DecSpace: A Multi-Criteria Decision Analysis Framework

André Dionísio Barbosa
Instituto Superior Técnico
andre.dionisio.barbosa@ist.utl.pt

ABSTRACT

Multi-Criteria Decision Analysis (MCDA) is a field of study that provides a myriad of methods that can be applied to several different domains of use. Using those methods can be a very complex task, which has motivated the development of software solutions to support them. This dissertation has the objective of designing and developing a new web application (called DecSpace) that addresses the usability and user experience issues attached to the use of MCDA methods. This new solution also aims at providing many different MCDA methods and making it easy to add new ones, so that it can support many problems of diverse domains. The most relevant tools, that also support the use of MCDA methods, were studied in order to understand and make use of their greatest aspects and avoid their biggest flaws. A throwaway prototype was built in a previous project as a proof of concepts for the user interaction issues. Its best ideas regarding those issues were perpetuated and adapted to this new solution. After its design and implementation, DecSpace was evaluated in order to comprehend if it was fulfilling its initial objectives. The observed errors and the most useful recommendations made by the users were integrated after the evaluation process ended. DecSpace also went through a validation process, which consisted of the handling and support of two problems that made use of two different MCDA methods. This process had the objective of demonstrating that the new solution can uphold complex issues.

Keywords

Multi-Criteria Decision Analysis (MCDA), web application, usability, user experience, interaction.

1. INTRODUCTION

Multi-Criteria Decision Analysis (MCDA) is an important branch of the Operational Research field, which proposes methods that deal with decision aiding problems involving numerous criteria.

One of the biggest issues of the current MCDA software solutions is that their interfaces expose the complexity of the methods they provide, which drives to some meaningful usability and user experience issues. Therefore, non-expert users are not able to understand or use those interfaces correctly, since they do not have deep knowledge of MCDA. Also, most of those systems provide only one or two methods and use outdated technology.

The lack of an application that does not suffer from any of the mentioned problems led to the development of a new software solution that overcomes them by making use of forefront technology and well-studied usability and user interaction techniques.

Its major objective is to facilitate the use of MCDA methods by having a simplistic design and an easy-to-use interface, while allowing the user to model complex workflows with various methods and datasets. It is web-based, so that it is easily accessible.

2. OVERVIEW OF MCDA

MCDA is a multidisciplinary branch of Operational Research that uses mathematical techniques to deal with decision-making situations developed by human activities. It aims at helping the decision-maker (DM) during the decision aiding process, according to their preferences and judgments, and with respect to the decision situation at hand.

It provides a wide range of approaches and methods to address situations of different kinds and within distinct contexts. It can be applied to a myriad of domains, e.g. healthcare, risk management, governance, environmental management, finance, among many others.

According to [1], MCDA methods can be classified into three different categories:

- **Multiattribute Utility and Value Theories**: the methods of this group address the preferences of the DM over a certain set of actions. Some of the methods it includes are MAUT, UTA, AHP, ANP and MACBETH.

- **Outranking methods**: these methods operate with the objective of building a preference (or outranking) relation among the alternatives evaluated on several attributes. This group includes the ELECTRE methods, PROMETHEE and many others that usually are linked to the general idea of the ELECTRE methods.

- **Non-Classical MCDA Approaches**: this group includes many proposed methods that do not fit in the other two groups due to their nature, such as fuzzy set, decision rule and fuzzy measures and integral approaches, and verbal decision analysis.
3. STATE OF THE ART OF MCDA TOOLS

The development of DecSpace was inspired by four different MCDA methods that presented very useful features and interesting ideas that had the potential to be explored.

3.1 MCDA-Ulaval

MCDA-Ulaval is a free and open-source desktop application that was developed at Université Laval.\(^1\)

The concept of project is fundamental to the understanding of this tool. A project may hold multiple datasets, including the criteria, actions, performance tables and decision configurations.

It only makes available methods from the ELECTRE family (specifically ELECTRE II, III, TRI-B, TRI-C, TRI-rC and TRI-nC). The input and output data of those methods can be visualized in some different graphics and charts, depending on the used method. For example, performance tables can be visualized as spider charts and the rankings of the methods ELECTRE II and III as graphs.

Another main feature of MCDA-Ulaval is the importation and exportation of some components of the methods to the CSV format, e.g. performance tables and output matrices.

MCDA-Ulaval is a standalone desktop program, which means that it does not perform any connections to external platforms (neither servers nor databases). Therefore, it can be assumed that its architecture consists of one tier only.

In terms of technologies, it is written in Java and uses several external libraries for the most diverse endings, e.g. the Java Universal Network/Graph framework is used for representing data as a network or graph and the SuperCSV package is used for converting the data of the methods to the CSV format.

As shown in Figure 1, the interface of MCDA-Ulaval suffers from some usability issues. Even though, the list of configurable aspects of the methods are nicely presented in the left side of the screen, their windows overlap in the right side of the screen, which makes it very difficult to access them at the same time. Ideally, the various components of the methods would be presented in the same screen (without the need of rearranging their windows), since their data may be related.

This tool provides various data insertion techniques, e.g. pre-defined tables with all the necessary parameters, having the possibility to automatically normalize weights and display buttons to easily add, move and delete data rows. These techniques allow the user to insert his data very efficiently and easily configure his projects, similarly to the way it is done in excel tables.

3.2 Decerns

The Decerns framework was created by the DECERNS international project with the intent of developing models and computer tools for supporting MCDA problems, specifically from the domains of risk management and land-use planning [2].

It contains a broad library of modules that are the basis of two distinct systems: DecernsMCDA and DecernsSDSS, which are described in the following subsections. These two systems share the same codebase.

3.2.1 DecernsMCDA

DecernsMCDA is a desktop application for the analysis of multi-criteria problems. Its main objective is to allow an effective implementation of MCDA processes within a multi-criteria decision problem. It also allows the selection of one or more pre-made MCDA models for the analysis of a specific problem.

Its main features are: generic decision aiding tools (e.g. value tree, performance table, value path, scatter plot and domination), MCDA methods (including AHP, TOPSIS, MAVT, PROMETHEE, among many others), sensitivity analysis, weighting and value types.

This tool confines all of its complexity to a single tier and accomplishes that with the use of various Java SE technologies (including the jSimpleGraph and jSimpleViz libraries for the visualization modules). These technologies require that the user has an installed Java Runtime Environment.

The problem structuring and model building in this tool consists of three main phases: value tree, where the criteria and alternative set are defined; performance table, which allows to change criteria attributes and the alternatives values on the criteria; results of the executed method.

The interface for structuring the value tree (see Figure 2) is very simple and clean, which allows the user to have plenty of space to work and have all the necessary tools at his disposal at the same time. The performance table is also very handy, since the user only has to fill in the required values. The results of each method are visualized with different techniques, e.g. the results of the MAVT method are visualized through a bar chart, which can become quite limited for advanced users who wish to analyze their data from different perspectives.

\(^1\)MCDA-Ulaval: http://cersvr1.fsa.ulaval.ca/mcda/?q=en/node/4

Figure 1: The interface of MCDA-Ulaval.
3.2.2 DecernsSDSS

DecernsSDSS (short for Decerns Spatial Decision Support System) is a web-based application specialized in the analysis of multi-criteria spatial problems and it consists of a classical three-layer architecture: presentation layer, which is responsible for the implementation of the user interface and for sending the user requests to the application layer; application layer, that is in charge of performing all the necessary calculations and transferring the data between the two other layers; the data layer keeps all the data safely stored and answers the data requests sent by the application layer.

This tool was developed using multiple Java SE technologies (among them are server pages, server faces and applets) and PostgreSQL 9 for the database. The XML markup language is used as the standard data format.

Figure 3 demonstrates the GIS module of DecernsSDSS. This module provides very powerful visualization techniques that are highly necessary to geographic problems. All the input data and corresponding results are properly displayed without much overlapping or visual confusion.

The major issue of DecernsSDSS is that the various modules do not seem to be fully integrated with each other even though their share data between them. Ideally, the constructed model, its visual representation and all the other used modules should be presented in a single screen without any overlapping, so that the user does not have to cross-check all the given data.

3.3 V.I.S.A

Visual Interactive Sensitivity Analysis (V.I.S.A) is a web-based multi-criteria decision making software, that is also available as a standalone desktop application. One of its main features is that it allows multiple DMs to have a say into the structuring of a single problem by promoting clear communication and organization. This is an unique feature that can be very useful for groups of DMs or analysts that wish to work on the same projects.

It also supports the construction of value trees (to show all the criteria influencing the final decision and their interactions), alternatives and weights (can be set by dragging bars and different weighting systems can be used). These aspects are exemplified by Figure 4.

The structuring of the problems is quite simple in this tool, while the display of various results of that same problem can be useful, yet it can also be visually confusing, since that most of the windows that contain that information overlap each other and are left to the user to rearrange them.

One of the biggest issues of this tool is that it does not allow the use of different MCDA methods and does not even mention the one that it uses.

3.4 Diviz

Diviz is an open-source software tool for building, executing and sharing workflows composed of MCDA methods [6]. It was developed by the Decision Deck Project.

One of the objectives of Diviz is to facilitate the construction of MCDA workflows by combining different elementary components. On one hand, setting up these workflows does

---


3Decision Deck Project: http://www.decision-deck.org/project/
not require any programming skills, since each algorithmic component is well defined by a container and can be easily connected to data files or other components, but on the other hand the user needs to fully understand the functioning of the components that make up the workflow.

A workflow built in this tools is composed of input data files and algorithmic files (as shown by ??). To obtain the output of the various components, it should be executed by the user. All executions are saved and made available to the user. This feature is particularly useful for comparing the data (and corresponding results) of different executions of the same workflow. Any workflow can be exported, with or without its data, and shared with other users. Then, those users can import the workflow to their workspace and continue working from the exact point where the original user left it. Workflows can also be deployed as new algorithmic components and made available to all other users.

In terms of architecture, diviz can be portrayed as a three-tier application comprised of the following tiers: the client (graphical user interface), the registry (accesses the necessary resources and other external platforms) and the server (plans and controls the execution of the workflows, distributes the computations through the dedicated web services and returns the output of the executions to the client).

Regarding the data format, diviz uses the XMCDA standard, which is a data standard that allows the representation of various MCDA concepts and data structures in a clearly defined XML schema. It is an instance of UMCDA-ML (Universal Multi-Criteria Decision Analysis Modelling Language) and was developed with the intent of facilitating the interaction of different MCDA algorithms and their execution on the same problem instance.

This tool is written in Java, which makes it platform independent, but still requires the user to download and install the desktop program (client-side).

Figure 5 illustrates an example of the interface of diviz. The construction of workflows is quite intuitive due to the drag-and-drop mechanism and the color scheme of the data input and output implemented in this tool. The interface can be confusing for users who are not familiarized with it, since it presents too much information at once, more specifically the list of workflows and corresponding executions on the left panel, the current workflow on the middle top side of the screen, the results of the current workflow on the middle bottom panel and the list of all algorithmic components on the right side of the screen. The description and various options of the algorithmic components can be accessed by double-clicking on the module, which is very convenient for getting a quick understanding of the component or setting a certain parameter. It should be possible to input data manually and not have the need of always having an external file with the data correctly structured.

3.5 Conclusions

After the analysis of the various MCDA tools presented above, it can be concluded that, even though they are very useful and each one of them have unique and innovative features, none of them address all the needs of the DMs that regularly use this kind of software. The main characteristics and features of those tools are summarized in Table 1.

On one hand, diviz is the tool that presents the most interesting and unique aspects, such as the simple way of building workflows that can be published as new MCDA methods. On the other hand, it lacks some relevant aspects, e.g. it would benefit from being a web-based application and having a way of adding or changing data manually.

MCDA-Ulaval also introduced itself as a very interesting software program, since it has a very well conceived project management feature that allows an easy comparison of results and provides specific tools that ease the introduction of data. It does not provide a workflow building environment and is not available online.

Decerns (MCDA and SDSS) is valuable for its specific problem-solving modules (e.g. the GIS module) and for providing many MCDA methods. It also provides a wide range of data visualization techniques for analyzing the results of the structured problems. It lacks a data importation and exportation feature, as well as a way of easily comparing results of different projects.

V.I.S.A is web-based and is the only studied tool that presents a feature specifically for allowing multiple users to work on the same project. Even though it is very restrictive (in the sense that it only provides one MCDA method and the problem structuring process is very linear), the myriad of data visualization techniques allow an effective analysis of the results.

4. PROBLEM ANALYSIS AND SOLUTION

The original MCDA framework is seen as a throwaway prototype of the new solution, since most of its usability and user experience ideas were perpetuated by DecSpace. Its re-
Table 1: Summary of the characteristics and features of the studied tools.

<table>
<thead>
<tr>
<th>Feature/Characteristic</th>
<th>MCDA-Ulaval</th>
<th>DecernsMCDA</th>
<th>DecernsSDSS</th>
<th>V.I.S.A</th>
<th>Diviz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web-based</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Interface optimized for different types of devices</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Open-source</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>Many MCDA methods available</td>
<td>×</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>Easy development of new methods</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Explanation and examples of the methods</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>User differentiation</td>
<td>×</td>
<td>×</td>
<td>x</td>
<td>x</td>
<td>×</td>
</tr>
<tr>
<td>Registration and log in</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
</tr>
<tr>
<td>Project management</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Workflow building</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Manual data input</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Data importation and exportation</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Data visualization</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Result comparison</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Multi-User projects</td>
<td>×</td>
<td>×</td>
<td>x</td>
<td>✓</td>
<td>×</td>
</tr>
</tbody>
</table>

requirements, architecture, main features and technology are detailed. Then, DecSpace is analyzed by the same aspects than the original prototype. Additionally, its domain model is presented as well.

4.1 Original Prototype

The original MCDA framework had the intent of being an easy-to-use web-based application that allowed an efficient use and exploration of various MCDA methods. It was developed in a previous project, that can be consulted in [7].

Its main focus is on the usability and user experience issues attached to the use of different MCDA methods. It tries to solve those problems by presenting an auto-explanatory and consistent interface, which allows an intuitive use of the application with little to no learning curve. It also aims at having an easy way for developers to extend the list of available MCDA methods, while maintaining its simplicity and consistency.

Regarding its requirements, anonymous users are able to register and login. Logged users can create, edit and execute projects; view method steps; edit and export method data; view method results; and logout. The administrator can manage user accounts and front pages. The developer can publish new methods; add and remove features; and maintain the framework, server and database.

This throwaway prototype has a classical three-tier architecture composed by:

- **Client tier**: consists of the user interface and interacts with the application tier in order to perform user requests of the various functionalities the interface provides. It implements the "User Authentication", "Projects List", "Project", "Data List Management", "Method Execution" and "Results List Management" components. Infrastructure-wise, this tier is located in the web browser only.

- **Application tier**: most of the logic of this prototype is confined to this tier, since it processes the user requests, performs the connections to the other two layers and external platforms, and carries out most of the necessary calculations. It only consists of the application server.

- **Data tier**: stores and manages the data of the application in an appropriate database. It is also responsible for retrieving that data whenever the application tier requests it.

4.2 Requirements of the Solution

The various requirements of DecSpace (illustrated in Figure 6) can be divided into two distinct groups: characteristics and features. Its characteristics are qualities expected to be built-in the application, while the features are aspects of the solution that the users can make use of.

Regarding the characteristics, DecSpace should be web-based, which makes the application platform independent and, consequently, easily accessible to most users. It also facilitates the its development, since its complexity is confined to the server-side and makes it easily scalable. DecSpace should be open-source, so that other researchers are able to develop add-ons and plug-ins or even continue the implementation of its features. It should provide many MCDA methods and, more importantly, make it easy for the developers to implement new ones without much programming effort. Each of those methods should have a clear explanation of its functioning and an example of the input data and corresponding results.

The most basic features required of DecSpace is user registering and login. The concept of "project management" should be present in the application as well. Each project should be identified by their name, date of creation, date of the last update and privacy setting (public or private), and it should be possible to create, open, duplicate and delete them. The most ambitious feature is the building of workflows. Similarly to the diviz tool, the user should be able to use and connect different modules, so that the results of one method can be used in another one. Another relevant feature is the importation of input data from the CSV and...
JSON formats, as well as the importation and exportation of workflows to the same formats. Public projects can be opened and cloned, but not changed, by any logged user. The administrator should have the right of managing the users and projects of the application. All users should be able to access the most basic information about DecSpace, which includes a brief introduction of the application, the description of the methods, the frequently asked questions and a form for contacting the developers. It should also be possible to create projects and workflows anonymously, i.e. without signing up. After logging out of an anonymous account, the user cannot recover its information.

Figure 6: Use Cases Diagram of DecSpace.

4.3 Architecture and Infrastructure of the Solution

The architecture of DecSpace consists of three main tiers and is illustrated in Figure 7 along with the infrastructure used by the application.

The client tier implements the user interface, which allows the user to accomplish the most various tasks that the system has to offer. Only interacts with the application tier to send user requests and corresponding answers. In terms of infrastructure, it consists of the web browser and communicates to the web server with simple HTTP requests.

The application tier confines most of the system complexity, i.e. where most of the computational work is performed. As mentioned before, it receives and answers user requests sent by the client tier, but also carries out the connections to the data tier. It is located in a web server and executes the workflows and respective MCDA methods.

The data tier safely stores and retrieves all the information of the application. Receives and replies to the data requests sent by the application tier. It is materialized by a simple database and communicates with the application tier with the help of the MongoJS library.

The developed solution is described, including the used technology, the structure of the code, the main pages of the application and the necessary steps for a correct implementation of a new MCDA method in this platform.

Figure 7: Architecture of DecSpace represented by a deployment diagram.

4.4 Domain Model

A user is identified by its email address (which must be unique) and must have an associated password, name and privacy setting. For each user, the date of registering, date of last login and if the user is currently online are also stored. A user may own various projects, which are distinguished by their singular id. Projects are also attributed a name, user (who created the project), date of creation and date of last update (i.e. the last time the project was opened). A single project may contain several workflows, which are singled out by their id as well. They also contain the date they were saved and, possibly, a comment made by the user. A workflow consists of various modules, which can be connected with each other. A module can be of two kinds: method or data (either input or output). Method modules are uniquely identified by the attribute "type_count" and, as of version 1.0, there are five different types of method modules that correspond to the available MCDA methods (CAT-SD, Inquiry, OrderBy, Sort and SRF). The attributes of those methods correspond to their input data. Data modules are simply identified by their name and contain the data that was imported by the user or generated by a method module. These concepts are represented by a class diagram in Figure 8.

Figure 8: The domain model of DecSpace.

5. SOLUTION IMPLEMENTATION

The developed solution is described, including the used technology, the structure of the code, the main pages of the application and the necessary steps for a correct implementation of a new MCDA method in this platform.
5.1 Technology of the Solution

In terms of programming languages, the MEAN stack approach was the main approach for the implementation of DecSpace. It simplifies and accelerates the development of web applications by suggesting a certain combination of programming languages and libraries that cooperate well with each other, which are: MongoDB\(^5\), which is an open-source document database that provides high performance, availability and automatic scaling; Node.js\(^6\), which is an easy-to-use JavaScript runtime environment for building scalable applications; Express\(^7\), which is a minimal and flexible Node.js framework that provides a robust set of features for web and mobile applications; and AngularJS\(^8\), which is a complete JavaScript-based open-source application framework specialized in web front-end development.

5.2 Structure of the Code

The structure of the code of DecSpace is illustrated in Figure 9 and its main components are the "server.js" file, the "node_modules" folder and the "client" folder. The file "server.js" contains all the functions that should be performed by the server. The "node_modules" folder simply contains the scripts of the various NPM packages required by the "server.js" file. The "client" folder contains all the files that are deployed in the client-side of the application. This folder is comprised of the "content" folder, "css" folder, "js" folder and the "index.html" file. The file "index.html" is the homepage of DecSpace and represents the main entry point into the application. The folder "content" contains the all of the HTML pages of the application. The "css" folder contains the necessary CSS files for stylizing the HTML files. The "js" folder comprises the AngularJS modules, controllers and services, as well as other libraries in the "shared" folder.

Figure 9: The structure of the code of DecSpace.

5.3 Main Pages

DecSpace can be divided into three main areas: the homepage, projects page and workspace. Each of these areas fulfil their own purposes through different means. The following subsections describe them in detail.

The homepage makes a general presentation of DecSpace to the user and gives access to the most basic features. The

5MongoDB: https://www.mongodb.com
6Node.js: https://nodejs.org/en/
7Express: https://expressjs.com
8AngularJS: https://angularjs.org/

5.4 How to Add New Methods

The necessary steps to correctly add a new MCDA method to DecSpace are: (1) Add the name of the new method to the list of methods in the file "MethodsService.js"; (2) Create the module of the new method in the file "workspace.html" and stylize it in the "workspace.css" file; (3) Redefine the "createFrom" function located in the "WorkspaceController.js" file; (4) Redefine the "createModuleData" function located in the "WorkspaceController.js" file; (5) Redefine the "reload-Module" function located in the "WorkspaceController.js" file; (6) Create the method service in a new file; (7) Redefine the "executeMethod" function located in the "WorkspaceController.js" file; (8) Define the modal box of the new module in the "workspace.html" file; (9) Create the controller for the new method in a new file; (10) Redefine the data deletion functions, if necessary; (11) Complement the "selectCurrent-Module" function in the file "WorkspaceController.js"; (12) Redefine the output module in the file "workspace.html"; (13) Add the new method to the method catalog.

6. IMPLEMENTED METHODS

As of version 1.0, DecSpace implements a total of five methods. The methods OrderBy and Sort are very simple to understand and were implemented solely to validate the capability of the platform. The other three methods (CAT-SD, Inquiry and SRF) are the most relevant, since they have real-world applications.

6.1 OrderBy

The objective of the OrderBy method is to order a set of actions (also named "alternatives") by their values on the defined criteria set. The criteria is characterized by a name, type (either text or number), direction (ascendant or descendant) and a single criterion should be selected to be the one to order by. The actions are defined by a name and the values on each criterion. Its output is simply the actions ordered by the selected criterion.

6.2 Sort

This method only requires the user to introduce the name of the objects to sort and, then, order them using the drag-and-drop functionality. To order the objects, the user can simply drag a certain object over another and drop it in that position, which will result in the exchange of positions.
between those two objects. The output of this method is simply the same set of ordered objects that was entered by the user.

6.3 CAT-SD
CAT-SD (short for CATegorization by Similarity-Dissimilarity) is a new MCDA method for nominal classification problems \cite{8}. The objective of this method is to assign actions to nominal categories. In order to achieve that objective, the similarity-dissimilarity degrees between the actions and the reference actions should be calculated. An action is assigned to a certain category if it is similar enough to the reference actions representative of that category; i.e. it surpasses the membership degree defined for that category. An extra category is always added for the actions that are not assigned to any of the other categories.

6.4 Inquiry
The Inquiry method is a simplified approach of the Delphi method. The objective of the Delphi technique is to obtain the most reliable consensus of opinion of a group of experts on a certain subject. It tries to achieve that consensus by employing a series of intensive surveys with significant intermissions between them \cite{9}. The Inquiry method differs from the Delphi method, in the sense that Inquiry does not allow consecutive rounds with the same module while being able to incorporate the new data.

Inquiry also makes use of the Q-Sorting technique for collecting the answers of the experts, which consists of three main stages: design of the study (identify the topic, collect the statements, arrange the cells of the block that will be used to answer the survey), administration and data analysis \cite{10}.

Summarizing, the experts are faced with a certain number of statements that they have to attribute a score or classify as "Most Important", "Neutral" or "Least Important". The output consists of the aggregation of the answers and, for each statement its title, total score, average score and standard deviation is presented.

6.5 SRF
SRF is an implementation of the revised Simos’ procedure for the determination of the weights of criteria in the ELECTRE family of MCDA methods (as specified by \cite{11}). It allows any DM to easily organize a certain criteria set into a hierarchy. This approach consists of associating "playing a card" with each criterion.

The results of this method are shown in the corresponding output module in a table containing the name of the criteria and the respective non-normalized, normalized or both weights (depending on the "weight type" parameter defined by the user).

7. SOLUTION EVALUATION
According to \cite{12}, a system should be evaluated from some different perspectives: usability, functionality and user experience. A great evaluation takes these three perspectives into consideration, so that the evaluated system can be fairly judged and compared with other similar systems. For that matter, it was necessary to develop a user manual, system evaluation plan, system evaluation guide and an usability survey. The tested version of DecSpace was 0.9, but after the analysis of the results, the system was updated to the version 1.0. It was also validated by supporting two different complex situations.

7.1 System Evaluation Plan
The system evaluation plan was built with the objective of clarifying the various aspects of the evaluation before it occurred, so that valid results could be achieved and that the test sessions would occur as smoothly as possible. The purpose of the evaluation was to detect any usability or user experience issues in DecSpace. The tests were performed in the laboratories of Instituto Superior Técnico - Campus Taguspark or via Skype, depending on the availability of the participants. The evaluation was carried out during the first two weeks of August of 2017. Each test session had an average duration of 50 minutes and consisted of four different phases: preparation (make sure the equipment and participant are ready), introduction (make a brief presentation of MCDA and DecSpace), tasks (the user completed the assigned tasks) and usability survey (the user answer a brief survey regarding the topics of usability and user experience). The users that participated in the system evaluation had little to no experience with decision support systems or MCDA. A total of 20 users participated in the evaluation. The participants had two kinds of help available to them during the test sessions: the user manual and the coordinator. During the whole evaluation process, the coordinator collected the following metrics: time to complete each task, successful task completion, user errors per task (critical or non-critical), help per task, user satisfaction and experience, and likes, dislikes and recommendations.

The stipulated objectives were: time to complete task 1 - 1 minute and 30 seconds; time to complete task 2 - 1 minute and 30 seconds; time to complete task 3 - 4 minutes; time to complete task 4 - 3 minutes; time to complete task 5 - 4 minutes and 30 seconds; time to complete task 6 - 3 minutes and 30 seconds; time to complete task 7 - 1 minute; successful task completion - it is expected that all the participants are able to complete all the assigned tasks; number of non-critical errors per task - 1.5; number of critical errors per task - 0.1; number of user manual helps per task - 0.5; number of coordinator helps per task - 0.5; usability and user experience questions - it is expected that all questions have, at least, an average score of 3.5. It was determined that DecSpace would be considered a successful system, if it met 80% of its objectives.

7.2 Results Analysis
In order to substantiate the statement that DecSpace provides a useful and intuitive solution for the use of MCDA methods and workflow building, it is necessary to compare the objectives with the results obtained in the evaluation process. The results of the collected metrics are presented in Table 2 and the results from the usability survey are shown in Table 3.

A great performance should have its scores below the spec-
Table 2: Average of the collected metrics during the system evaluation.

<table>
<thead>
<tr>
<th>Task</th>
<th>Time</th>
<th>Successful Completion</th>
<th>Non-critical Errors</th>
<th>Critical Errors</th>
<th>Manual Help</th>
<th>Coordinator Help</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1:01</td>
<td>Yes</td>
<td>0.17</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0:51</td>
<td>Yes</td>
<td>0.17</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>3:25</td>
<td>Yes</td>
<td>1.83</td>
<td>0</td>
<td>0</td>
<td>0.17</td>
</tr>
<tr>
<td>4</td>
<td>2:49</td>
<td>Yes</td>
<td>2.17</td>
<td>0</td>
<td>0.3</td>
<td>0.83</td>
</tr>
<tr>
<td>5</td>
<td>3:30</td>
<td>Yes</td>
<td>1.7</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>2:35</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>0.5</td>
<td>0.17</td>
</tr>
<tr>
<td>7</td>
<td>0:44</td>
<td>Yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 3: Average score of the survey questions.

<table>
<thead>
<tr>
<th>Question</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Only the features &quot;Connecting two modules&quot; and &quot;Disconnecting two modules&quot; had an average score under 3.5</td>
</tr>
<tr>
<td>2</td>
<td>4.17</td>
</tr>
<tr>
<td>3</td>
<td>All the listed features had an average score above 3.5</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>4.5</td>
</tr>
<tr>
<td>6</td>
<td>3.83</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>4.3</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
</tr>
</tbody>
</table>

ified objectives. All users were able to successfully finish all of the proposed tasks and the average times for completing them were all below the objective times. The non-critical errors were above the expected on the tasks 3, 4 and 5. There was an average of critical errors above 0.1 in task 5 as well. The average use of the user manual did not surpass the specified objective at any time. The average use of the help of the coordinator to complete a certain task did not exceed 0.5 with the exception of task 4.

Regarding the survey questions, only question 1 did not meet the objective of having a minimum score of 3.5. The features "Connecting two modules" and "Disconnecting two modules" were considered the less intuitive by the users and had a score of 3 and 3.3, respectively.

These results lead one to accept DecSpace as a proper and trustworthy solution to the depicted problem, since 88.5% of the proposed objectives were met.

The participants liked the simplicity of the interface the most, while some of the icons used in certain buttons were not easy to understand. They also found it hard to connect different modules. They also recommended the implementation of a confirmation in the modal boxes; an alternative button to redirect to the "My Projects" page; a mechanism to easily duplicate data; and use a proper color scheme for "Yes" and "No" buttons.

Two failures were detected during the tests and properly identified, which were: extra empty data rows appeared after importing a workflow and duplicated projects were incorrectly identified, i.e. were assigned the wrong copy number. The failures were corrected and the recommendations were implemented, which resulted in the version 1.0 of DecSpace.

8. CONCLUSIONS AND FUTURE WORK

The growth of the MCDA research field and the expanding of the spectrum of domains of use that it can be applied to led to the necessity of a broader and more inclusive software solution that can support the methods developed by MCDA.

DecSpace resulted from the study of the current MCDA frameworks and from the analysis of their best and worst features. Also, it was designed considering various MCDA methods and their characteristics, so that it could provide support to most of them. This way, the usability and user experience subjects were intensively studied in order to correctly approach them during the design of the new solution.

The original prototype also had an enormous influence on the ideals and requirements of DecSpace, since it was the first software solution to actively try to resolve the usability and user interaction issues attached to use of MCDA methods of different natures.

DecSpace continues to tackle those issues by being a web-based application with the intent of facilitating the building of workflows that make use of MCDA methods. This way, DMs can easily formulate and find a possible solutions to their particular problems without the need of an intensive understanding or experience with DSS or MCDA methods. It also has the objective of making available many different MCDA methods and, more importantly, make it easy for the developers to implement new ones, so that the application can keep up with the growth of the MCDA field with little programming effort.
The version 0.9 of this new system was evaluated with the participation of 20 users. After a careful analysis of the results, it was concluded that DecSpace can be accepted as an easy-to-learn application even for non-expert users, who were able to quickly understand and get a grasp of its features, since the stipulated objectives were met. It was received with very positive feedback, but also with some improvements to make as well, which led to the correction and development of some features that resulted in the version 1.0 of DecSpace.

The system validations were successful too, i.e. the users were able to correctly build and execute their workflows and the system was capable of handling and supporting real-world problems.

It can be concluded that the developed application presents an innovative approach to the usability and user interaction issues that most of the DSS suffer from. Its most notable characteristic is its simplistic design and the most praised feature is the flexible workflow building.

As mentioned before, this dissertation was focused on the usability and user interaction issues attached to the use of MCDA methods, therefore the building of the application was devoted to its user interaction and extensibility characteristics. This way, there is an opportunity to greatly expand the platform with many different MCDA methods, since that the version 1.0 of DecSpace only makes five methods available.

The platform should deploy an interface that is able to communicate with external servers. This way, some MCDA methods can be executed outside of the application, which implies that there is not the need of implementing those methods in the platform (e.g. the server of diviz, see ??, could be used for this purpose).

It would be very advantageous to have a recommendation system to MCDA methods. This way, non-expert users that have a particular problem at hand would just have to describe it and the system would suggest the most adequate methods.

Most of the results of the available methods are presented in the form of a table, but it might be interesting to display that data in different charts, so that DMs can better analyze it. The framework "d3.js" (a JavaScript library specialized in data visualization) could be used for that purpose.

The Inquiry method does not provide all of the functionalities proposed by the Delphi technique. Therefore, it is suggested that the Inquiry method is changed to make better use of the concept of round and to allow the incorporation of the output of previous rounds.

DecSpace would benefit from a mechanism that would differentiate its users as well, which would consist of categorizing the users by their level of knowledge of MCDA and by their experience with DSS. Less experienced users would be guided by a step-by-step tutorial in order to correctly set up the MCDA methods, while expert users would be able to freely use the workspace (as it is implemented in version 1.0).

9. REFERENCES


