create a local mongoDB database in one of the servers provided by IST. To so so, the 3.6 version of MongoDB was installed on the Linux server. The new database has authentication control enabled to protect it from anyone other than the DecSpace developers from accessing it. The connection to the server is done automatically by the DecSpace platform and it's possible to define which database the build will access.

5.4 Interface Improvements

Various small changes were made to the Decspace interface in order to improve the user experience or add some new non-critical functionalities to the platform.

As the number of available methods grew it became apparent that the current solution for selecting them (using a simple drop-down menu) would be insufficient. As such, the drop-down menu was replaced by a button that brings up the method selection modal, a new component that displays all available methods in a dynamic table and split into two groups: the local methods and the remote methods.

In order to better separate the different types of modules from each other in the workspace, it was decided to change their overall appearance based on their type. The input files are now a light blue whereas the output files are a light green color. The method modules remained the same gray color.

In order to avoid having the users fill out the method input data incorrectly, when inputting them manually through the modals, some safeguards were implemented to make sure the user knows that there are mistakes in the input data.

Although some measures were taken to notify users when the method inputs are incorrect, some methods also condition the user to make it impossible to enter invalid data manually.

5.5 Remote Method Example: ELECTRE

The remote methods are all available through the Generic Diviz method. The method calculations are done in a remote server and the answer is sent back as an XML file, viewable in the DecSpace platform. Example input files for all available diviz methods are available to download as an effort to make the use of these methods simpler, since users will have to adhere to the XMCDA data format which is not always intuitive to write.

It should also be noted that many of the services available through diviz are not full MCDA methods but instead modular algorithms that perform specific tasks. By linking these modules together the user can create a full MCDA method as seen in figure 6.1. This was done to make the extensibility of the platform easier since simple modules can be used in various methods.

There are two main parts to an ELECTRE application: first, the construction of one or several outranking relations, which aims at comparing in a comprehensive way each pair of actions; second, an exploitation procedure that elaborates on the recommendations obtained in the first phase. The nature of the recommendation depends on the problem being addressed: choosing, ranking or sorting.

The Electre Methods are usually used to discard unacceptable alternatives to the problem after which another MCDA to select the best one is used. The Advantage of using the Electre Methods before is that it saves time by eliminating unacceptable alternatives.

Through the “Generic Diviz” module it becomes possible to recreate these complex worflows through the diviz webservices.

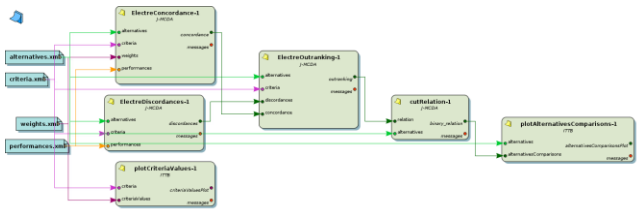

Figure 6.1 ELECTRE workflow in Diviz

5.6 Conclusions

In this chapter, the main changes to the platform are described in detail.

The design philosophy behind all this project's solution development was to extend the platform with minimal changes to the user experience and interface. The technological changes were limited to the database solution and a few NodeJS libraries that were imported to the project as needed.

The main features developed for the platform were presented as well as the process involved with coding them. The main focus of the thesis, the ability to call remote methods from the diviz server, expanded the platform's catalog of available methods considerably but managed to integrate with the local methods. These remote methods are called by a user the same way one would call a local method. This followed the design principle of the platform of being as user-friendly as possible, although some knowledge of data formats is needed to use the "Generic Diviz" method.

A few of the minor changes to the interface aimed at improving the user-experience are explained with a few examples given.

In the final section an example method workflow was shown.

6. EVALUATION PROCESS

The role of evaluation is to assess the system design and test the system to ensure that it behaves as expected and meets user requirements.

6.1 Evaluation Process

The system evaluation was carried out with the objective of identifying possible issues with the system's usability, functionality, and user-experience. The evaluation process follows an "experimental evaluation" and was conducted on 20 participants. The following sub-sections describe the various aspects of the evaluation in detail.

Prior to the tests, the coordinator sent the user manual and test guide to the participants. They were asked to look through the manual before beginning the tasks to get a chance to understand what Decspace is and how it works.

The participants chosen to take part in this evaluation were all considered potential users of the platform that had no prior experience with Decspace. Although none of them had experience with MCDA software, they did have some knowledge of MCDA concepts and software systems. The tasks proposed involved using some simple methods and did not require any specific knowledge of either computer systems or MCDA. A total of 20 users took part in the test, 7 of which were knowledgeable in the field of MCDA.

The author of this dissertation and developer of Decspace (Francisco Todo Bom) acted as the coordinator and observer during these test sessions.

At any time during the test sessions, the participant could consult the user manual if they were having difficulty completing a certain task and in case that proved insufficient, ask the coordinator directly for some assistance. In case either of these

measures proved necessary, the coordinator/observer recorded what caused these difficulties.

During the test sessions, the following metrics were recorded and analyzed:

- Time to complete each Task.
- Successful task completion.
- User errors per task.
- Help per task.
- Usability and User Experience.
- Recommendations.

6.2 Evaluation Results

The objectives for what would be considered a successful evaluation were created based on the expert opinion of the system designer and programmer, as well as the results from the previous evaluation of DecSpace.

The user-evaluation process described above was conducted on 20 participants. The average results for each of the metrics can be seen in table 6.1.

	Successful Completion	Time to Complete (minutes)	Non-Critical Errors	Critical Errors	Help (User Manual)	Help (Coordinator)
Task 1	1	0:57	0.15	0	0	0
Task 2	1	1:15	0.05	0	0	0
Task 3	1	2:14	0.05	0	0	0
Task 4	1	4:05	1.37	0.05	0.05	0.10
Task 5	1	3:15	1.95	0.05	0.10	0.10
Task 6	1	3:38	1.47	0.55	0.11	0.45

Table 6.1: Evaluation Results

Throughout the user-evaluations and through the survey some changes were suggested on how to improve the platform's usability. The coordinator also took note of the most common mistakes and how to change the interface to minimize them. The following improvements to the interface were developed after the user-evaluation phase:

- Option to delete all modules from workspace
- Change initial position of modules to avoid overlap
- Add a representative symbol to the search bar
- Add a placeholder title in the "Generic Diviz" method catalog
- Improved buttons to upload files in "Generic Diviz" module
- Erase content from "Generic Diviz" modal
- Placeholder text for empty field in catalog

6.3 Conclusions

The User-Evaluation can be considered a success since it fulfilled the objective of clearing over 80\% of the defined objectives. The main purpose of the evaluation was to test how intuitive the interface was to people inexperienced with the platform. Every task had the participants test a unique functionality. Therefore all of them required different interactions where lessons learned from previous tasks would not improve performance.

The test was conducted on both MCDA experts and novices with no significant differences in the results, which proves the concept that DecSpace is a user-friendly platform for non-experts. Although the experts had more familiarity with the concepts at play, the interface proved to be equally simple to both groups.

The constructive feedback given by users throughout the evaluation process and through the open ended survey questions also helped further improve the platform by identifying the major usability problems. These improvements were then developed for the final version of DecSpace, improving the overall usability.

7. CONCLUSIONS AND FUTURE WORK

DecSpace was originally created as an answer to the expanding use of MCDA methods in various fields of study. It's main objective was to become a more inclusive platform to allow non-experts the use of these methods.

The design and technology used in the platform was chosen according to a few key features that were deemed necessary: online availability, ease of use, as well as simplicity and extensibility of the platform by developers. As such, the MEAN stack was chosen as the main technological base for the platform, a choice made in the early versions of DecSpace. Since the work on this thesis aimed to build on the previous version of DecSpace, the technological solution remained consistent.

In order to decide which features should be added to the platform in this project, a study was done on the current state of the art platforms for executing MCDA methods. Although they all possessed their own strengths and weaknesses, none of them provided the ease of use and availability DecSpace aims to provide. They also suffered from having completely different interfaces and data formats from each other, whereas DecSpace aims to be a unified platform where multiple methods can be developed for and used.

During the development of this thesis the main objectives of the platform remained consistent with those of the previous version and as such the platform was extended more so than changed. The new features aimed to blend into the existing interface as an effort to keep the platform as simple as possible for users.

The development done on DecSpace during this thesis proved its extensibility, which was a core requirement. A new local method was added, as well as access to the entire library of remote methods created by the diviz team. The list of remote methods can now be updated in real time and consulted through the method catalog. The database system was moved from a cloud service to a local infrastructure in order to reduce dependencies and improve availability, security and performance. Finally, some changes to the interface were done in order to improve usability.

The user evaluation was successful and proved that the new features for DecSpace were usable by non-experts of MCDA. The participants were able to quickly learn the main functionalities of the platform. Some recommendations on improving the platform were also suggested by users and developed for the final version.

It can be concluded that the framework was successfully enhanced and its extensibility was demonstrated by the inclusion of new features.

The work done during this thesis was focused on improving the platform through the application back-end. This leaves a lot of

work to be done on the user-experience side which can still be greatly improved.

In terms of functionality, there are a few key aspects that would greatly improve the platform.

Although extensible, adding new local methods to the platform is currently a complicated procedure that only developers familiar with the platform will be able to do. This greatly reduces the potential of the platform since simplifying this process would allow outside users to develop their own plugins and extend it. The way local methods work in DecSpace should be reworked to be more modular and generic to allow for this extensibility.

Having access to the diviz web-services greatly increased the number of methods for the platform, but using and interpreting the results form these methods is still a complicated task due to the nature of the XMCDA format. In order to be improved, DecSpace should be able to interpret all XMCDA data formats, which are available to consult in the diviz website. By having them in DecSpace's own database, the methods would become a lot more usable.

The data produced as a result from all methods in DecSpace is still limited to text, which is not optimal in MCDA methods. As such, having a way of displaying results through more graphical outputs such as graphs would be interesting to the decisions makers in order to better analyze the results.

Lastly, DecSpace would benefit from having a system that guides users to the appropriate MCDA method for their specific problem, such as a recommendation system. It's already been proven users can use a specific method if they follow instructions but being to solve their own real-world problems through the platform would be an interesting inclusion. Less experience users could follow a step by step guide that had them insert their data and the platform would then choose an appropriate method and present them with the results.

REFERENCES

[1] Alessio Ishizaka, and Philippe Nemery.
Multi-Criteria Decision Analysis - Methods And Software.
Wiley, United Kingdom, August 2003.

[2] Bernard Roy.
Multicriteria Methodology for Decision Aiding.
Kluwer Academic Publishers, Dordrecht, 1996.

[3] Patrick Meyer and Sébastien Bigaret.
Diviz: A software for modeling, processing and sharing algorithmic workflows in MCDA.
Intelligent Decision Technologies, 2012.

[4] Denis Bouyssou, Thierry Marchant, Marc Pirlot, Alexis Tsoukiàs and Philippe Vincke.
Evaluation and Decision Models with Multiple Criteria - Stepping stones for the analyst.
Springer Science + Bussiness Media Inc., New York, 2006

[5] Ana Sara Costa, Ricardo Vieira, Cristina Verdasca, and José Rui Figueira.
The Applicability of Multi-Criteria Decision Aiding Methods to Risk Management.
Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal

[6] André Barbosa
DecSpace: A Multi-Criteria Decision Analysis Framework.
Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal

[7] Boris Yatsalo, Vladimir Didenko, Sergey Gritsyuk, and Terry Sullivan.
Decerns: A Framework for Multi-Criteria Decision Analysis.
International Journal of Computational Intelligence Systems, 2015.

[8] Thais Cristina Sampaio Machado, Plácido Rogerio Pinheiro, and Isabelle Tamanini.
OrclassWeb: A Tool Based on the Classification Methodology ORCLASS from Verbal Decision Analysis Framework
Mathematical Problems in Engineering, vol. 2014, Article ID 238168, 11 pages, 2014.

[9] Irène Abi-Zeid, Nicolas Couture-Grenier, William Jones Yankeu Ketchapa, Luc Lamontagne, Mohammed Mouine, and Oscar Nilo
MCDA-ULaval: Multi-Criteria Decision Aiding Tool
http://cersvr1.fsa.ulaval.ca/mcda/?q=en

[10] Alan Dix, Janet Finlay, Gregory D. Abowd, Russel Beale
Human-Computer Interaction
Pearson Prentice Hall, Lancaster University, 2004

[11] Roy, Bernard.
Classement et choix en presence de points de vue multiples (la methode ELECTRE)". La Revue d'Informatique et de Recherche Operationelle (RIRO), (8): 57-75. 1968

[12] G. Bous, P. Fortemps, F. Glineur, M. Pirlot.
ACUTA: A novel method for eliciting additive value functions on the basis of holistic preference statements,
2010.