Abstraction Hierarchy Tool: A Software for Developing Abstraction Hierarchies as Network Graphs

Joshua Duvnjak[1], Luciana Blaha[1], Theodore Lim[2], Guy H Walker[3], David Flynn[4] & Phillip Greening[1]

[1]Edinburgh Business School, Heriot-Watt University, [2]Institute of Mechanical, Process & Energy Engineering, Heriot-Watt University, [3]School of Social Sciences, Heriot-Watt University, [4]James Watt School of Engineering, University of Glasgow

SUMMARY

The Abstraction Hierarchy (AH) is a Human Factors (E/HF) method used to model the functional structure of complex systems. Although developments have combined the AH with network statistics, this approach is cumbersome to perform. We aim to resolve this issue with our development of a python-based software tool to enable easy access to quantitative statistics within the AH and visualise it as a network graph.

KEYWORDS

Human Factors Methods, Network Theory, Novel Software Tool

Introduction

The Abstraction Hierarchy (AH) is a method for mapping out complex systems from abstract purposes at the top level to tangible artefacts at the lowest level. The approach (often termed as part of a Work Domain Analysis) is robust enough to model small systems such as Apple's iPod (Jenkins et al., 2010) and large systems such as Star Wars' Death Star (Walker et al., 2016). By describing systems in the abstract through the AH, it is possible to discover insights as where there are potential weak points or redundant functions in systems. As systems are becoming ever larger and more interconnected with the rise of approaches such as IoT, Digital Twins and Smart Cities using methods such as the AH are more important than ever. The AH enables this type of analysis by providing a tree structure, decomposing complex goals of the system and values to functions and artefacts interacted with by users.

One problem with the AH method is in its inaccessibility with many Human Factors methods being hand drawn (Holman et al., 2020) and lack sufficient software tools to support research. For the AH the current tool (Jenkins et al., 2007) is difficult to access and does not easily facilitate advances in the method (e.g. graph theory statistics, see McClymont et al., 2022). If practitioners have to spend significant time searching for and setting up software just to apply a new method or approach in a fast-moving business world, it is likely they will seek an alternative method. Tools, like Figma (popular in the UX world; Jain, 2024), are widely used they are simple and easy to learn, even if they do not provide all the functionality of prior wireframing packages.

This work proposes a new solution for generating AH diagrams. On examining current tools, it is clear that a new approach is needed. During our project on digital twins for transport decarbonisation, we found that most prominent tool was lacking in output stability, and there was not a way to export this to other platforms.

We aim to address these shortcomings by designing and developing a software solution using an existing network package (igraph) as the main enabling technology. We call them AH graphs (as opposed to AH diagrams) because they enable the statistical analysis of the hierarchical structure. The following section describes the design, development, and prototyping of our Abstraction Hierarchy Tool (AHT). This work contributes a novel software tool for creating AH graphs with associated network statistics, allowing practitioners to utilise the AH method more widely within research.

Background

Abstraction Hierarchies

According to Lind (1999, p. 199) 'the main principle of the AH is to describe a system on several levels of abstraction' - this makes AH ideally suited for mapping relationships in complex systems. This is typically done over five layers: Functional Purpose, Abstract Function, Generalized Functions, Physical Functions and Physical Form (Rasmussen, 1986). Each layer contains nodes with a text label describing one instance of the layer. For example, one function form could be a car in a wider transportation system. These are connected between the layers. Naikar and Sanderson, (2001) state:

> *'Links from a target function to lower levels of abstraction indicate how a is operationalized or engineered (means). Conversely, links from a target function to higher levels indicate why that function exists(ends).'*

The AH can be used to understand where in systems problems could arise or to identify where certain values are not being met. For example, in the iPod AH (Jenkins et al., 2010) the relationships of different parts and functions of the device can be made more visible. If the "click wheel" ceases to function, the AH shows that the "ease of use" and the function to "receive user inputs" are also compromised. Such insights enable HF designers to design-for or build in redundancy for system-critical components or remove those that do not contribute to the system's goals.

Current Tools

The most prominent tool to create AH is the Cognitive Work Analysis tool (CWA Tool; Jenkins et al., 2007). It also features functionality to perform a range of different methods, such as Decision Ladders. It uses primarily an easily understandable click-and-drag interface and saves files into its own format or exports into commonly available formats. The interface becomes cumbersome when editing medium to large AH, as interface elements "fall off" the screen boundaries meaning it is difficult to drag items to create links. It also appears that on modern operating systems the export feature appears to have visual artifacts causing lines to appear disconnected from nodes. Hingu et al. (2017) described a novel tool with similar base functionality as the CWA Tool that could run on a web browser to improve accessibility for users. However, conversations with one of the authors revealed it was never completed and is therefore unavailable.

Beyond the current purpose-built software solutions, commonly available software could be used to create AH diagrams. Microsoft PowerPoint can be used to draw diagrams. However, due to the structure of AH diagrams it is quickly to one who wishes to use this approach that it is unmanageable as there are too many objects to manipulate.

Prior Usage of Graph Theory

A network is a representation of the edges (connections) between different nodes (e.g. instances of artefacts or actors). The field of graph theory provides ways of calculating how related each of the nodes are to each other. Existing work has bridged graph theory into the AH by calculating eigenvector centrality for each part of the diagram (McClymont et al., 2022). Houghton et al.' s software (2008) features an unspecified graph theory metric that 'is an overall indication of how close a node is (in terms of geodesic distance) from all other nodes in a network'. This approach provides a quantifiable way of understanding which parts of the AH are most critical to the functionality of the system (as shown in McClymont et al., 2022). The graph theory approach has the advantage of identifying both system-critical nodes and nodes that are peripherally connected.

There is a need for new software tool that should also be easily available to researchers who wish to use the software (to avoid inaccessibility issues, e.g. Hingu et al., 2017). Additionally, the tool should enable practitioners to easily apply graph theory methods to quantify the strength of relationships between parts of the AH (e.g. McClymont et al., 2022).

The Abstraction Hierarchy Tool

The Abstraction Hierarchy Tool was developed to address the issues with the current prominent tool (CWA Tool) (Figure 1) (code available at: github.com/Heriot-Watt-Digital-Twin-Project/AbstractionHierarchyTool). These issues can be thought of in three categories: Features, Development and Accessibility. Our proposed tool is designed to provide additional functionality compared with the current approaches, while being developed to provide consistent results and be easily to accessed by HF practitioners.

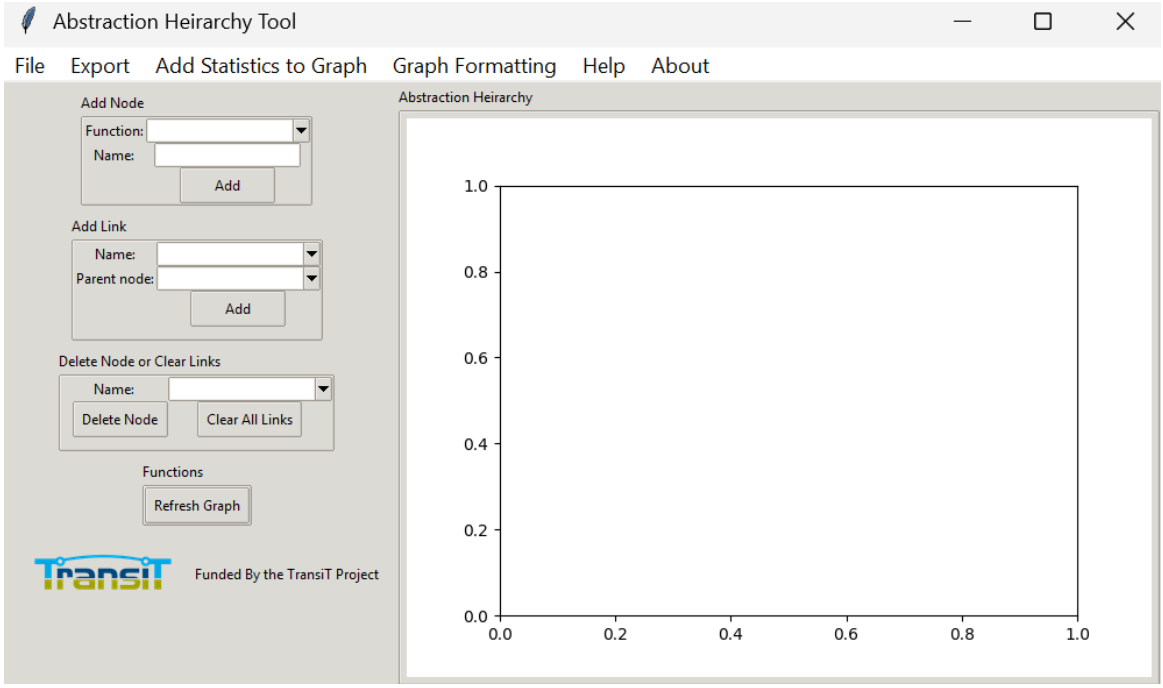

Figure 1: The main interface of the AHT

Our Design Approach

To guide the development of the AHT, we created a set of requirements for each development version of the application (Figure 2). This took the application from an initial Jupyter notebook prototype to a desktop application with a user interface.

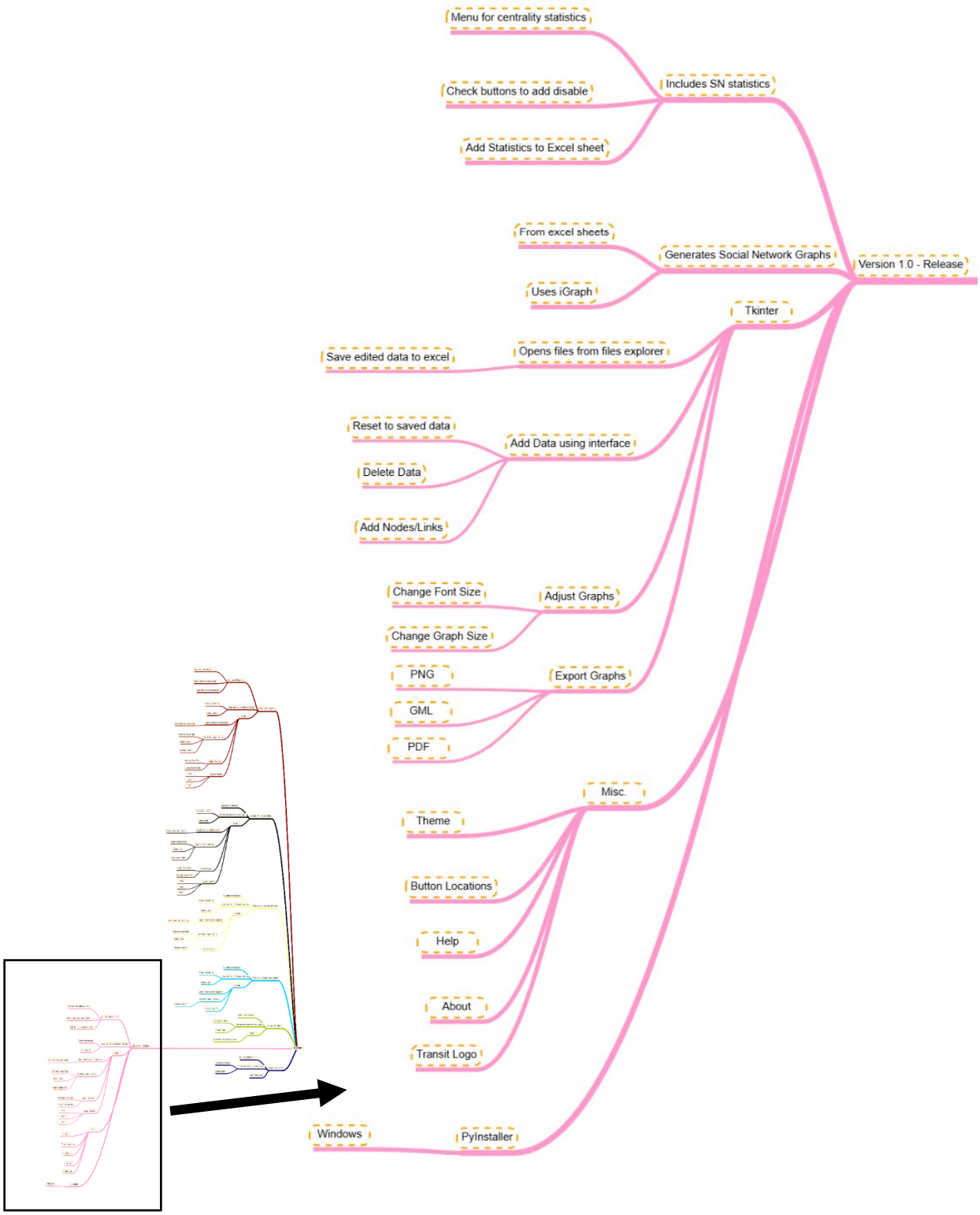

Figure 2: An iteration of the Design Requirements for the AHT, zoomed in on one of the version requirements. Each coloured branch represents a new version. From the mind map it is possible to see how the feature set was expanded for each version. As the system was developed certain features were expanded and some were removed from versions.

The requirements were based on the existing features of the current tool (Jenkins et al., 2007) and the needs of our associated project. The key requirements identified include the need for calculating graph theory statistics, exporting of files, changing the formatting of graphs and saving any edited files back into spreadsheets for use in other software.

During its development, some of the new requirements proved to be unfeasible to implement. The "release" version of the software is what we believe are the essential functions users require to create AH graphs. This should cover a wide range of research use cases and by releasing the code openly, researchers can tweak or build additional functionality for their needs.

The AHT simplifies data editing to create an AH. The software uses an excel sheet as the main data storage. Users can load data from a spreadsheet (if the format includes three columns: phase, id and parent). This allows users to add new parts to the diagram from outside of the application. The AHT's interface includes text-based input elements to allow for users to make changes to their AH within the application. This approach may be slightly slower on very small AH compared to the existing CWA tool (Jenkins et al., 2007) but should be significantly easier to use on larger AH graphs. These can be saved back to an excel sheet, allowing users to retain control of their own data without being locked into a proprietary format. The spreadsheets could be loaded into other software to produce other visualisations or analysis.

To maximise accessibility, we follow the open-source principles in developing the AHT while encouraging wider adoption of HF methods. The source code for the AHT is released on the online repository hosting platform GitHub to allow for researchers with programming experience to change the code to meet the needs of their research projects. As Python is a language which can run on most mainstream consumer hardware and the openness of the software allows future users to resolve any future compatibility issues, the tool should have a long working lifespan. Further, the data formats used within the AHT are chosen to be cross-platform, from importing in spreadsheets to exporting as PDFs.

Using Graph Theory as an Approach

The AHT was developed to produce AH network graphs with integrated graph theory statistics. This can be broken down into the visualisation of the AH and the inclusion of graph theory statistics. The specific nature of the goal allows the developed software to be highly specialised for this usage. The software generates abstraction hierarchy visualisations in the form of network graphs. This presents multiple benefits for HF professionals. Using a network approach (representing the elements and links as Nodes and Edges) enables the ability to calculate centrality statistics. Prior work has used these calculations to understand how different parts of the abstraction hierarchy relate to each other in a quantifiable way (McClymont et al., 2022). The AHT simplifies this to allow easy access to a range of centrality measures and allows these to be appended onto the input file or displayed on the graph itself. Network graphs also feature a range of different visualisation styles, of which two are included (see documentation: https://igraph.org/python/tutorial/0.9.9/tutorial.html) representations of graphs (Figure 3). It should

be noted that none of the included styles fully replicate the traditional AH format but the included styles bring fresh perspective to how the AH is represented.

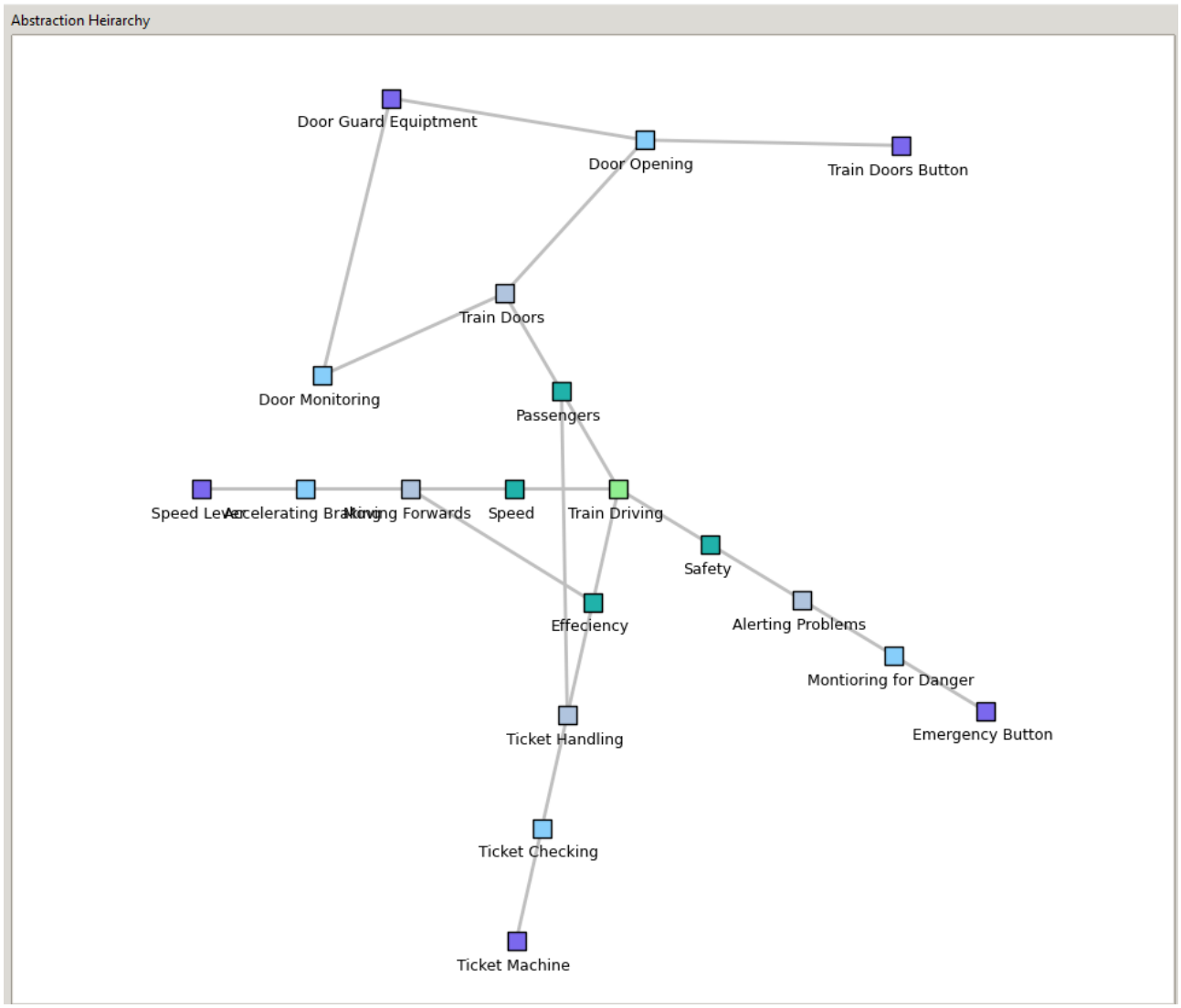

Figure 3: A network layout of an Abstraction Hierarchy using the AHT

Developing the Tool

The AHT was developed in Python 3 (Python Software Foundation, 2022). Python was chosen as it is widely used in academic and data science domains and has the required libraries for creating AH graphs. Users can extend functionality by accessing the source code to customise the software to their needs. The data is loaded into the Python based pandas (McKinney, 2010; Reback et al., 2022) and numpy (Harris et al., 2020) packages and then turned into a social network using the igraph package (Csardi & Nepusz, 2005). The igraph package is available in multiple languages that can be used to create social network graphs and provides all the necessary functions to calculate statistics and generate visualisations. As Python is often run within a terminal it does not have an easy way of making a User Interface (UI). The AHT uses the Tkinter package (https://docs.python.org/3/library/tkinter.html) to provide the UI elements. This includes dialog boxes for opening and saving files and a "window" interface.

Key Takeaways

The Abstraction Hierarchy tool is novel approach to generate AH and its visualisation. The new tool improves on existing software by creating AH graphs which are robust and customisable by the end user. The software also provides an easy access to the graph theory metrics for analysis of AH.

The AHT can provide new opportunities for wider adoption of AH models in communicating how different elements of systems relate to each other. Future research can use the AHT to not only generate production-ready graphs but also to quantitatively analyse the relationship between nodes. Additionally, the tool can support the development of different parts of the Cognitive Work Analysis (as with the prior tool; Jenkins et al., 2007).

Acknowledgements

This work for the TransiT Research Hub https://www.transit.ac.uk/ was supported by the Engineering and Physical Sciences Research Council (EPSRC) Grant Number: EP/Z533221/1

References

Csardi, G., & Nepusz, T. (2005). The Igraph Software Package for Complex Network Research. *InterJournal*, *Complex Systems*, 1695.

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hingu, D. P., Muthukrishnan, S., Rantanen, E. M., & Lintern, G. (2017). A Software Tool for Cognitive Work Analysis. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *61*(1), 212–216. https://doi.org/10.1177/1541931213601537

Holman, M., Walker, G., Lansdown, T., & Hulme, A. (2020). Radical systems thinking and the future role of computational modelling in ergonomics: An exploration of agent-based modelling. *Ergonomics*, *63*(8), 1057–1074. https://doi.org/10.1080/00140139.2019.1694173

Houghton, R. J., Baber, C., Cowton, M., Walker, G. H., & Stanton, N. A. (2008). WESTT (workload, error, situational awareness, time and teamwork): An analytical prototyping system for command and control. *Cognition, Technology & Work*, *10*(3), 199–207. https://doi.org/10.1007/s10111-007-0098-4

Jain, A. (2024, December 12). *2024 UX Tools Survey insights and trends (Future Trends in 2025 UX Design AI and ML)*. UXness. https://www.researchgate.net/publication/389676560_2024_UX_Tools_Survey_insights_and_trends_Future_Trends_in_2025_UX_Design_AI_and_ML

Jenkins, D. P., Stanton, N. A., Salmon, P. M., Walker, G. H., & Rafferty, L. (2010). Using the Decision-Ladder to Add a Formative Element to Naturalistic Decision-Making Research. *International Journal of Human-Computer Interaction*, *26*(2–3), 132–146. https://doi.org/10.1080/10447310903498700

Jenkins, D. P., Stanton, N. A., Salmon, P. M., Walker, G. H., Young, M. S., Whitworth, I., Farmilo, A., & Hone, G. (2007). The Development of a Cognitive Work Analysis Tool. In D. Harris (Ed.), *Engineering Psychology and Cognitive Ergonomics* (Vol. 4562, pp. 504–511). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-73331-7_55

Lind, M. (1999, September). *Making Sense of the Abstraction Hierarchy*. CSAPC'99. https://www.researchgate.net/publication/2390022_Making_Sense_of_the_Abstraction_Hierarchy

McClymont, K., Bedinger, M., Beevers, L., Visser-Quinn, A., & Walker, G. H. (2022). Understanding urban resilience with the urban systems abstraction hierarchy (USAH). *Sustainable Cities and Society*, *80*, 103729. https://doi.org/10.1016/j.scs.2022.103729

McKinney, W. (2010). Data Structures for Statistical Computing in Python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56–61). https://doi.org/10.25080/Majora-92bf1922-00a

Naikar, N., & Sanderson, P. M. (2001). Evaluating Design Proposals for Complex Systems with Work Domain Analysis. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *43*(4), 529–542. https://doi.org/10.1518/001872001775870322

Python Software Foundation. (2022). *Python 3* (Version 3.9.13) [Computer software]. https://www.python.org/downloads/release/python-3913/

Rasmussen, J. (1986). *Information processing and human-machine interaction: An approach to cognitive engineering*. North-Holland.

Reback, J., jbrockmendel, McKinney, W., Bossche, J. V. den, Roeschke, M., Augspurger, T., Hawkins, S., Cloud, P., gfyoung, Hoefler, P., Sinhrks, Klein, A., Petersen, T., Tratner, J., She, C., Ayd, W., Shadrach, R., Naveh, S., Garcia, M., … Li, T. (2022). *pandas-dev/pandas: Pandas 1.4.4* (Version v1.4.4) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.7037953

Walker, G., Salmon, P., Bedinger, M., & Stanton, N. (2016). What the Death Star can tell us about ergonomics methods. *Theoretical Issues in Ergonomics Science*, *17*(4), 402–422. https://doi.org/10.1080/1463922X.2015.1130879