Journal of
open research software

Stieg, J, et al. 2019 UN-CODE: Software for Structuring and Visualizing Collective Decision-Making Based on Qualitative Data. *Journal of Open Research Software,* 7: 25. DOI: https://doi.org/10.5334/jors.246

SOFTWARE METAPAPER

UN-CODE: Software for Structuring and Visualizing Collective Decision-Making Based on Qualitative Data

Julian Stieg[1], Peter Marks[2] and Lasse Gerrits[1]

[1] Innovative and Complex Technological Systems, University of Bamberg, Bamberg, DE

[2] Department of Public Administration, Erasmus University Rotterdam, Rotterdam, NL

Corresponding author: Peter Marks (marks@essb.eur.nl)

UN-CODE is a web-based tool for structuring and visualizing collective decision-making processes using qualitative, case-based data. It offers a database management tool and visualization method in one. The structure of the database and the visualizations derive from a model that is rooted in evolutionary biology and that has been transformed for social scientists. It features three principal dimensions: problem and solution definitions (PSD), weighted connectedness (c_score) as a network measure, and fitness (FIT) to describe the probability of actors reaching their goals in the collective decision-making process. The results are visualized in a scalable 3D-environment that shows the main dynamics of such in one quick overview.

Keywords: Decision-making process; Fitness Landscape; Complexity Sciences; Social Sciences; 3D Visualization; case-study management

Funding statement: This research has been funded by The Netherlands Organisation for Scientific Research research grant no. 451-10-022.

(1) Overview

Introduction

Collective decision-making processes concern situations where two or more actors aim to push through their ideas and preferences about a certain issue. There is a long history of research on collective decision-making processes. A considerable part of that research utilizes qualitative, case-based data. Such data is notoriously hard to store and utilize in a structural fashion. We developed a model and a method with which such data can be processed [1]. To assist researchers using the method we developed the software tool UN-CODE (**UN**derstanding **CO**llective **De**cision-making). The tool can be found at www.un-code.org. UN-CODE can be used to store, structure, process and relate (unstructured) qualitative data, which subsequently can then be visualized for analytical purposes. We first explain the background of the method before discussing the details of the software.

Collective decision-making processes take place over time: there is a starting situation, several negotiation stages, and an outcome. The proposed method maps those processes as sequences of events [2]. Grouped sequences form a lineage. Major events may demarcate the start or end of such a process (cf. the rounds model [3]). Each period between such major events is captured in a (somewhat) static situation called the fitness field. Each fitness field is populated by the relevant actors. The positioning of each actor is primarily mapped in two dimensions. The first dimension represents the 'problem and solution definitions' (*PSD*). This expresses the substantive stance an actor has in the face of a certain collective issue. The second represents the connections between actors, as expressed in the *c_score*. Connectedness is the number of actual links in a network as ratio of the number of possible links (see *density* in social network analysis [4]). Following Abbott [2], the researcher has to select and connect the events in a plausible way; i.e. the researcher will have to reconstruct what has happened and how it has happened.

In the unfolding of the decision-making process actors will find out which elements of their *PSD's* are similar to elements of other actors' *PSD's*. (Dis)similarities in the problem and solution definitions mean actors (dis) connect on content. That is, actors' connectedness is affected by the extent to which *PSD's* converge or diverge. Conversely, elements in *PSD's* are affected by interaction with other actors, i.e. actor connectedness affects the *PSD's* of actors. In other words, connectedness and PSD's are configurational [1]. An adjustment based on content (similar elements of the *PSD's*) enters, attributing weight (*w*) to the *c_score* for every actor. UN-CODE makes automated model calculations for this. *PSD* and *c_score* are visualized on the *x* and *y*-axis, respectively.

Goal attainment means that the collective decision-making process is concluded in favor of an actor or a group of actors and occurs when certain *PSD*'s are fulfilled. In the model, this is expressed by fitness (*FIT*), and is visualized on the *z*-axis. Fitness is attributed on the basis of the empirical data because *any* configuration of *PSD* and *c_score* can be associated with fitness gains or losses. Which particular *PSD* comes out on top in the short and long run is not a given (e.g. [5]) as it depends on the situational feedback as well as the strategies followed by actors. As such, researchers will have to interpret the data and assign scores by themselves. UN-CODE will help in structuring and visualizing the data.

When all data is processed in UN-CODE, it outputs a string of fitness fields. Each fitness field provides a snapshot of the positions of actors relative to each other for that particular time frame, as well as the extent to which their goals were reached. By looking at the fields as they occur over time, one will be able to discern if and how the field changed over time, what strategies were used by actors in their attempts to reach their goals, the persistency of certain *PSD's*, and what *PSD*'s have survived the selection process. As such, the researcher will be able to associate certain strategies and inter-actor dynamics with outcomes. In addition, UN-CODE can output so-called persistence maps, which are overviews of the survival (or demise) of certain *PSD*'s. The approach is based on the adaptive fields and subsequent iterations in the form of fitness landscapes from evolutionary biology. Readers interested in the background of this approach are kindly referred to [1, 6, 7].

There was no software, open source or otherwise, available that could process and visualize the data in the ways intended. Consequently, we developed UN-CODE, to be found at www.un-code.org. UN-CODE is implemented as a web-software in order to facilitate (distributed) research teams to work with the software. With this contribution, we release the source code as open source software to the scientific community to work and experiment with it.

Implementation and architecture

UN-CODE is implemented as an open source web-based software. Due to its web-based nature, it can either be accessed using www.un-code.org – a web space hosted by the scientists behind the project – or it can be downloaded and installed on a local standard web server, such as XAMPP.

The backend (PHP, SQL) and the frontend (HTML, JavaScript, JQuery, WebGL) are combined using the Smarty template engine, in order to provide a clear distinction of markup language (HTML) and backend language (PHP). Throughout the front-end, many dynamic AJAX functionalities – asynchronous server-interactions – are used to provide a fluid user experience and to provide advanced functionalities, such as to facilitate 3D visualization or data management.

All client requests are first being captured by the main index.php file. At this point, two kinds of distinctions are made. First, is the request (a) targeted at loading a new front-end file (i.e. a different Smarty templates), or (b) is the backend required to provide an answer to a dynamic AJAX call? Secondly, is the request being made in by (a) an authenticated user that has logged in before (shown by cookies tied to a specific user ID) or by (b) an unauthenticated user? When necessary, database queries are made by PHP's MySQLi interface towards the MySQL database server. UN-CODE employs a separate MySQL databases for security purposes: one specifically for storing sensitive user data, such as passwords (1 table) and one specifically for entered case data (2 tables + 5 per case). User passwords are encrypted using SHA-512 hash generation.

The 3D visualization functionality of UN-CODE is done within the browser using the Three.JS library, taking advantage of the WebGL interface – thus enabling the software to use real 3D graphic acceleration from within any compatible browser. This approach allows a dynamical (rotation, zoom) and on-the-fly data visualization in a scalable environment. Although a dedicated video card is recommended for improved performance, graphics chips supporting OpenGL 2.1 or higher also work.

As UN-CODE is also a data management tool, it provides the possibility of storing (qualitative) data and data sources connected to the study, such as documents or images up to the size of 32 Mbytes per file in any format, directly online on the server. The files are uploaded via the browser and stored in the case study database only, accessible from within a personal account. This account is password-protected, stored encrypted in the database, all data is confidential and only accessible through the personal account (all user passwords in the UN-CODE database are stored with the secure SHA-512 algorithm), offering always-online availability and data protection at the same time. **Figure 1** presents an overview of the system architecture.

Illustrated example

UN-CODE has already been used for various studies [1]. There are two specific forms of output generated by UN-CODE: 3D visualizations and persistence maps. The data entry for a study (in this illustration for the Sports in the city case study) is done in a main input mask as presented in **Figure 2**.

3D visualization (see Figure 3)

The data, once processed, are rendered on three axes: *PSD* on the *x*-axis, *CON* (i.e. final *c_score*) on the y-axis and *FIT* on the *z*-axis. This complies with the common 3D representation of fitness landscapes in other literature. The results of *PSD* and *CON* are normalized between 0 and 1 to allow for clear visual representation. Researchers can opt to make the *PSD* values relative to the whole lineage or to certain selected fields in the options menu. This can be useful in those instances where actors keep entering and leaving the fields, i.e. when their presence is intermittent.

Researchers can select any combination of fields and actors to visualize. This allows the observation of a specific configuration at a specific point in time. Or, if multiple fields are selected, it allows monitoring the movement of one or multiple actors throughout the lineage. The movement can also be highlighted by using arrows to connect actors' positions in time. Different time positions

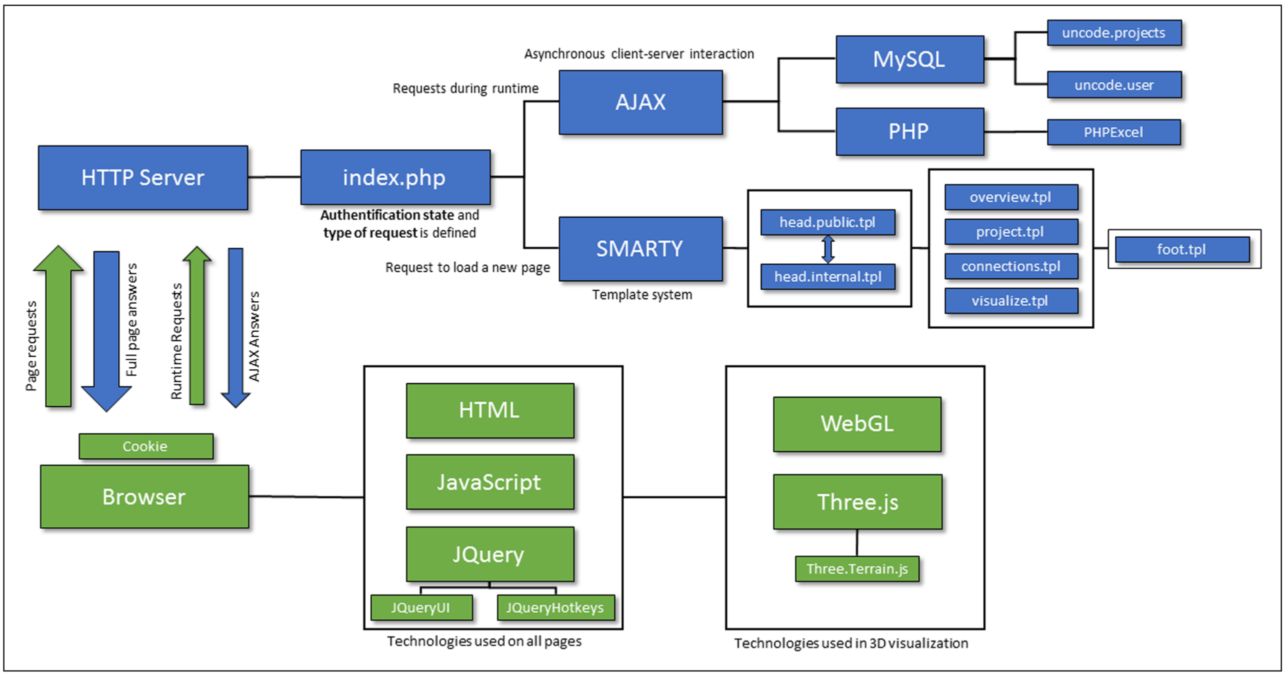

Figure 1: Overview of the un-code.org system architecture and utilized technologies.

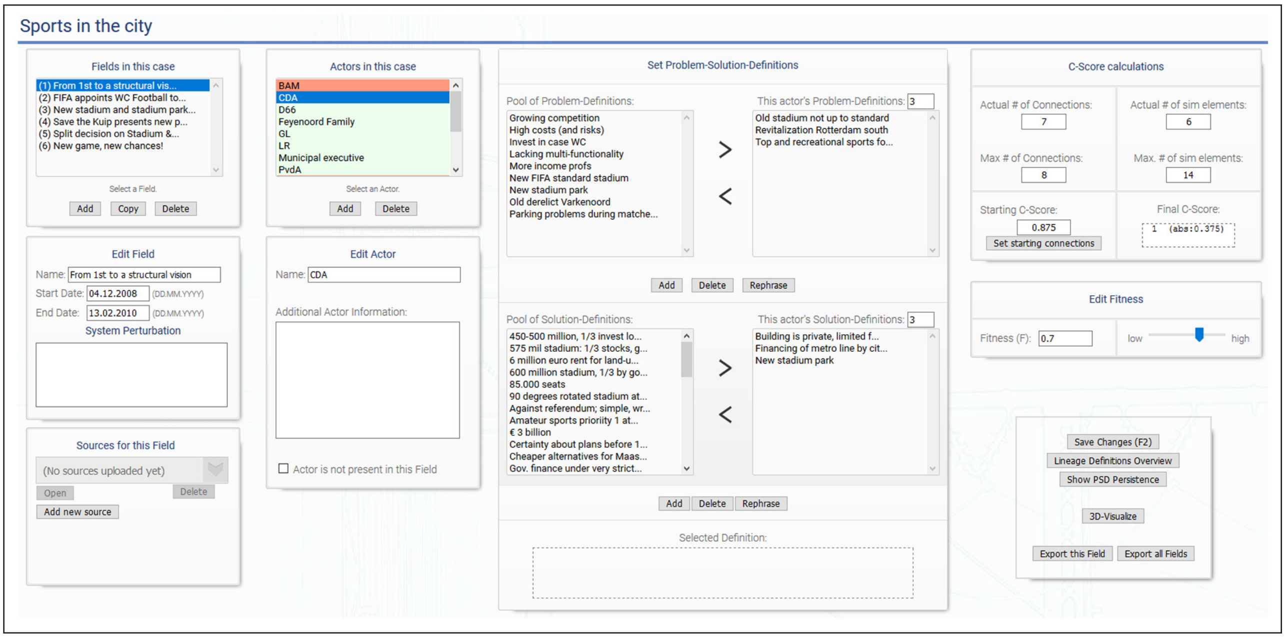

Figure 2: UN-CODE's main data input mask. From left to right: fields/sources, actors, problem/solution definitions, *c_score* calculation/fitness, visualization/export methods.

are labelled with *t1…tn*, which depending on preference can be turned off or on. Also, each earlier positions can be rendered transparent to aid visually.

To allow for detailed examinations and view small iterations within the visualization, the camera view features free 3D-rotation by holding the left mouse button or using the arrow keys. Zooming is supported using the mouse wheel. 3D-labels for actors and grid will always rotate facing the camera. UN-CODE has a function to return to a standard original camera position, offering comparability between snapshots. The output can be saved as *.jpg files. Options include (1) colored or gray scale output, (2) highlighting one or more specific actors in the field, and (3) sloped, peaked and columned fitness representation. Whenever selecting any option, changes to the visualization apply instantly and no reload of the web page is required due to the utilization of AJAX and WebGL-abilities. These options help in uncluttering the visual information in crowded fields and lineages.

Persistence mapping (see Figure 4)

Over time, the interaction and alignment between actors in search for fitness produces a number of substantive outcomes, i.e. certain problem and solution definitions

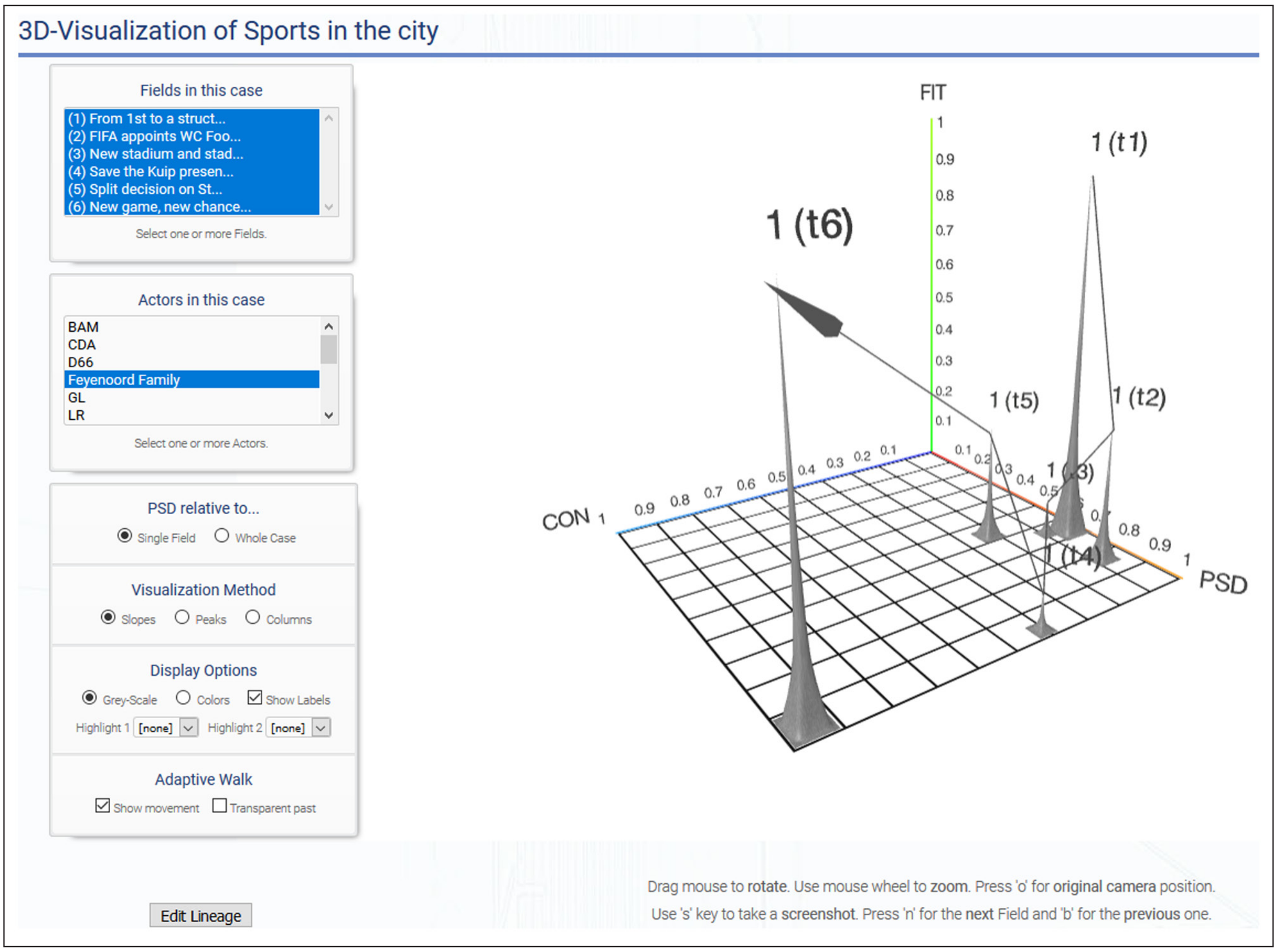

Figure 3: UN-CODE's 3D Visualization: A typical case examination, highlighting the movement of an actor throughout multiple fields in relation to other actors. Here, the actor in question is highlighted in grey, but can also be shown in color. The arrow shows the route that the actor followed through time. Alternatively, one can select a number of actors to be visible at a given time stamp.

PSD persistence "Future of the Gotthard Region"	First initiative Porta Alpina (t1)	Initiative council of Graubunden (t2)	Council of States requests Bundesrat (t3)	Bundesrat requests concept Gotthard (t4)	Release Gotthard development concept (t5)	MoU between cantons signed (t6)	No further pursuing Porta Alpina (t7)	Release PREGO concept San Gottardo (t8)
Problem Definitions								
Lack of clear strategic concept for the region				100% (1/1)	100% (5/5)			
Lack of cooperation among actors in the region					100% (5/5)			100% (5/5)
Projected costs of Porta Alpina			100% (2/2)					
Time pressure pairing Porta Alpina with GBT							100% (2/2)	40% (2/5)
Unknown cost/benefit ratio Porta Alpina			100% (2/2)	100% (1/1)				
Weak development of tourism						20% (1/5)		40% (2/5)
Weak economic situation in Surselva	100% (1/1)	100% (1/1)	100% (2/2)	100% (1/1)				40% (2/5)
Weak infrastructure links of region to rest of Switzerland	100% (1/1)	100% (1/1)	100% (2/2)	100% (1/1)				20% (1/5)
Weak social-economic structure of the region					100% (5/5)	100% (5/5)		100% (5/5)
Solution Definitions								
Diversification of the regional economy						100% (5/5)		100% (5/5)
Initiate regional projects such as Parc Adula						20% (1/5)		
Investigation of possible solutions			100% (2/2)					
Maintain frequent railway connections with the rest of Switzerland						20% (1/5)		
More cooperation and coordination among actors in the region					100% (5/5)	100% (5/5)		100% (5/5)
Promote the region								100% (5/5)
Promote tourism						20% (1/5)		100% (5/5)
Build station inside GBT and to use the emergency exits as access to the station.	100% (1/1)	100% (1/1)	50% (1/2)			60% (3/5)		
To develop a joint strategy for the future of the Gotthard region				100% (1/1)	100% (5/5)	80% (4/5)		

Figure 4: UN-CODE's Persistence Mapping: Helping to see which definitions survived over time. The example shown here is slightly adjusted from [1].

are retained while others disappear. Tracing the evolution of *PSD's* over time, and the positions and actions of actors tied to those *PSD's*, will give a thorough insight into which options survive in the long run. For this reason UN-CODE offers the possibility of mapping the persistence of *PSD's* throughout a lineage. The output is done per lineage as an automatically created *.xlsx sheet (using the library PHPExcel).

In this output, the fitness fields of the selected lineage are arranged by time on the x-axis, the *PSD*'s used anywhere in the study are presented on the y-axis, group-separated by problems and solutions. Each cell shows the percentage of actors that share this definition within this very field. Next to the percentage, the exact number of sharing actors and the total number of actors active during this field are shown. The strength of shared definitions is highlighted through different shades of grey – with black representing non-shared, dark grey weakly-shared definitions and light grey representing broadly-shared definitions.

There is also the opportunity to present two extra columns on the right to show the number of total shares of this definition throughout the lineage, and the persistence score. The latter expresses in how many fields of the lineage a definition was active, i.e. shared by at least one actor. This presentation allows the researcher to keep track of definitions getting weaker or stronger at specific points in time and helps pointing out possible explanation approaches why some definitions ended up being successful. It is possible to render the output in colors for other presentation purposes.

Quality control

UN-CODE has been developed in an agile iterative process with development and feedback cycles over a duration of almost two years. Each iteration was put through both technical and functional testing. Technical testing consisted of going through each program component and searching for technical bugs or incompatibilities. Functional testing was done by the academic team and other numerous individuals in order to find scientific implausibilities and to determine in what ways the functioning of UN-CODE had to be altered or expanded. The feedback on each version has been incorporated in each subsequent release.

(2) Availability

UN-CODE is available as open source to every interested individual or scientific organization, either to be used online under www.un-code.org (hosted by the creators) or to be downloaded via GitHub (https://github.com/uncodecomplexsystems/Un-Code) and to be installed locally using a common http server and a MySQL server.

There are tutorials to show the workings of UN-CODE, how accounts can be created, how data can be stored and processed, and how output can be created (http://un-code.org/?page_id=39). A technical information page explains what is necessary for full (local) usage and how to contact the developers in case of extra questions (http://un-code.org/?page_id=82). Also, users can create a demo account to check the functionality of the software (http://un-code.org/app/). This demo account will automatically reset for each user. For actual usage of the software, users can create the aforementioned personal account for safe storage of their data and related analyses.

Operating system

UN-CODE has been tested in modern web browsers, including the most widely-used ones, i.e. Google Chrome and Mozilla Firefox. Due to its web-based architecture, it can be used independently of the operating system.

Programming language

UN-CODE has been written using PHP 5.6 (www.php.net). The database queries have been written in MySQL 5 (www.mysql.com). The templates have been written using the Smarty code of the Smarty template engine. Common JavaScript and JQuery (www.jquery.com) code is used throughout the software. The 3D visualization component includes code written in WebGL and THREE.JS (www.threejs.org).

Additional system requirements

On the server side, UN-CODE's file upload component benefits from php.ini settings of memory_limit (recommended: 64M or more), and upload_max_filesize (recommended: 16M or more).

On the client side, UN-CODE has been designed towards Google Chrome or Mozilla Firefox and a screen resolution of 1680x1050 or higher. Additionally, the 3D visualization component – utilizing WebGL technology and therefore hardware 3D-acceleration –benefits heavily from a dedicated graphics card. Even though, the visualization will still work with modern onboard GPUs, frame rate and responsiveness are improved dramatically with a dedicated one (e.g. Nvidia or AMD hardware).

Dependencies

All included dependencies are open source.

- JQuery, version 2.1.3 or higher (MIT license): http://www.jquery.org
- Smarty, version 3.1.21 or higher (GPLv3): https://www.smarty.net/
- PHPExcel, version 1.8.1 or higher (LGPL): https://github.com/PHPOffice/PHPExcel
- Three.JS, version R73 or higher (MIT license): https://threejs.org/
- Three.Terrain.js, version 1.2.1-20150607 or higher (MIT license): http://www.isaacsukin.com/coding

List of contributors

1. Julian Stieg: Development and programming.
2. Peter Marks: Development, documentation and testing.
3. Lasse Gerrits: Development, documentation and testing.

Software location

Code repository

The code is available via GitHub and is published as-is under the open source license GPLv3.

Name: un-code
Identifier: https://github.com/uncodecomplexsystems/Un-Code
License: GNU General Public License v.3 (GPLv3)
Version published: 1.0
Date published: 28/07/2018

Language
English

(3) Reuse potential

UN-CODE has a high reuse potential "as is" and has already been a useful tool for researchers working on different projects from a wide spectrum of collective decisions-making processes. Future development will be driven by user requirements and needs.

Competing Interests
The authors have no competing interests to declare.

References

1. **Gerrits, L M** and **Marks, P K** 2017 *Understanding Collective Decision Making: a fitness landscape model approach.* Cheltenham, UK: Edward Elgar. DOI: https://doi.org/10.4337/9781783473151
2. **Abbott, A** 2001 *Time Matters: On Theory and Method.* Chicago: The University of Chicago Press.
3. **Teisman, G R** 2000 Models for Research into Decision-making Processes: on Phases, Streams and Decision-making Processes. *Public Administration,* 78(4): 937–956. DOI: https://doi.org/10.1111/1467-9299.00238
4. **Tichy, N M, Tushman, M L** and **Fombrun, C** 1979 Social Network Analysis for Organizations. *Academy of Management Review*, 4(4): 507–519. DOI: https://doi.org/10.5465/amr.1979.4498309
5. **John, P** 1999 Ideas and interests; agendas and implementation: An evolutionary explanation of policy change in British local government finance. *The British Journal of Politics & International Relations*, 1(1): 39–62. DOI: https://doi.org/10.1111/1467-856X.00003
6. **Gerrits, L M** and **Marks, P K** 2015 The evolution of Wright's (1932) adaptive field to contemporary interpretations and uses of fitness landscapes in the social sciences. *Biology and Philosophy*, **30**(4): 459–479. DOI: https://doi.org/10.1007/s10539-014-9450-2
7. **Marks, P, Gerrits, L** and **Marx, J** 2019 How to use fitness landscape models for the analysis of collective decision-making: a case of theory-transfer and its limitations. *Biology & Philosophy*, 34(1). DOI: https://doi.org/10.1007/s10539-018-9669-4

How to cite this article: Stieg, J, Marks, P and Gerrits, L 2019 UN-CODE: Software for Structuring and Visualizing Collective Decision-Making Based on Qualitative Data. *Journal of Open Research Software,* 7: 25. DOI: https://doi.org/10.5334/jors.246

Submitted: 05 September 2018 **Accepted:** 19 July 2019 **Published:** 31 July 2019