# Hierarchical Task Analysis for Choropleth Map Design Process

>This repository and associated Binder were prepared by G. Aiello, F. Groß, and J. Houghton

## Abstract

----------

This repository contains the files and documentation relating to a Hierarchical Task Analysis (HTA) of a Choropleth mapping activity. The work presented in this readme.me markdown file, supplemented by the discussion of the [Basic Example](https://github.com/james-houghton/ISE521FINALPRJ\_HTA/blob/master/Basic%20Example.ipynb), serves as the author’s final project report for ISE/PSY:521 Human Systems Engineering course at the University of Rhode Island (Spring 2020). A brief introduction to the background of HTA literature is discussed with a focus on its relation to human-computer systems, software development, and unit and integration testing. The HTA is applied to better understand the Task Flow and Information Processing requirements inherent to creating the Choropleth map below. By studying the author's [previous work](https://github.com/james-houghton/ISE521FINALPRJ\_HTA/blob/master/HTA%20for%20making%20Choropleth%20maps%20v0.ipynb) through the lens of Hierarchical Task Analysis, a [Basic Example](https://github.com/james-houghton/ISE521FINALPRJ\_HTA/blob/master/Basic%20Example.ipynb) notebook is created that extends the customization capability of the original Choropleth mapping code. The implementation of the developed in this work is presented as a flow diagram to illustrate task flow and sub goal decomposition [(Stanton, 2006)](#references). A complete process plan with task definitions is created following the procedures defined by [(Annett & Duncan, 1967)](#references). The Basic Example also documents observations pertinent to the future method analysis seeking to broaden the applicability of the Choropleth mapping code.

## Hierarchical Task Analysis for Choropleth Map Design Process via Binder

---

[![Binder](https://mybinder.org/badge\_logo.svg)](https://mybinder.org/v2/gh/james-houghton/ISE521FINALPRJ\_HTA/binder\_version)

Launch Binder Repository using Binder icon above or click \_\_[here](https://mybinder.org/v2/gh/james-houghton/ISE521FINALPRJ\_HTA/7d62ae4bbce4d88e9117e6857e0f7d72450e7e3d?filepath=Basic%20Example.ipynb)\_\_ to be directed to the "Basic Example" found in this repository.

### Choropleth Map Example

>![Alt text](./data/static/test0.png "Choropleth Map for Rhode Island")

## Usage of HTA Maping Process

\_\_\_

- [Hierarchical Task Analysis for Choropleth Map Design Process](#hierarchical-task-analysis-for-choropleth-map-design-process)

\* [Abstract](#abstract)

\* [Hierarchical Task Analysis for Choropleth Map Design Process via Binder](#hierarchical-task-analysis-for-choropleth-map-design-process-via-binder)

+ [Choropleth Map Example](#choropleth-map-example)

\* [Usage of HTA Maping Process](#usage-of-hta-maping-process)

- [Introduction:](#introduction-)

- [Literature Review:](#literature-review-)

+ [Integration Test / Unit Test](#integration-test---unit-test)

\* [HTA Procedure](#hta-procedure)

- [HTA Procedure:](#hta-procedure-)

+ [1. Define Task Under Analysis](#1-define-task-under-analysis)

+ [2. Data Collection Process](#2-data-collection-process)

+ [3. Determine overall Goal of the Task](#3-determine-overall-goal-of-the-task)

+ [4. Sub Goal of the task](#4-sub-goal-of-the-task)

- [Conclusion:](#Conclusion)

- [References](#references)

<small><i><a href='http://ecotrust-canada.github.io/markdown-toc/'>Table of contents generated with markdown-toc</a></i></small>

#### Introduction:

Hierarchical Task Analysis (HTA) was first introduced by Annett and Duncan in 1967

[(Annett & Duncan, 1967)](#references)

and has been further developed since then. It is known to be the best task analysis according to Kirwan & Ainsworth [(1992)](#references). Identification of error variance in system performance from systems theory was a crucial influence in HTA beginning in the 1950's [(Chapanis, 1951)](#references). Annett [(2004)](#references) stated that the top-down systems approach, that HTA uses, enables analysts to identify and deal with factors that generate the largest error variance. Further development of HTA concentrated on the need for greater understanding of cognitive tasks [(Annett, 2004)](#references). At the time HTA was originally developed, existing approaches to structure tasks focused on observable aspects of performance while HTA sought to represent system goals and plans. That was a radical departure from contemporary approaches. HTA offered a way to describe a system in terms of goals and sub-goals, with feedback loops in a nested hierarchy [(Stanton, 2006)](#references).

HTA belongs to a category of task description methods that focus on crucial aspects of the task within the context of the overall task [(Kirwan & Ainsworth, 1992)](#references). The basic hierarchy therefore represents a system without control that is influenced from above (authoritarian) or below (delegatory) [(Shepherd, 2000)](#references). This makes the approach goal oriented instead of action-oriented [(Annett & Stanton, 2000)](#references).

#### Literature Review:

Applications of HTA are not limited by the type operation because of the approach’s variety and adaptability. The method’s success can be brocken down into two key findings. On the one hand, the approach is inherently flexible. Users are able to describe mostly any system. Astley and Stammers [(1987)](#references) state that HTA has been used for decades in order to describe generations of major technological systems. On the other hand, the approach is driven by high applicability. Personal specifications, training requirements, error prediction, team performance assessment and system design are possible areas of application [(Stanton, 2006)](#references). Especially popular applications took place in training design, interface design and job design [(Piso, 1981; Hodgkinson & Crawshaw, 1985; Bruseberg & Shepherd, 1997)](#references). This does not conclude that the application of HTA is easy. Even though there are only a few rules, it requires experience and skills acquired under expert guidance to apply it effectively [(Stanton & Young, 1999)](#references).

Of special relevance in this paper is the application of HTA in software design. Bellotti [(1990)](#references) for example used HTA to model situations within the field of human-computer interaction including the computer interface. Because of previously mentioned characteristics HTA is easy to implement into software design methodologies [(Mills 1998)](#references), which may account for increasing use in the area of system’s design. In Mills [(2007)](#references) a detailed discussion of advantages and disadvantages can be found. The most relevant advantage for software design is stated by Kirwan and Ainsworth [(1992)](#references). It says that HTA offers two distinct training benefits to people engaged in the analysis. First, the analysts can gain insight into processes and procedures that are entailed in plants and organizations. Secondly, training benefits users since they are required to articulate their understanding of the system itself. The point is that by doing the actual HTA the developers acquire deeper understanding of their system which in fact is crucial if the resulting system is supposed to function well.

###### Integration Test / Unit Test

When developing software it is of fundamental importance to test it. The main purpose of testing is to reveal defects. Defect detection capabilities vary with the applied testing methods [(Trautsch, 2020)](#references). Two important testing methods are unit testing and integration testing. The IEEE standard ISO/IEC/IEEE 24765-2010 [(IEEE, 2010)](#references) defines the most important vocabulary for software engineering. It defines a unit as a separately testable element specified in the design of a computer software component. It is a logically separable part that is not subdivided into other components. Unit tests are tests of individual hardware or software units or groups of related items. Integration testing relates to the progressive linking and testing of programs or modules in order to ensure their proper functioning in the complete system. Typically unit tests would be performed prior to testing the overall system with integration tests even though some authors argue about the sequential arrangement of these procedures [(Shahabuddin, 2017)](#references). Some of the given definitions might be outdated due to rapid developments in modern software development contexts [(Trautsch, 2020)](#references). Anyway, an overall understanding of the developed system is required in order to design good tests that allow successful applications. HTA as previously described is a method that makes developers acquire the relevant deep understanding which reveals possible synergies in the development process.

##### HTA Procedure

The first paper to lay out approaches for conducting HTA was Annett et al. [(1971)](#references). It states that the methodology is based upon a theory of human performance. Shepherd [(2000)](#references) added that it can be regarded as a model of human behavior. As a lack of more information about the human operations can be dangerous, it is critical to follow some kind of guidance. There are three main principles governing the analysis [(Annett et al., 1971)](#references):

1. At the highest level of hierarchy a task is considered to be consisting of an operation. This operation is defined in terms of its goal. The goal describes the overall objective of the system in terms of applicable criteria.

2. Operations can be broken down into sub-operations. Each of them is defined by a sub-goal which describes the contribution to the overall system’s goal. It is therefore measurable by performance standards and criteria.

3. The relationship between operations and sub-operations is one of inclusion which means it is a hierarchical relationship.

These principles have not changed over the past years.

A procedure for development of the sub-goal hierarchy with the plans is presented in Stanton [(2006)](#references) in Figure 1. It offers a useful heuristic for breaking the tasks down into a sub-goal hierarchy. This process is of iterative nature and provides a systematic approach for the construction of an HTA structure through repetitive checks. The underlying basic heuristics can be defined as:

- Describing the system goals and sub-goals

- Trying to keep the number of immediate sub-goals under any sub-ordinate goal to a small number (i.e., between 3 and 10)

- Linking goals to sub-goals, and describe the conditions under which sub-goals are triggered

- Stop re-describing the sub-goals when you judge the analysis is fit-for-purpose

>Figure 1

>

>![Alt text](./data/static/sub\_goal\_stanton06.png "Statons procedure for development: Figure 1")

#### HTA Procedure:

The Hierarchical Task Analysis framework is applied to define a generalized task sequence while simultaneously documenting the inherent Task Flow and Information Processing Requirements for a given Choropleth mapping activity. Stanton's [(2006)](#references) procedure for identifying sub-goal hierarchies and process plans is applied at the top level first. Then, assuming that the initial states of systems are under human control(or supervision), a "clairvoyant" approach often referred to as Belady's algorithm is used to create process plans for a specific subtasks.

The test-operate-test-exit (TOTE) procedure [(Miller et al, 1960)](#references) was tested various task sequencing plans at different sublevels to observe the information processing tasks requirements. The purpose of this was to develop an understanding of where opportunities exist for automating modular task functions to increase process efficiency without jeopardizing the generalizability or extensibility. In this vein, process plans created in [previous work](https://github.com/james-houghton/ISE521FINALPRJ\_HTA/blob/master/HTA%20for%20making%20Choropleth%20maps%20v0.ipynb) were considered in the following Hierarchical Task Analysis with the goal of introducing subtasks that allow for limited customization of the Choropleth map above.

###### 1. Define Task Under Analysis

The task is to create a data visualization tool for rendering a Time-Slider Choropleth map for times-series data that are indexable by voter precinct and the geographic location area. The Choropleth map features a color scale that is linearly proportional to a the arrival rate of voters in each precinct during a given interval of time. The existing data visualization tool aims to be extended to include additional customization options. It is also important that the process plan is designed with in a generalized sense such that it is adaptable for future use. As the specific customization requirements, time-series data structures, and geographic indicie schemas are not well documented in the [existing process plans](https://github.com/james-houghton/ISE521FINALPRJ\_HTA/blob/master/HTA%20for%20making%20Choropleth%20maps%20v0.ipynb), there is a need to investigate this further using Hierarchical Task Analysis.

###### 2. Data Collection Process

The primary source of information available at the initial time of the HTA came from the [HTA for making Choropleth maps v0](https://github.com/james-houghton/ISE521FINALPRJ\_HTA/blob/master/HTA%20for%20making%20Choropleth%20maps%20v0.ipynb) file provided by Houghton. This describes a series of task functions that can successfully complete the task at hand but does not specify the order. Moreover, the process steps in the file are labeled using a cumulative count index with all data stored in a fixed initial state. The data are from locally stored files or the [RIGIS website ](https://www.google.com/url?q=https://rhody.webex.com/rhody/j.php?MTID%3Dm731efac0733e19b728ec3d1ded4e70a9&sa=D&ust=1589111926976000&usg=AOvVaw1fGdDe-lPmO73-fmXn44rC) but cannot be directly merged at the initial state of the system. The data collection process began by executing all task sequentially and observing an acceptable outcome for the overall Goal described as follows.

###### 3. Determine overall Goal of the Task

The overall goal is apply the principles of Hierarchical Task Analysis previously discussed to define an alternate set of process plans and functions that successfully accomplish the same goal of creating a [Time-Slider Choropleth Map](#Choropleth-Map-Example) in the initial state defined in [HTA for making Choropleth maps v0](https://github.com/james-houghton/ISE521FINALPRJ\_HTA/blob/master/HTA%20for%20making%20Choropleth%20maps%20v0.ipynb) as well as a set of predefined future states.

###### 4. Sub Goal of the task

The sub goal of the task is to document the outcome of executing the original processing code refactored into different functions/classes. The information observed visually when executing the process is used to inform how the existing Choropleth mapping code can be further developed for broad application. Sub goals of this task are to also define a process plan for selecting a user-defined file paths to geo-data, time-series data, and color scale alternatives.

###### 5. Sub-Goal Decomposition

For the decomposition of sub-goals the earlier mentioned iterative procedure by Stanton [(2006)](#references) has been applied. After having defined the overall goal, subordinate goals have been stated. These mainly cover the choice of geo-data, time-series data, elections-data and a color scheme. After several iterations more sub-goals were added in order to guarantee the systems integrity and functionality. These additional goals cover functional goals such as merging the data set, rendering the map, inspecting and saving the results.

###### 6. Plans Analysis

In order to illustrate the structure of the defined sub-goal decomposition visualization was needed. Flow diagrams are very useful because they easily illustrate the connections and relations between the chosen sub-goals. This method makes it possible to better understand the task hierarchy and allows it to emphasize the task requirements and feasible sequencing options. Obviously the previously achieved results through the sub-goal decomposition have continuously developed and were expanded. This allowed defining the plans that are relevant for the structure of the final HTA. It needs to be noted that the plans analysis was not created in order to reflect the overall complexity. It is considered a helpful tool whose objective is to show basic relations. We created the basic example to understand the Task Hierarchy and Flow in order to gain a better understanding of the task requirements and feasible sequencing options.

>![Alt text](./data/static/HTA\_flowchart.png "Basic HTA Flowchart Example")

7. Stopping Rules

Definitions of stopping rules require advanced understanding of the code and were therefore translated from the previously presented flow diagram. While developing the software some steps needed to be added since they were not obvious when the plans analysis was originally performed. After several adjustments during the process the final structure resulted. The stopping rules are included in the respective plans. They have been implemented through if-loops in combination with HTA’s inherent characteristics. The homescreen for the process refers to the mainpage of the github repository that the project was built in.

> HTA Basic Example Diagram

>![Alt text](./data/static/HTA\_final.png "HTA Basic Example Diagram")

#### Conclusions:

The HTA of the Choropleth mapping activity considered in this work provided deeper insights into the hierarchical structural of task flow and information processing activities. An alternative process plan and task definitions are successfully implemented allowing minimal-customization capabilities to be introduced. The strategic conditional branching of within process plans is designed to allow for recursion to adjust the Choropleth map inputs. Further more, the Python code for sub goals related to precinct data, geodata, and map styling tasks can be modularized thus creating a linear task flow. This is advantageous for future developed of automated task evaluation functions as it allows for independent sub goal decomposition.

Future subtasks accounting for the fact that the codification of precincts and their corresponding geographic indicies do not follow a standardized data structure across the United States can be developed as needed an verified through unit testing methods as previously discussed. Integration testing procedures may also be developed and administered at key steps of the process to ensure repeatability and mitigate the impact of functional interdependencies while of automating future tasks.

\_\_\_\_

#### References:

\_\_\_

Annett, J., Duncan, K.D., Stammers, R.B., Gray, M.J., 1971. Task Analysis. Department of Employment Training Information Paper 6. HMSO, London.

Annett, J., 2004. Hierarchical task analysis. In: Diaper, D., Stanton, N.A. (Eds.), The Handbook of Task Analysis for Human–Computer Interaction. Lawrence Erlbaum Associates, Mahwah, NJ, pp. 67–82.

Annett, J. and Stanton, N.A., 2000. Research and developments in task analysis. In Task Analysis, J. Annett and N.A. Stanton (Eds), pp. 1 – 8 (London: Taylor & Francis, 2000).

Astley, J.A., Stammers, R.B., 1987. Adapting hierarchical task analysis for user–system interface design. In: Wilson, J.R., Corlett, E.N., Manenica, I. (Eds.), New Methods in Applied Ergonomics. Taylor & Francis, London, pp. 175–184.

Bellotti, V.M.E., 1990. A framework for assessing applicability of HCI techniques. In Human-Computer Interaction, Interact 90, D. Diaper, D. Gilmore, G. Cockton and B. Shackel (Eds), (North Holland: Elsevier Science, 1990), pp. 213 – 218.

Bruseberg, A., Shepherd, A., 1997. Job design in integrated mail processing. In: Harris, D. (Ed.), Engineering Psychology and Cognitive Ergonomics. Job Design and Product Design, vol. 2. Ashgate Publishing, Aldershot, Hampshire, pp. 25–32.

Chapanis, A., 1951. Theory and methods for analysing errors in man–machine systems. Ann. NY Acad. Sci. 51 (6), pp. 1179–1203.

Hodgkinson, G.P., Crawshaw, C.M., 1985. Hierarchical task analysis for ergonomics research. An application of the method to the design and evaluation of sound mixing consoles. Appl. Ergon. 16 (4), 289–299.

IEEE, 2010. Systems and software engineering –vocabulary. ISO/IEC/IEEE 24765:2010(E) 1–418. doi: 10.1109/IEEESTD.2010.5733835.

Kirwan, B., Ainsworth, L.K. (Eds.), 1992. A Guide to Task Analysis. Taylor & Francis, London.

Miller, GA, Galanter, E., & Pribram, KA (1960). Plans and the structure of behavior. New York: Holt, Rhinehart, & Winston.

Mills, S., 1998. Information integration – a task orientated approach. Interacting with Computers, 9, 225 – 240.

Mills, S., 2007. Contextualising design: Aspects of using usability context analysis and hierarchical task analysis for software design. Behaviour & Information Technology 26 (6), pp. 499-506.

Piso, E., 1981. Task analysis for process-control tasks: the method of Annett et al. applied. Occup. Psychol. 54, 347–352.

Shahabuddin, S.M., Yalla, P., 2017. Impact of Lean Software Development into Agile Process Model with Integration Testing prior to Unit Testing. Journal of Theoretical and Applied Information Technology 95 (22), pp. 6163-6175.

Shepherd, A., 2000. HTA as a framework for task analysis. In Task Analysis, J. Annett and N.A. Stanton (Eds), pp. 9 – 24 (London: Taylor & Francis, 2000).

Stanton, N., 2006. Hierarchical Task Analysis: Developments, applications, and extensions. Applied Ergonomics 37, pp. 55-79.

Stanton, N.A., Young, M.S., 1999. A Guide to Methodology in Ergonomics. Taylor & Francis, London.

Trautsch, F., Herbold, S., Grabowski, J., 2020. Are unit and integration test definitions still valid for modern Java projects? An empirical study on open-source projects. The Journal of Systems and Software 159.