Designing and assessing an interactive sunburst diagram for ICD

Viktor Lindholm
Abstract

Designing and assessing an interactive sunburst diagram for ICD

Viktor Lindholm

The international classification of diseases (ICD) is used in healthcare as a health care classification system. The classification provides a system of diagnostic codes for classifying all diseases. The diseases includes nuanced classifications of a wide variety of signs, symptoms, abnormal findings, complaints, social circumstances, and external causes of injury or disease. The current digital systems for navigating through medical records are in some areas sub-optimal and could require an extensive revision. When designing a system for navigating through a database, in this case a set of diagnostic codes, retrieving to familiar layouts such as lists of links could intuitively become the default approach. However, resorting to familiarity, as done with the current configuration of digital ICD navigation, may leave beneficial perks unexplored. For this thesis, an interactive sunburst diagram for the ICD codes was created to explore areas in which a more unorthodox navigation model could possibly excel - This included measuring ease of use, navigation efficiency measured in time to achieve specified goals and number of clicks accumulated when navigating. The results gave no clear indication that a sunburst diagram used in this domain would be disadvantageous or inadequate but rather suggested certain benefits. The participants using the Sunburst model for ICD navigation accumulated less clicks, gave a more accurate estimation of how much time was spent to accomplish navigation tasks and also suggested an improvement of ergonomics.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1</strong> Introduction</td>
<td>1</td>
</tr>
<tr>
<td><strong>2</strong> Purpose</td>
<td>2</td>
</tr>
<tr>
<td>2.1 Delimitation</td>
<td>2</td>
</tr>
<tr>
<td><strong>3</strong> Prototype</td>
<td>3</td>
</tr>
<tr>
<td>3.1 Method</td>
<td>4</td>
</tr>
<tr>
<td>3.1.1 Software Requirement Specification</td>
<td>4</td>
</tr>
<tr>
<td>3.1.2 The data</td>
<td>5</td>
</tr>
<tr>
<td>3.1.3 Design choices</td>
<td>6</td>
</tr>
<tr>
<td>3.1.4 Visualization</td>
<td>7</td>
</tr>
<tr>
<td>3.1.5 Interactivity</td>
<td>8</td>
</tr>
<tr>
<td>3.1.6 Framework</td>
<td>9</td>
</tr>
<tr>
<td>3.2 Results</td>
<td>10</td>
</tr>
<tr>
<td>3.2.1 File format</td>
<td>10</td>
</tr>
<tr>
<td>3.2.2 Implementations steps</td>
<td>11</td>
</tr>
<tr>
<td><strong>4</strong> Evaluation</td>
<td>20</td>
</tr>
<tr>
<td>4.1 Tree</td>
<td>21</td>
</tr>
<tr>
<td>4.2 Treelist</td>
<td>22</td>
</tr>
<tr>
<td>4.3 Method</td>
<td>23</td>
</tr>
<tr>
<td>4.3.1 The test</td>
<td>24</td>
</tr>
<tr>
<td>4.4 Results</td>
<td>26</td>
</tr>
<tr>
<td><strong>5</strong> Discussion</td>
<td>29</td>
</tr>
<tr>
<td>5.1 Observations</td>
<td>29</td>
</tr>
<tr>
<td><strong>6</strong> Conclusion</td>
<td>30</td>
</tr>
<tr>
<td><strong>7</strong> Future development</td>
<td>31</td>
</tr>
</tbody>
</table>
1 Introduction

Statistics can be a valuable asset for making progress - regardless of field of study. If the software used for compiling statistics is designed without user experience in mind, collected data can be inaccurate and erroneous. Health care is an organization that might be suffering from this issue. This thesis sets out to make a different take on a significant system used in healthcare - the ICD. An interactive sunburst diagram for navigating through ICD codes will be created. The effects an interactive visualization with animated transitions between states may have for the user experience will be tested.

The International Classification of Diseases (ICD) is the international "standard diagnostic tool for epidemiology, health management and clinical purposes"[1]. The ICD is originally designed as a health care classification system, providing a system of diagnostic codes for classifying diseases, including nuanced classifications of a wide variety of signs, symptoms, abnormal findings, complaints, social circumstances, and external causes of injury or disease. This system is designed to map health conditions to corresponding generic categories together with specific variations, assigning a designated code, up to six characters long. Major categories are designed to include a set of similar diseases. The ICD is published by the World Health Organization (WHO) and used worldwide for morbidity and mortality statistics, reimbursement systems, and automated decision support in health care. This system is designed to promote international comparability in the collection, processing, classification, and presentation of these statistics.

The ICD is important because it provides a common language for reporting and monitoring diseases. This allows the world to compare and share data in a consistent and standard way between hospitals, regions and countries and over periods of time. It facilitates the collection and storage of data for analysis and evidence-based decision-making. However, the usability of Electronic Medical Records has been claimed deficient and a revision is needed[2].

ICD is a domain which is hierarchically ordered. There are countless ways hierarchically ordered data can be visualized, but at the time of writing, no interactive sunburst visualization of the ICD-10 database can be found online. The data can exclusively be accessed by navigating through lists of links. Whether using lists for ICD is chosen as an arbitrary default format or if extensive interactive evaluations were performed is an uncertainty. The performance of the list format will be compared to a sunburst visualization.

A sunburst visualization shows hierarchy through a series of rings, that are sliced for each category node. Each ring corresponds to a level in the hierarchy, with the central circle representing the root node and the hierarchy moving outwards from it (see figure 1). Rings are sliced up and divided based on their hierarchical relationship to the parent slice. The angle of each slice is either divided equally under its parent node or can be made proportional to a value.[4]

The advantages and disadvantages of a sunburst visualization, when implemented with ICD-codes, was explored and was also be compared to a tree view and a tree list of some sort.
2 Purpose

Designing and evaluating an interactive sunburst diagram with animated transitions between states, tailored for ICD-10, is the main purpose of this thesis. Displaying ICD-10 as a sunburst diagram represents a set of challenges. Since ICD-10 contains over sixty thousand codes with various names and number of characters, no assumptions of sunburst size requirements can be made. A problem with a sunburst visualization is that it requires the comparison of angles, instead of lengths, which is a more difficult task to solve. The scale can easily become misrepresented and give the user a false sense of which sector contains the most data.

Labeling the sunburst for optimal user experience is an issue. If the labels are strictly aligning with the arc of a sector, the text could border on vertical orientation and compromise reading comprehension. However, if the labels are programmed to remain horizontal, the maximum limitation on number of characters will decrease since overlapping labels is considered an unacceptable design flaw.

The implementation of the sunburst managed to reach a quality level good enough for testing and the model was compared to two types of tree views. The goal is to find tendencies in areas where each visual representation excels and where it will underperform.

Research questions

- Is a sunburst diagram a viable alternative for ICD navigation?
- What benefits could a sunburst diagram possible offer?

2.1 Delimitation

The focus of the study mainly consists of finding tendencies in areas where a sunburst diagram could be beneficial for management and interaction with ICD codes. Optimizing code and lowering computational effort was not prioritized. As long as the visualization manages to represent the data in an acceptable manner with no transitional animation loss, the minute details of algorithmic optimization was not covered.

JavaScript was the only programming language considered for designing the sunburst diagram and the benefits offered by other programming languages are not examined. If ultimate performance within a web domain was the goal, WebAssembly could have been a viable option[5].

Design evaluation consists of pilot tests and, to some extent, expert evaluations. Only one designed model of a sunburst diagram was tested by participants. The tests consists of comparing a sunburst diagram with a tree view and a tree list.

The final design is not evaluated for integration with current systems in healthcare and only acts as an isolated prototype. The participants did not test the implementation in variable form factors such as screen resolution, screen size, screen brightness, touch-screen or not touch-screen et al.
3 Prototype

The components of the build and the methodology will be covered in this section. The final prototype consists of only one version for eventual user testing and comparison. Figure 1 displays a sunburst diagram with two layers.

Figure 1: An illustration of a sunburst
3.1 Method

The initial step in the process was to construct a fundamental software requirement specification (SRS)[3]. If the most significant components are not identified before the implementation process begins, the risk of straying from the ultimate goal increases. Identifying requirements intuitively during implementation is seldom an efficient model for a project with a determined deadline.

3.1.1 Software Requirement Specification

1. Purpose:
   Identify weaknesses and strengths with a sunburst diagram when managing ICD data.

2. Scope:
   One model with mouse interactivity written in Javascript utilizing the d3.js framework.

3. Description:
   A zoomable sunburst diagram which is able to display and render ICD-10 codes without stuttering.

4. Requirements:
   No stuttering and clearly visible labels. No slice of the sunburst should be too small for a mouse click to be feasible. The sunburst diagram has to be fully visible on a 14 inch monitor with a resolution of at least 1920 * 1080 pixels. Figure 2 shows the hardware specification for the machine that should be able to handle the prototype (see figure 2).

![Figure 2: Target machine (minimum hardware requirement)](image)
3.1.2 The data

An ICD code contains of one letter followed by two or three digits. The letter can be any of the letters in the English alphabet. Each letter holds a specific category of diagnosis. For example, the letter S designates that the diagnosis relates to "Injuries, poisoning and certain other consequences of external causes related to single body regions." S, used in conjunction with the numerals 7 and 2, indicates that the diagnosis falls into the category of Fracture of femur. A three-character category that has no further subdivision (i.e., no greater specificity) can stand alone as a code. Adding an additional digit would make the diagnosis more specific. If 0 was added to S72, becoming S72.0, the diagnosis would change from "Fracture of femur" to "fracture of neck of femur", providing more specific locality and condition.

The 2016 edition of the ICD-10-CM is divided into 21 chapters, based on the subject of the ICD codes each chapter contains. Each chapter is identified by a chapter number and description. The set of ICD codes contained in each chapter is specified by a range showing the first three digits of the code range included. Table 1 shows the format of the chapters.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Code range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A00-B99</td>
<td>Certain infectious and parasitic diseases</td>
</tr>
<tr>
<td>2</td>
<td>C00-D49</td>
<td>Neoplasms</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>20</td>
<td>V00-Y99</td>
<td>External causes of morbidity</td>
</tr>
<tr>
<td>21</td>
<td>Z00-Z99</td>
<td>Factors influencing health status and contact with health services</td>
</tr>
</tbody>
</table>

Table 1: ICD Chapter formatting
3.1.3 Design choices

Before considering design choices to enhance the user experience, a fundamental component had to be resolved. Transitions between levels in the sunburst could not be rendered smoothly if all data was handled simultaneously. Therefore, only the data relevant for the user should be rendered and displayed. Deciding relevance of on-screen information can not always be done intuitively and often requires an evaluation. Since this study was unable to obtain data of the usage of the ICD-10 domain, assumptions of key interactivity would increase the risk of underestimating significant passages and overestimating redundancies. Making the sunburst diagram as uniform as possible should be the initial approach when dealing with unexplored system requirements.

Making the performance of the sunburst diagram good enough to eliminate stuttering transitions can be achieved in many ways. However, as mentioned in the delimitation, optimal performance is not desirable. As long as the program can perform transitions between different layers of the sunburst smoothly without stuttering, further performance optimization, refactoring of code or other improvements are discontinued.

Drawing shapes and figures in a browser environment can be done with a SVG or a canvas. The two methods differ when it comes to creating imagery. Canvas is raster-based while SVG is vector-based. The vector-based model will store the shape as an object in a Document Object Model or scene graph. When the shape is stored, it will be rendered to a bitmap. Bitmap rendering enables automatic re-render of a scene if any attributes of the SVG object are changed.

With a raster-based model, imagery is drawn in immediate mode. When a figure is drawn, the model responsible for making the figure is not saved. This results in the need to redraw an entire scene if any positions of a figure is altered. An SVG model would change the position attributes and the browser would only have to redraw the altered data. However, scene-graph capabilities can be enabled for a canvas model by utilizing JavaScript libraries.

Since the sunburst diagram has to respond to mouse clicks, the model must support such a feature. A vector-based model as SVG enables event handlers connected to figures without specifying co-ordinates. If the same functionality is to be achieved with a raster-based model such as a canvas, the coordinates of the figures has to be manually specified to attach imagery to an event.

The support for event handlers and the use of scene graph without third-party libraries makes SVG a better candidate than canvas for a sunburst diagram implementation. However, SVG requires more computation than a canvas. Using DOM to a great extent decreases rendering speed while a canvas gives high performance well suited for rendering graphics.

Whether SVG, canvas or a combination of both models will be utilized will be further evaluated as the implementation begins and previous works with similar goals are reviewed.

Coloring the sunburst is another aspect to consider. Without giving the component too much attention, some basic color theory could prove an aspect worth considering. For creating balanced color palettes, there is a concept as the 60/30/10 rule which cites how colors should be distributed:[9] 60 % should be a primary color, 30 % should be a complementary color and the last 10 % should be an accent color. It will be investigated whether this rule is applicable to a sunburst diagram.
3.1.4 Visualization

If the sunburst diagram is displayed on a high resolution display of large dimensions, the format of the sunburst can be fully utilized by giving a perspicuous illustration of the hierarchy with all its depth and groups. Since the sunburst diagram has to oblige with the screen dimension constraints of the software requirement specification, not all data can be visible at once. The sunburst has to render new data as the user interacts with the model.

A rough data amount estimation gives an indication that a reasonable visualization would be to only display two levels of the sunburst as the user navigates. Figure 3 shows the top level of a sunburst diagram (See figure 3), while figure 4 displays the view which would appear if the user clicked on the yellow inner layer (see figure 4).

![Figure 3: Initial sunburst diagram](image1.png)  ![Figure 4: After the inner yellow is clicked](image2.png)

With the help of d3.js, the transition between figure 3 and figure 4 can run smoothly with a transition animation.

Labeling the sunburst is a process that was improved iteratively as the development proceeded. Making the labels big enough for visibility but small enough to not overlap other labels and other slices will demand testing and configuration tweaks.
3.1.5 Interactivity

The sunburst diagram responds to user input corresponding a left mouse click on a sector. When a sector is clicked, the dimensions of the new, zoomed in, sunburst is calculated. The radian location for the start and end of the arc, as traversed around the circle, the radian location for the inside arc and the radian location for the outside arc are all components that has to be updated for the new sunburst diagram.

When the new dimensions are calculated, the program utilizes a transition function from the d3.js library. The transition function takes a selection of elements and for each element it applies a transition to a part of the current definition of the element. When it came to how fast the transition should be, 750ms proved to be an adequate number. The Neilson Norman Group wrote that 100ms is perceived as instant, and 1 second is considered the upper limit of a users flow of thought.[8] 750ms proved enough time to render the transition properly while not pushing this upper limit. The new data points are interpolated and filtered.

When a new sunburst is rendered, all remaining data should not be rendered. If all levels and labels of the sunburst where rendered, it would result in an indistinguishable clutter. To deal with the dilemma, parameters where constructed to ensure that labels where not displayed when exceeding sector size and that the sunburst could not render more than three levels simultaneously.
3.1.6 Framework

**D3.js** is a JavaScript library for manipulating documents based on data. D3 helps you bring data to life using HTML, SVG, and CSS. D3’s emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation.

D3 allows you to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document. For example, you can use D3 to generate an HTML table from an array of numbers. Or, use the same data to create an interactive SVG bar chart with smooth transitions and interaction.

D3 is not a monolithic framework that seeks to provide every conceivable feature. Instead, D3 solves the crux of the problem: efficient manipulation of documents based on data. This avoids proprietary representation and affords extraordinary flexibility, exposing the full capabilities of web standards such as HTML, SVG, and CSS. With minimal overhead, D3 is extremely fast, supporting large datasets and dynamic behaviors for interaction and animation. D3’s functional style allows code reuse through a diverse collection of official and community-developed modules.

**JSON** (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

The Document Object Model (**DOM**) is a cross-platform and language-independent application programming interface that treats an XML document as a tree structure wherein each node is an object representing a part of the document[10]. The DOM represents a document with a logical tree. Each branch of the tree ends in a node, and each node contains objects. DOM methods allow programmatic access to the tree; with them one can change the structure, style or content of a document. Nodes can have event handlers attached to them. Once an event is triggered, the event handlers get executed.

**Http-server** is a simple, zero-configuration command-line http server. It is powerful enough for production usage, but it is simple and hackable enough to be used for testing, local development, and learning. This server will be used to run the program on a local server.
3.2 Results

3.2.1 File format

The program reads data files of JSON-format (JavaScript Object Notation). JSON is a compact, open-standard format written in human readable text derived from JavaScript, but is still language-independent. This format was chosen for its ease of use in combination with JavaScript, which has functionality for parsing this format. The files used with the program should have the layout of the example in the code snippet displayed in listing 1.

```
1
2  
3  
4  
5  
6  
7  
8  
9  
10 
11 
12 
13 
14 
15 
16 
17 
18 
19 
20 
21 
22 
23 
24 
25 
26 
27 
28 
```

The extract displays the first 27 lines of the JSON file. The JSON is structured as a four dimensional array with more specific detail corresponding to depth. The first level contains the codes A-Z. The second level contains the chapters corresponding to its parent code letter. The third level contains a code and a name. The code corresponds to the letter of the first level with the addition of two digits. The name gives the first specified ICD diagnosis. The fourth level further specifies level three with an additional digit to the code and and more thoroughly described diagnosis.
3.2.2 Implementations steps

The process from initial start to approved prototype can often be divided into different steps. This subsection aims to describe the different phases of the development.

Since no previous experience with d3.js had been accumulated, the initial step was to become familiar with the library. The d3.js website covers the basis of the library with many examples and forums.[11]

D3.js operates on sets of nodes called selections. Since what the program should accomplish is the main focus, rather than how the program should make the accomplishments, D3.js is categorized as a declarative library. The selector nodes can be configured in multiple ways. Styles and attributes can be altered, event listeners such as a mouse click can be added, nodes can be sorted, removed or added, HTML can be modified etc. D3.js does not strive to meet every need possible, but get the fundamental implementations correct. Since a D3 selection is an array of nodes, access to the underlying DOM is granted. Figure 5 displays the final prototype which was used for pilot testing (see figure 5).

![Figure 5: The final result](image-url)
Find a template

The d3 website is full of listed examples created by the founder Mike Bostock. However, to get a full sense of how a sunburst diagram is created in d3.js, a tutorial seemed to be an appropriate tool. David Richards offers a 4 part introduction to a d3.js sunburst at the bl.ocks website[13]. After following the tutorial, a basic sunburst diagram with labels was made. Figure 6 shows the result of following David Richards four step tutorial (see figure 6).

Figure 6: Finished David Richards tutorial
The template version was not reading data from the ICD-10 JSON file. Changing the code to fetch and display data from the correct JSON file required two tweaks. The first step would be to fetch the correct data. This could be done by simply changing the d3.json line of code:

```javascript
  d3.json("icd-10-3.json")
```

The second step was to change what the appended text should retrieve its data from. With the template, it read from a key named "name". However, with the ICD JSON file, it had to be read from a key called "code" instead. The following line of code had to be changed:

```javascript
  .text(function(d) return d.parent ? d.data.code : "")
```

The result was not visually appealing but at least it was working. Figure 7 displays the result of having labels requiring more space than the slices offer (see figure 7).
Utilize d3.js transitions

Next step would be to actually utilize transitions available in d3.js and only display one, two or at most three layers depending how the graph will look. Once again, a template was found and built upon.[14]. However, when the sunburst diagram had been tweaked to use the ICD JSON file, it did not comply with the requirement in the software requirement specification. The graph could not render without stuttering and some of the slices could not be reached due to a too small scope. As shown in figure 8, the data is still available if the user moves the cursor to a position where data is available but no displayed. Figure 8 shows the filter functionality in action. The data is only filtered visually but still available. This causes a performance hit since all data still has to be loaded regardless of visibility.

Figure 8: Stuttering and missing the letters U, W, X
Load new data instead of filtering

To improve the performance, the program had to be adjusted so that only the data that should be visible is loaded. This was achieved by continuously adding and subtracting data from the sunburst as the user interacts with the model. Initially, the program constructs a diagram with data from only two levels which are drawn by calling an update function. If the user clicks on any of the slices, the program has many cases to cover.

If the users clicks on a slice containing only one letter, the sunburst will transition to a state where the clicked slice is the new root and all children to the root will load their respective children effectively giving the sunburst a third layer only on the clicked slice. Figure 9 and figure 10 illustrates this.

Figure 9: Before click

Figure 10: After click
If the user clicks on a slice with a label of one letter followed by two digits, the program will check if the previous click before the current interaction was on a single letter. If it was, a new level will be added with data corresponding to the users interaction. The previous check is a necessity since it might have skipped the single letter click and instantly gone to a chapter. If a chapter is clicked from the initial sunburst, two layers instead of one will be added. Figure 11 and figure 12 illustrates what happens when clicking on a chapter.

Figure 11: Before click on chapter

Figure 12: After click on chapter
The sunburst will always consist of a maximum of three levels regardless of interaction. The sunburst was no longer stuttering but the labels were not displaying correctly. Figure 13 displays the result of having a sunburst diagram that would only render visible data (see figure 13).

![Sunburst model which fetches new data depending on user interaction](image)
Labeling

When it came to labeling the sunburst, the same logic as adding and removing layers seemed applicable. However, this was not the case. After extensive debugging, refactoring and rethinking, a working prototype was still not finished. The labels did not disappear with its corresponding slice. Figure 14 shows how the sunburst would appear if a user were to navigate throw multiple levels and letter and later return to the initial view (see figure 14).

Figure 14: Sunburst model after multiple clicks
Regress to filter

With a very low hardware requirement, the project was moving in a direction heading for minute optimization. When the stuttering model which relied on filtering and thereby hiding, instead of not loading data, was proven efficient enough for a computer with higher computing power, the software requirement specification was changed. Figure 15 shows the new minimum hardware requirements (see figure 15).

![Figure 15: The new minimum hardware requirement](image)

However, the filter version provided by Mike Bostock has room for improvement. Since a diagnosis can contain many letters, a limitation had to be set to how many letters should be displayed to not overlap other slices. The preferable alternative was to display a maximum of 18 characters. And to provide coherence to the user, only full words were displayed. If the user wants to read the full diagnosis, the title becomes visible by hovering over the slice with the cursor.

To improve navigation and clarity, two layers were added to the middle of the sunburst. The center takes the user back to the start where the second layer takes the user back one level.

The filtering of the graph was another aspect that had to be tweaked. The letters with not much data had to be enlarged to be visible. Figure 16 shows the initial view of the final prototype (see figure 16).

![Figure 16: The final prototype](image)
4 Evaluation

To properly evaluate the sunburst diagram implementation seen in figure 17, it was compared to other models using the same data. Two models were created for the sake of comparison (see figure 18 and figure 19). As for screen occupancy, the sunburst model does not require more screen estate as the user navigates - In contrast to the other two models which requires both vertical and horizontal expansion.

Figure 17: Sunburst

Figure 18: Tree

Figure 19: Treelist
4.1 Tree

The tree model has one root node that can be clicked to display the underlying data. As a computer screen is by default wider than it is tall, it made sense to give the model a horizontal orientation. This would further differentiate it from the treelist model. A blue node can be clicked to display more data while a white node has no underlying data. The nodes change color depending on user interaction. Red labeling further enhances the claim that nodes have different color if they contain more data or not. Additionally, the differentiation in label coloring brings a starker contrast between overlapping labels. As illustrated in figure 20, if node M51 and node M50.9 would have had the same color, M50.9 would have been harder to read (see figure 20). The model can expand both vertically and horizontally. Going deeper in a node will extend the screen occupancy horizontally to the right while opening multiple nodes will extend the screen occupancy vertically.

Figure 20: The tree when opened
4.2 Treelist

This model will resemble the current ICD navigation in healthcare more than the tree model. The specific codes are given different color to decrease an overlap reading experience (see figure 21). This model will extend the screen occupancy vertically as more data is displayed.

---

Figure 21: Treelist opened to reveal codes
4.3 Method

Each model had three participants interacting with the data - for a total of nine participants. Before any interaction began, the participants where informed of what ICD is as well as a brief hint of what type of interaction would follow. The test director gave an verbal representation as follows:

"In healthcare they have something called ICD. The International Classification of Diseases is the international "standard diagnostic tool for epidemiology, health management and clinical purposes". The ICD is originally designed as a health care classification system, providing a system of diagnostic codes for classifying diseases, including nuanced classifications of a wide variety of signs, symptoms, abnormal findings, complaints, social circumstances, and external causes of injury or disease. This system is designed to map health conditions to corresponding generic categories together with specific variations, assigning for these a designated code. Thus, major categories are designed to include a set of similar diseases. I will read an ICD code for your and give you the task of finding the codes corresponding diagnosis. The task will be repeated and I will ask for various diagnoses and also ask for other information. The system will only respond to left click of the mouse and is therefore your only tool of interaction. You are allowed to ask questions which I will answer if I recognize the question as a question that would not defeat the purpose of the study by giving away too much of information. Good luck!".

As of age span for the participants, all of the participants were born in 1990s.
4.3.1 The test

Question 1: Given the code A00.0, can you find its corresponding diagnosis?

As the first code of the ICD database, this question seemed as the correct initial question as it should be the first letter the users read due to its top position. Answering questions of how to achieve the goal will be avoided as the first interaction with a system could give an indication of how the configuration is intuitively understandable.

Question 2: If you were to guess how many codes the model contains, what would be your guess? You are allowed to interact freely with the system

This question fulfills two purposes. The first purpose is to measure if the participant overestimates or underestimates the data set. If the three models differentiate significantly, reasoning of scope perception could be an interesting element to discuss. The second purpose is to give the participant a brief chance to get familiar with navigation before question number three.

Question 3: The participants are given a code that they should find the corresponding diagnosis for:

The first code is C20. When the participant has found the corresponding diagnosis and read it out loud, it is Malignant neoplasm of rectum for this code, a new code will be given. The process will continue for nine given codes, C20, F06.4, J09, R94.6, W76, D71, I50.1, T96, G32.0. The purpose of this test is to accumulate clicks and interaction time for measurements.

Question 4: The user is asked to count number of codes for previously visited codes

The first code is C20. The user is asked to count how many codes are under this code. When they have given the answer 1, a new code will be presented. The participant is done with this question when they have counted the number of codes for C20, C25, F07, T98 and T95. Just as question 3, this part is for click and time accumulation.

Question 5: The user is asked which chapter in B has the most and the least amount of data

This test is for measuring how the user makes sense of the scope of data. If the models differ with their answers, there might be a reason for it.

Question 6: How many chapters does the letter O have?

A question for accumulation of clicks and time with a different formulation.

Question 7: The user is asked to count how many codes a given chapter has (Only children and not the children of the children)

A question for accumulation of clicks and time with a different formulation.
**Question 8: How is ICD constructed?**

When this question is asked, the user can no longer look at the screen nor make any further interactions for any of the succeeding questions. For this question, the user is asked to freely explain the structure of ICD. This question is for measuring how well the user got a sense of the hierarchy and the overall structure.

**Question 9: How many codes do you think there are in this system?**

This is the same question as question number 2. This will give an indication of the user felt as if the underestimated, overestimated or made an initial valid guess regarding scope.

**Question 10: How many clicks do you think you made in total and how much time has surpassed from your first click to your last click?**

This will give an indication of how much navigation the user felt was need for the given tasks.
4.4 Results

As pilot tests were conducted with merely nine participants, the tests will only give hints of how a user interacts. Evaluating the test configuration is more important than the resulting data of a few interactions.

The number of clicks a user has to make to navigate to a desired destination or get access to a data set is an aspect that should intuitively be kept low for improved accessibility. However, blindly following a rule such as the "three-click-rule" has been proven as an inefficient model for improved user experience[15]. Whether counting clicks is at all a valid tool for measuring user experience is a subject to broad to discuss in this section. But gathering data, regardless of measurements, will always enhance the knowledge of a subject. For the three models tested, the number of clicks needed for finishing the given tasks varied.

<table>
<thead>
<tr>
<th></th>
<th>Sunburst</th>
<th>Tree</th>
<th>Treelist</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of clicks (Average)</td>
<td>84</td>
<td>224</td>
<td>224</td>
</tr>
</tbody>
</table>

Table 2: Click comparison

As table 2 shows, the tree and the treelist model required roughly 2.6 times more clicks as the sunburst model to finish the given tasks. This result was somewhat expected as the sunburst model requires far less clicks to display a large amount of data on the screen. If the other two models would have displayed as much data simultaneously, the user experience would have been compromised by forcing the user to scroll - due to lack of screen-estate. Question 5 in the questionnaire was one of the questions where those who used the Sunburst model generated far less clicks. As the Sunburst model gives and indication of data size by scaling a slice according to number of data points encapsulated within, clicking for estimation is not necessary. The other two models requires the opening and counting of data points for estimation.

The participants were asked to guess the data amount at the beginning of the session and when the session was finished. The model stored 13565 codes. Only one of the participants made an adjustment when asked for a second estimation. This participant was navigating in the tree model and lowered the estimation by roughly twenty percentages.

<table>
<thead>
<tr>
<th></th>
<th>Sunburst</th>
<th>Tree</th>
<th>Treelist</th>
</tr>
</thead>
<tbody>
<tr>
<td>data amount (first estimation)(average)</td>
<td>11 500</td>
<td>1 200</td>
<td>9 100</td>
</tr>
<tr>
<td>data amount (second estimation)(average)</td>
<td>11 500</td>
<td>1 100</td>
<td>9 100</td>
</tr>
</tbody>
</table>

Table 3: Data estimation

The participants were asked to guess how much time had surpassed from their first click to their last click.

<table>
<thead>
<tr>
<th></th>
<th>Sunburst</th>
<th>Tree</th>
<th>Treelist</th>
</tr>
</thead>
<tbody>
<tr>
<td>time estimation in minutes (average)</td>
<td>16.6</td>
<td>9.6</td>
<td>9.3</td>
</tr>
<tr>
<td>actual time in minutes (average)</td>
<td>16.6</td>
<td>15</td>
<td>15.3</td>
</tr>
</tbody>
</table>

Table 4: Time estimation

The participants using the sunburst model spent more time on average to finish the tasks when compared to the other models. For those who used the tree and treelist model, the user gave an indication that less time was spent during the task than the actual time.
The performed of the sunburst model differed from the other models in scope estimation for a specific section. When the participants where asked to estimate which chapter in the letter B had the most respectively the least amount of data, the results gave an indication of divergence. All three participants using the sunburst model gave the correct answer when asked which chapter contained the most data. For the other two models, only one of three gave the correct answer. However, when asked for the chapter with the least amount of data, the sunburst model was configured in a manner that deceived the user. Since the slices with a small data set would not display enough pixels for feasible visibility and cursor interactivity, a fix had to be implemented to force slice size when undercutting a threshold. This could result in slices occupying more space than adjacent slices with more data - effectively deceiving a user when asked for scope estimation.

Figure 22 displays the result of not carefully elaborating the result of fixing a small problem such as slice size. If a user has become accustomed with the intuition of slice size representing data size, there would be no hesitation to choose chapter 1-19 as the smallest data set - despite chapter 1-21 being the correct answer (see figure 22). One of three participants chose this incorrect answer and thereby actively displaying the reluctance of dealing with an inconsistent system. If the prototype designed disabled the case of inconsistent slice sizes, the probability of correct answers would increase. As for the other two models, all participants gave the correct answer.

The tree and treelist model generated a significant amount of clicks for this interaction as many nodes and folders had to be clicked for displaying data.
There seems to be a correlation to how the user estimated the duration and how many clicks were performed. The results implies that generating more clicks gives a sense of activity which in return impairs the sense of time. The person generation the most clicks, 244 clicks when using the tree model, also made the least accurate guess by estimating a duration time of seven minutes when the actual time was sixteen minutes. The bar chart as seen in figure 23 shows the correlation between number of clicks and accuracy of guessing estimated time. Given the number of clicks, the corresponding actual duration for the test is divided by estimated duration time to so how close the participant were to guessing correctly. If the estimated duration exceeds the actual duration, the numerator and denominator will be swapped to give a number less than 1. The x value starts at 55 since it was the lowest number of clicks accumulated by a participant.

The plot does not take into account how the different models could be a major factor contributing to time perception but merely measures how number of clicks could contribute. Whether underestimating how much time has surpassed during a task is negative or positive can not be determined by intuition. On one hand, if having fun is analog with loosing track of time, more clicks could give a sense of activity and fruition. On the other hand, planning relies on good time estimation which becomes increasingly difficult if duration estimation has been proven inaccurate.
5 Discussion

A sunburst diagram could be deemed a sound alternative for displaying hierarchical data. How the hierarchy levels are displayed with the root in the middle gives coherence. As the brief pilot tests gave no strong indications against a sunburst diagram as a tool for ICD, with further testing, it might prove an efficient model for enhancing the current inadequate systems in healthcare. However, there are some components in need of a thorough inspection before a potential deployment. As the results indicated misinterpretation of scope for the sunburst model, the prototype would have to consider how the sizes of the slices are designed. The prototype gave a false sense of scope which could negatively affect user comprehension. Programming a slice to not exceed the size of slices with more data does not have to be the single simple answer for user perception. Giving a slice with the majority of data an extensive proportion might prove the best alternative. Testing multiple configurations would filter the optimal model.

The importance of color in a user interface was never examined. Whether users with color vision deficiencies encounter efficiency loss due to color selection remains unanswered. Besides from lowered opacity when no further navigation is possible, the selected color has no correlation to data scope or diagnostic code. A connection between color and diagnosis could prove beneficial if tested properly.

The hardware specification of current systems which would run the sunburst model in an actual deployment is unknown. Optimization could prove an important aspect if the hardware matches or falls short of the initial hardware specification in the software requirement specification.

5.1 Observations

An interesting observation with the participants using the sunburst model was how it may differ in ergonomics. Since many of the slices in a sunburst diagram has labels which are not perfectly horizontal, the user instinctively puts a temporary tilt of the head to align the eyesight corresponding to text orientation. Whether this form of neck activation could prove beneficial for minimizing neck strain and improving ergonomics is a model evaluation that could be interesting to continue.

The pilot test gave an indication that the participants using the sunburst model felt as if more time was spent to finish set tasks. If the trend were to continue as the number of participants increased, there are arguments on both sides as if the data measurement should be interpreted as something positive or something negative. Studies has been made to support the claim that "time flies when you are having fun". When people believe that time has passed unexpectedly quickly, they rate tasks as more engaging, noises as less irritating, and songs as more enjoyable[16]. If interacting with a system for accessing ICD codes is supposed to be fun, the sunburst model might miss the mark with the time estimation aspect. However, the pilot test also gave an indication that the user got the correct time estimation with the sunburst model. A system that gives the user a true sense of how long a task will take, is a better system than one giving the user a false sense of speed and efficiency.

There seemed to be no vast difference when it came to learning curve for the three models. As stated in the test form, hints for the user were kept at a minimum. None of the participants asked for help at a noteworthy rate. The sunburst model could have been flagged as the least intuitive model, given the participants familiarity with treelist structures, but this was not the case - as the model did not generate more questions. Whether d3.js was to most suitable tool for making a sunburst diagram, with the given requirements, is an uncertainty. More time researching alternatives could have proven beneficial.
6 Conclusion

This section will answer the research questions specified in the purpose section.

Is a Sunburst diagram a viable alternative for ICD navigation?

As pilot tests merely gives an indication of possible trends which could be disproved at a larger scale, whether a sunburst diagram is an optimal solution for ICD navigation is uncertain. However, since the tests gave no pronounced indication of inefficiency for the model, it is plausible that a sunburst diagram could be a viable alternative. If the model would have shown a vast discrepancy for duration time to finish the set tasks, it would have been deemed deficient. Aside from efficiency, if the ergonomics of the system would be further investigated and proven beneficial, it could possibly be the key aspect for a potential system change. Neck pain is one of the most significant health problems worldwide.\[18\] It has been ranked the fourth leading cause of years lived with disability, according to the Global Burden of Disease Study.\[19\] If a system could subconsciously improve a known health problem, it would be a step in the right direction.

What should be considered for an actual implementation into current healthcare systems?

A thoroughly constructed survey for those working with ICD codes would be a necessity. Presumptuous needs and requirements could possible render a system heavily emphasizing the redundancies while missing essentials. The prototype has not taken into account conditions such as, color-blindness, blindness, hearing impairment or any neurodiversity. By extending the software requirements specification, these aspect would be covered as needed.
7 Future development

If the user could navigate the sunburst diagram with the keyboard, the implementation could be compared with mouse navigation. Test subjects consistently report that keyboarding is faster than mousing but the stopwatch consistently proves mousing is faster than keyboarding[17]. Whether this statement by Bruce Tognazzini from 1992 rings true for a sunburst model would be an interesting inquiry.

How the model is scaling is an aspect that could be further investigated. Slices could dynamically adjust size and perhaps opacity by calculated relevance or frequency. This could either render incoherence or improve user efficiency depending on implementation format.

A personal adjacent sunburst built upon the diagnosis most visited could prove a reasonable utility. The additional sunburst would act as a dynamic favourites menu for faster accessibility. This could however nurture a behaviour where users select less specific diagnosis out of convenience - effectively increasing erroneous data.
References

[1] The international statistical classification of diseases and related health problems, ICD-10

[2] John B. Smelcer, Hal Miller-Jacobs, Lyle Kantrovich,
Usability of electronic medical records.

Addison Wesley, Acm Press Books, 1995-06

[4] Claus O Wilke, Fundamentals of Data Visualization,
A Primer on Making Informative and Compelling Figures.
O'Reilly Media, Inc, USA, 2019-04

[5] Andreas Haas, Andreas Rossberg, Derek L. Schuff, Ben L. Titzer, Michael Holman, Dan Goth-
man, Luke Wagner, Alon Zakai, JF Bastien,
Bringing the web up to speed with WebAssembly
PLDI 2017 Proceedings of the 38th ACM SIGPLAN Conference on Programming Language
Design and Implementation Pages 185-200

[6] Philipp K. Janert, D3.js for the impatient, O'Reilly Media, Inc, USA 2019-05


[9] Theresa-Marie Rhyne, Applying Color Theory to Digital Media and Visualization, Productivity
Press, 2016-11

O'Reilly Media; 1 edition (February 8, 2013)


06-29

AIS Electronic Library

[16] Aaron M. Sackett, Tom Meyvis, Leif D. Nelson, Benjamin A. Converse, Anna L. Sacket,
You’re Having Fun When Time Flies: The Hedonic Consequences of Subjective Time Progression,
Volume: 21 issue: 1, page(s): 111-117, SAGE journals, May 07, 2009

[17] Bruce Tognazzini, Tog on interface, Addison-Wesley Professional, 1992-02

and national incidence, prevalence, and years lived with disability for 310 diseases and in-
2016;388(10053):15451602.

<table>
<thead>
<tr>
<th></th>
<th>Sunburst</th>
<th>Tree</th>
<th>Treelist</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code estimation (first guess)</td>
<td>2600/2000/30 000</td>
<td>1400/1000/1200</td>
<td>2400/15 000/10 000</td>
</tr>
<tr>
<td>Code estimation (second guess)</td>
<td>2600/2000/30 000</td>
<td>1100/1000/1200</td>
<td>2400/15000/10 000</td>
</tr>
<tr>
<td>Estimated duration (minutes)</td>
<td>20/15/15</td>
<td>10/7/12</td>
<td>10/7/11</td>
</tr>
<tr>
<td>Real duration (minutes)</td>
<td>17/17/16</td>
<td>14/16/15</td>
<td>15/15/16</td>
</tr>
<tr>
<td>Estimated number of clicks</td>
<td>500/25/80</td>
<td>120/52/85</td>
<td>100/250/200</td>
</tr>
<tr>
<td>Actual number of clicks</td>
<td>101/55/96</td>
<td>206/244/223</td>
<td>235/229/210</td>
</tr>
<tr>
<td>Question 5* (Most)</td>
<td>1-15/1-15/1-15</td>
<td>1-17/1-15/1-17</td>
<td>1-17/1-15/1-17</td>
</tr>
<tr>
<td>Question 5* (Least)</td>
<td>1-19/1-19/1-21</td>
<td>1-21/1-21/1-21</td>
<td>1-21/1-21/1-21</td>
</tr>
</tbody>
</table>

Table 5: User data

**Participant data**

*Question 5: The user is asked which chapter in B has the most and the least amount of data. The correct answer was 1-15 for the most and 1-19 for the least.*
Participants describing ICD:

Those who used the sunburst model:

"ICD is built upon letter with diagnosis. The F section is anxiety. First a letter for a category. The letter category has many subcategories."

"In groups and subgroups. Those who belong to the same group belong together somehow. As a catalog structure with a different visual representation. Choose a category which becomes narrower and narrower as a folder. Becomes more specific further down. If you click on a part, you can assume that everything under that part in a specific code is present. As a folder from left to right."

"A system which lets you search for diagnosis which you can access in different ways. You can search by letter or by chapter. Will gradually lead you down to the specifics your are looking for. Filter gradually down for what your looking for. The structure gives more meaning for those with ICD knowledge. As a library."

Those who used the tree model:

"Index under a chapter and then every chapter has a sub chapter. Diseases from the same family are grouped together."

"Some form of tree structure. With chapters, with one or two levels below. Annoying with letters and then chapters and then letters again. The amount of data is not evenly distributed. Hard to find chapter."

"A system to find diagnosis. I think it should have been ordered with only letters and digits such as a license plate. Letters, chapters and digits becomes very incoherent."

Those who used the treelist model:

"First you have letters a-z. For every letter there are chapters 0-10. Under every chapter there are sub folders 0-10 or codes. In the sub folders there are codes 0-10. They are structured as a main group of diseases. order by diseases. How letters and chapters are decided is not apparent."

"Categories a-z. Under every letter there is a chapter. Under every chapter there are diagnosis and perhaps more specific diagnosis."

"Every diagnosis has a code. A code can be a letter followed by a number of digits. The codes are grouped depending on how diseases are related."