

TaskArchitect: Taking the Work out of Task Analysis

Jon Stuart

TaskArchitect Inc.
Ottawa, Ontario, Canada
Jon@TaskArchitect.com
+1 613 231 8512

Richard Penn

Positive Interaction Inc.
Ottawa, Ontario, Canada
dpenn@acm.org
+1 613 241 3973

ABSTRACT

This paper takes a pragmatic approach to the design of a task analysis support tool. Instead of proposing a new approach to analysis, it looks at the common requirements for providing support to a wide range of task analysis practitioners, each applying their own style of analysis. The paper describes the range of activities undertaken when practicing what is commonly referred to as “task analysis”. It is proposed that users will only tolerate a level of syntactical complexity in a tool that is sufficient to meet their task analysis needs. Further complexity becomes a barrier to use. Hierarchical Task Analysis (HTA) is selected as the method to be supported because it is a widely used, generic approach that is also the basis of a number of more specialized methods. A commercial tool supporting these requirements is described along with the benefits that may be accrued through its use.

Author Keywords

Task analysis tool, task analysis, hierarchical task analysis

ACM Classification Keywords

H.5.2 User Interfaces, *Training, help, and documentation*

D.2.1 Requirements/Specifications, *Tools*

INTRODUCTION

The breadth of papers on task analysis at TAMODIA and many other conferences illustrates the popularity and success of task analysis as a basic tool for systems design. A large number of different approaches to task analysis have been developed, but far fewer tools are available to help the practitioner use these methods quickly and effectively. This paper examines the needs for a tool to support task analysis and describes a solution that will support a wide range of approaches to the discipline.

© ACM, 2004. This is the author's version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version was published in TAMODIA 2004

<http://doi.acm.org/>

A SURPRISINGLY SUCCESSFUL ANALYTICAL METHOD

Task Analysis has long been used as fundamental step in system design [20]. Diaper [15] described it as “potentially the most powerful method available to those working in HCI and has applications at all stages of system development, from early requirements specification through to final system evaluation”. It provides a concrete representation of the actions taken towards user goals and the logical relationship between those steps. With this foundation, many system design activities can be undertaken, for instance:

- Documentation design,
- Training needs analysis and design,
- Human error prediction,
- Procurement studies,
- Interface design, and
- System architecture design.

Not only has task analysis been successfully applied throughout the design process, it has also been applied across a large range of industries [3].

Each approach to task analysis has its own syntax and method of displaying the results. These were developed to guide the analyst in teasing out the relevant human performance issues for use in design activities. However, they also add to the work to be carried out – the syntax to be maintained through edits; the results to be re-displayed as changes are made. Despite this overhead, task analysis continues to be used by a large number of practitioners.

User Types

Users of task analysis were coarsely grouped into the following categories in order to identify the requirements for a task analysis support tool:

Safety critical system designers – chemical plants, air traffic control, aircraft, military systems

In these cases, a comprehensive task analysis is a requirement of the design process. Large analyses are common and are conducted according to well-defined in-house styles that include detailed descriptions of the tasks. The analysis is used as a basic step of the Human-System Integration, for example training needs assessment. Often

the results are imported into other analysis tools, such as cognitive modeling tools and system simulations. These users are human factors specialists, often working in multidisciplinary teams alongside subject matter experts and simulation specialists.

Human Factors Consultants

This is a more varied environment, in which the analysts apply their favorite approach to requirements gathering and human factors analysis. Projects may require adherence to the customer's in-house style but it is more likely that the end result will be the focus of the work, rather than the method used. Speed of learning of any task analysis tool and adaptation to a range of problems is essential. Analyses usually range from small to medium sized. Users can be described as human factors generalists.

Academic researchers

In this domain analytical methods are highly customized in order to support the research approach in question. The approach used is often closely connected to the researcher's theoretical perspective on tasks, knowledge and cognition [6, 30]. The method of analysis may be complex and result in sophisticated illustrations of relationships between tasks and system features. The tools developed in this area are often highly specific to the research in question [16], and do not necessarily function well across a wide range of applications/domains. Most methods are used by their inventors only [33].

Trainers, technical writers, human resources specialists, business analysts

These users are often the recipients of the output of task analysis in large systems design programs. In smaller design projects they may carry out the analysis themselves. The output is quickly converted into the framework of the final product by the analyst – a syllabus outline, a document outline, or a job description. These users generally do not have formal in-depth knowledge of human performance issues.

Task analysis trainers

Small analyses are created to illustrate the usefulness of the task analysis approach and to teach competence in using this method. As part of small design projects, students may use task analysis to gain an initial understanding of user's activities. The students are often at the undergraduate level, studying human factors or computer science.

User Interface Designers

The majority of the task analysis that is carried out during the design of user interfaces is simple description of the user's tasks as an initial step before design. However, the common use of very structured approaches to design and the use of interface development tools by software developers have led to an increasing interest in tools that move data from task analysis into automated design generation.

User Requirements

Based on the wide variety of applications described above, any tool used to record and structure the results of a task analysis would need to incorporate the following requirements:

- Ability to handle large models,
- Flexibility to adapt to the practitioner's analysis style,
- Ease of recording a wide range of task attributes,
- Flexibility of representation – a variety of means of viewing the task hierarchy,
- Capable of supporting a house style of task descriptions
- Quick to learn,
- Exports data to other tools including spreadsheets, word processors and drawing tools,
- Suitable for users who do not have a formal, in-depth knowledge of human factors.

Costs and benefits of using task analysis

In this section the usability of task analysis is considered in terms of the costs and benefits to the practitioner. Business costs and benefits to the stakeholder brought through its use as part of a human factors project, such as those described by Beevis [10], are not considered in this paper.

The benefits of applying task analysis to system design include:

- It provides a concrete means of summarizing user interaction with the system and user attributes.
- It is a well-known and easily understood stage in the design process, familiar to both human factors practitioners and domain experts within the company. Task analysis can therefore act as an effective communications tool across different parts of the design team.
- The act of carrying out an analysis focused on the user's goals and communicating across the design team brings significant learning about the user into the project team's understanding of the design challenge [9]
- The output of the analysis is re-usable for multiple purposes – it provides a foundation for activities such as training and documentation design and risk analysis.

However, task analysis has the following costs, many of which are mundane tasks, secondary to the actual purpose of the activity – building a foundation for design.

- There is a large overhead in revising the task numbers and plans as tasks are edited and moved within the hierarchy.
- A large effort is required to keep visualizations of the hierarchy synchronized with textual representations of the same information.

- The application of house styles requires that there is frequent editing of the analysis in order to maintain consistency within and across projects.
- The more complex the syntax, the greater the overhead of maintaining it. A complex task analysis method and notation also acts as a barrier to its adoption within industry – both because of the amount of time and effort involved and the barriers it can lead to between the analyst and the receivers of the information – the design team.
- Experience in applying the method is required in order to correctly apply wording conventions and stopping rules, and to create elegant task decompositions.

With the exception of the last of these, all of these costs are relatively mechanical issues: chores that act as barriers to doing the substantive part of the analysis.

To some extent these barriers to the application of task analysis within industry have been overcome by the adoption of a tabular task analysis format. However, this approach does not include a visual representation of the hierarchy and still includes the cost of keeping task numbering and plans up-to-date as tasks are moved around. By using a tabular task analysis format, some researchers have reduced the work involved in the analysis, but in doing so have had to restrict their analysis tool-set, for instance dropping task diagrams as a possible output. While a tabular display is in some respects easier to read, a diagram of the hierarchy makes it easier to identify relationships between tasks [3]. It remains largely a manual approach that is a costly process for large real-world problems.

SOLUTIONS

This section considers a range of possible ways to reduce the effort involved in task analysis. A survey of approaches to task analysis was conducted to find an approach that would not constrain the analyst's problem solving while minimizing the work involved. Existing software tools were reviewed for their ability to support the full breadth of task analysis requirements.

Surveys of Task Analysis Support Tools to date

Two surveys of task analysis support tools to date cover the breadth of current solutions. Bass et al. [7] describe the pros and cons of using software designed to generate outlines, organizational charts and graphics to capture and show the results of task analysis. They also assessed the use of project planning, and spreadsheet software. The problem common to all of these is lack of support for some of the key aspects of HTA – maintaining task numbering, maintaining plans and generating different views of the hierarchy. A task hierarchy can be drawn manually, but since it is not linked to the dataset, it must be manually updated after every change. Without support for the syntax of HTA, each tool requires significant extra work, making impractical the support of larger analyses. Because of their

original intended use, some of these non-HTA specific tools are suited to tabular analyses or drawing hierarchies only. However, no single existing tool combines all of these features, forcing the analyst to spend a lot of time maintaining the analysis or ceasing to use some of the HTA conventions.

The second survey [21] compares a number of tools that support specific approaches to task analysis and simulation. Unfortunately, not all of these tools are available at the current time. Others have costs or learning curves that make casual use prohibitive, or impose highly specific task analysis methods.

Several tools were designed and tested within the academic environment, for example LUTAKD [16]. These demonstrated general benefits that a dedicated tool can bring to task analysis, for instance, the reduction of time and cost that occurs with a tool that can handle large amounts of information easily, a need that was identified quite some time ago [25]. Specific benefits, such as reducing errors through the ability to generate plan constructions [24] were also demonstrated.

Euterpe [35] and CTTE [23] are examples of a more recent generation of tools. These tools are representative of the body of work that has developed definitions of task characteristics [36] task models in order to allow the modeling of tasks. Attributes such as temporality, collaboration and other algorithmic components are built into the model of the task to allow the simulation or stepping through the task model and checking of its logical consistency and completeness. The public availability of the tools has enabled their application within industry. While these tools create the opportunity for designers to check the design implications of a model containing a large number of tasks (which is particularly useful in large safety-critical systems) and are a step closer to interface design specifications, the sophistication of the models can also be a disadvantage.

Some designers found the semantics of temporal operators and multi-agent collaboration difficult to understand [23]. Although these models have been described as 'more expressive' [22] the complexity of the models were found to be difficult to understand by people who were not used to such a structured approach. The detailed, elaborate task description that was a rigorous model of task dynamics became a barrier to understanding what was being represented. Formal models do not by themselves improve communication within a design team or with a client.

Different tools for different user needs

The term "Task Analysis" actually covers of a number of activities, all focussed on task information [11]:

- Collection
- Description
- Organization

- Analysis
- Evaluation
- Modelling
- Simulation

We propose that along this continuum the complexity of notation and concepts required of the tool increases. For data collection an unstructured tool is required for fast and easy recording in the field that will not constrain later problem solving. Description, organization, analysis and evaluation require a toolset that allows the analyst to construct successively more formalized descriptions of the data. Modeling and simulation requires the logical relationships between aspects of the task to be tightly specified within the tool, requiring more complex and rigid syntax as the model develops into computational form.

Users will tend to select a tool that has the best fit along this continuum with the activity that they are undertaking. The pragmatics aspects of the tool use – the robustness, ease of learning, availability and fit with other tools – will also impact this choice. This trade off has been reported in a number of papers [8, 9, 11, 12, 13].

Currently available tools and task analysis methods are aligned with the different task analysis activities in Figure 1. TaskArchitect spans the processes from data collection to evaluation, with exporting to tools used for modeling and simulation.

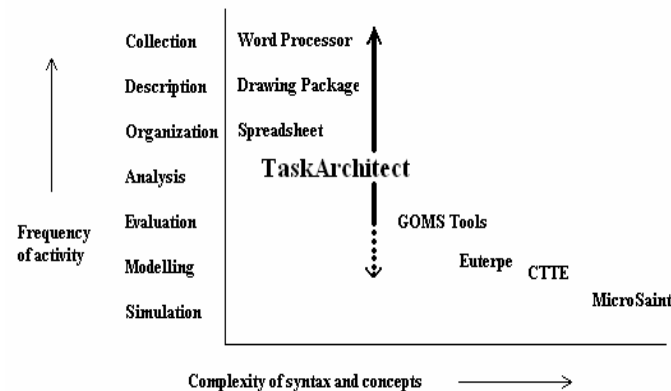


Figure 1 Suitability of tools for task analysis activities

Selection of HTA as the method of task analysis

There may be more than twenty established task analysis methods [6, 16, 18] - ample choice for any analyst and a challenge for the developer of a task analysis support tool. The authors chose to focus their task analysis support tool on Hierarchical Task Analysis (HTA) [3, 4, 28] in part because it is one of the most widely used methods in North America and the United Kingdom.

HTA was developed as a method that can be applied in a wide range of task domains [6], and has been demonstrated to be successful in many projects [20]. Shepherd [27] argues that not only does the decomposition of goals, goal hierarchies and plans provide a general framework for all task analysis (for instance as TAKD [17] and GOMS [14]), it is a sensible step before applying these more specialized tools. In some cases it is a required step [26]. It has been described as “the nearest thing to a universal task analysis technique” [1].

By supporting this basic form of analysis and providing wide ranging opportunities for customization, it is hoped that the support tool developed by the authors will be of use to a large number of analysts. This approach acknowledges the suggestion that “[a] method which to one researcher or practitioner is an invaluable aid to all their work may to another be vague or insubstantial in concept, difficult to use and variable in outcome.” [37, p21]

HTA has the benefits of being readily understandable by a wide audience – people commonly think about their work as a structured hierarchy of tasks [26]. It provides few constraints on the analysis – task statements are written in plain language, plans can also be specified as simple statements without need to resort to a predefined syntax. While this lack of syntax may be said to limit its expressiveness because it does not include the constructs of a more formal modeling language, HTA can also be seen as a more powerful medium for communication in that the analyst is free to express the user’s task without syntactical constraints. [24]

A hierarchical structure is a central component of most approaches to task analysis and task modeling, so a tool based on this approach can act as an initial data collection tool through to analysis and evaluation and then export the results to more complex modeling and simulation tools.

When a tabular approach to HTA is taken, the analyst collects detailed information about each task for later analysis and problem solving. These details are commonly recorded in fields or columns adjacent to the task name. The format allows the collection of virtually all of the attributes discussed in recent papers on task analysis models, for instance role, object, duration and triggering [22, 23]. There are a small number of task attributes that HTA is not as well suited to express, in particular temporality and collaboration. This is because these attributes are logical relationships across items in the task analysis that are most readily recorded through encoding using a programming notation. However, these are also the types of attributes that analysts have found hardest to understand in existing tools [23]. By not attempting to record them in HTA the approach retains its ease of use, while still allowing them to be expressed when the results are exported to a modeling tool.

While the successful conduct of HTA is known to require some experience with how best to construct the task

hierarchy [3], this is true of any task analysis approach. However, because HTA uses many concepts commonly used in standard planning methods, learning to use this method is relatively easy. This also results in a high degree of face validity in the results, which are easy to communicate to all team members. Training in this method is also widely available [6].

While the lack of concrete rules for carrying out HTA may be of cause for concern [29], this approach also enables the analyst to adapt it to their needs - a major usability benefit [29, 31]. By allowing a flexible approach to the analysis, HTA and the tool can be tailored to solve a particular problem as opposed to the rote collection of large amounts of data [2].

Stammers and Shepherd [30] summarize - “HTA is a framework for examining tasks. It is not a dogmatic procedure but a strategy for exploiting a hierarchical description.”

MORE REQUIREMENTS

Further requirements were identified in the two surveys of tools currently available for supporting task analysis. Tools should:

- Support the HTA syntax, including the formation of plans, and support a variety of analytical styles,
- Retain the ease of understanding, lack of formality and flexibility of expression that comes with HTA
- Support both tabular and graphical representations of the analysis,

- Be functionally focused on task analysis rather than more complex modeling applications,
- Be based on a technology that is widely available and easy to update,
- Support a collaborative team approach to information gathering, by recording stopping decisions and tracking revisions, and
- Be designed and tested for deployment within a commercial environment.

THE TOOL

This section describes TaskArchitect, a task analysis tool based on the requirements discussed above. Key features are identified in order to provide the reader with the opportunity to review the extent to which the tool meets a broad range of task analysis methods and applications. For a full description of the tool see www.TaskArchitect.com

A commercial product

TaskArchitect is a released, commercial product – designed for industrial use. It is robust, documented and usability tested. The pragmatics of copying and pasting to other documents, interchange with standard tools such as spreadsheets and drawing packages and export to modeling tools have been addressed. Training packages are available as are support packages. It is designed to be a foundational tool for analysts to build their work on.

Initial input of tasks

The easiest way to record tasks is to simply enter them in a

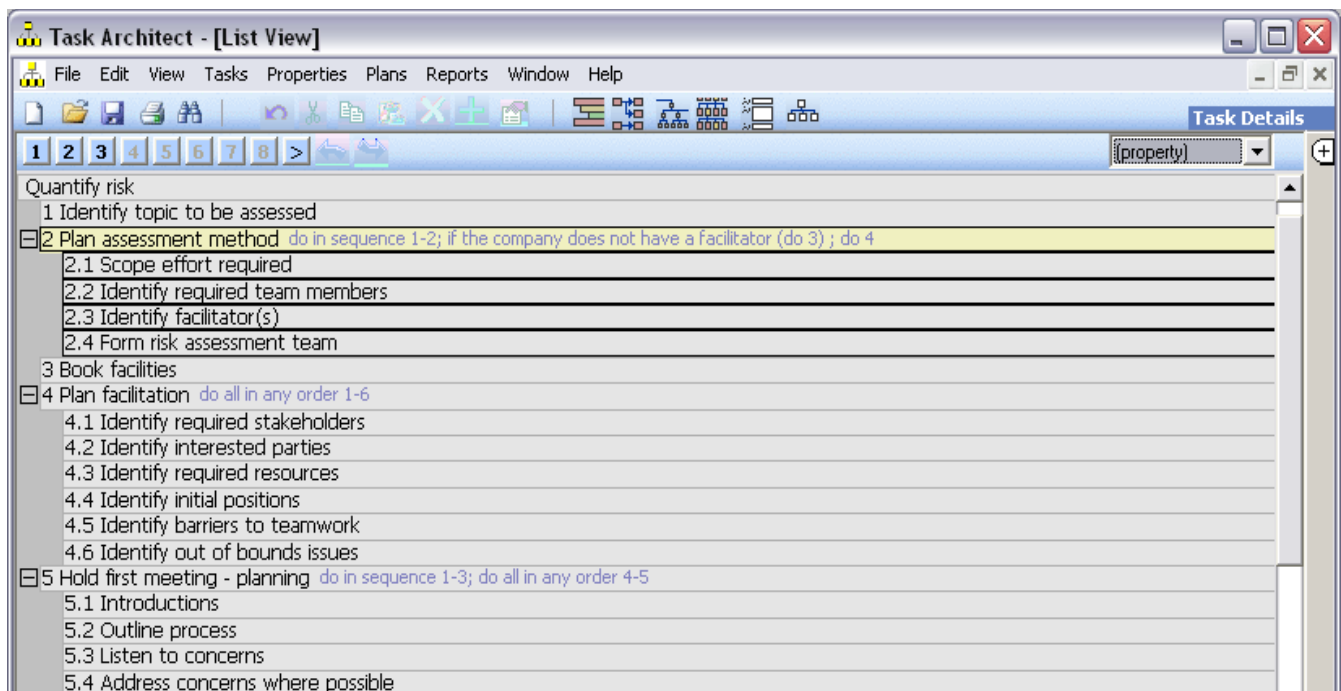


Figure 2 Task List showing indented list of tasks and plans

list editor (the Task List) with the hierarchy indicated by text indents, a method used by most outline editors. This approach matches the style of many analysts – capturing the tasks on the fly as a list then re-arranging them to show the relationships (Figure 2). The tool supports the importing of these types of lists (tab delimited files), making it easy to import existing analyses. Information captured using brainstorming tools can be imported straight into the tool.

Editing the task hierarchy

The tool supports the direct manipulation of tasks in both the Task List and in one of the several graphical views of the hierarchy. Drag-and-drop, multiple selection, and cut-and-paste are all supported according to the normal conventions. All of the task details can be recorded in either of these views.

Plans and numbering automatically updated

Maintaining the HTA syntax is simplified, since the tool automatically updates the numbering of tasks and the contents of plans as tasks are moved within the hierarchy. This saves a considerable amount of time in any analysis. The user can define different numbering conventions for different levels of the hierarchy.

Plans graphically defined

TaskArchitect enables plans to be created by selecting operators and tasks from a dedicated Plan Editor, producing a graphical illustration of the plan using conventional logic symbols. This approach further simplifies the maintenance of the HTA syntax, enabling the user to specify plans without re-typing tasks. This ability to quickly and easily create plans supports the development of complex logical statements, and provides clear feedback on the logic of the plan. The Plan Editor is shown in Figure 3. Initiating and

branching conditions can be specified for each plan. The Plan Editor allows for a wide variety of logical sequences to be applied to the tasks, including:

- Do in sequence,
- Do in any order,
- Do only one,
- Optionally do any,
- Do concurrently, and
- Do not do.

This mechanism supports the development of plans that could later be imported into a performance modeling tool.

A variety of views of the hierarchy

TaskArchitect automatically constructs a variety of views of the hierarchy that can be switched between with a single click. Because the diagrams are laid out automatically, no effort is needed to update them every time a change is made. The wide variety of views supports the analyst in producing outputs tailored to different project objectives [30]. TaskArchitect creates the following task analysis representations:

- Task List: a text based representation. Details of the tasks can be shown here to create a tabular view. The list is indented to represent the hierarchy.
- Left-Right view: a graphical hierarchy with task boxes laid out from left to right across the screen. All aspects of the hierarchy and tasks can be moved and edited in this view as well.
- Overview: this is the traditional HTA diagram, showing the vertical hierarchy of task boxes. The number of

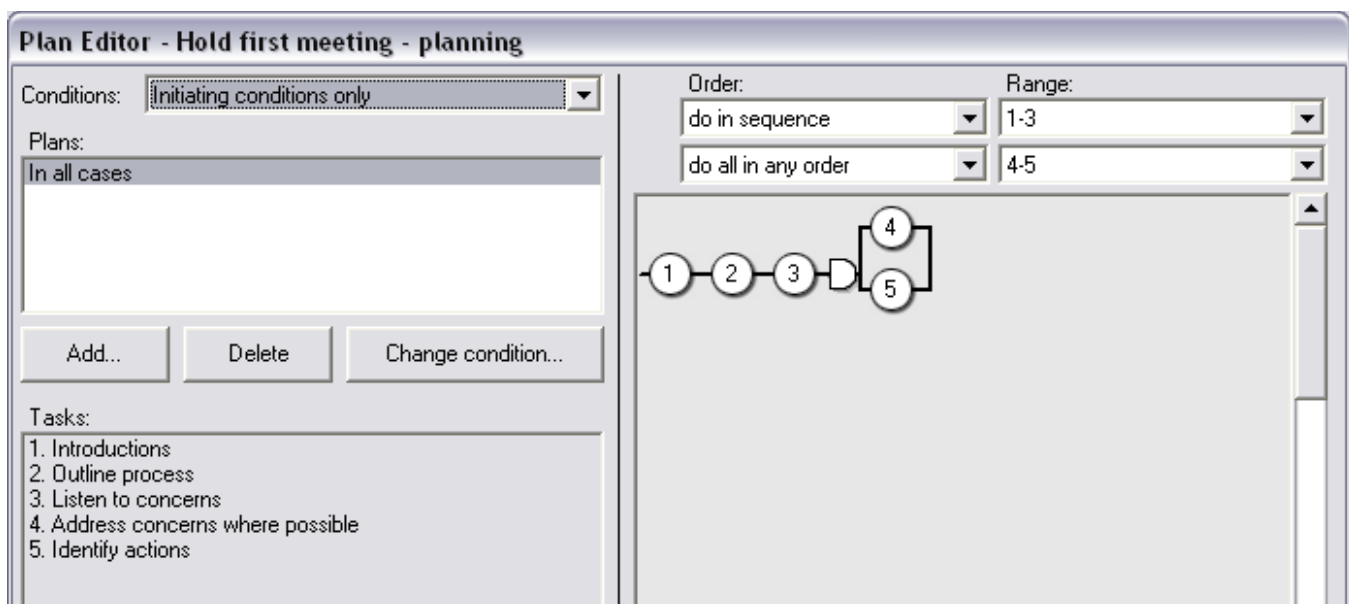


Figure 3 The Plan Editor for point-and-click creation of plans

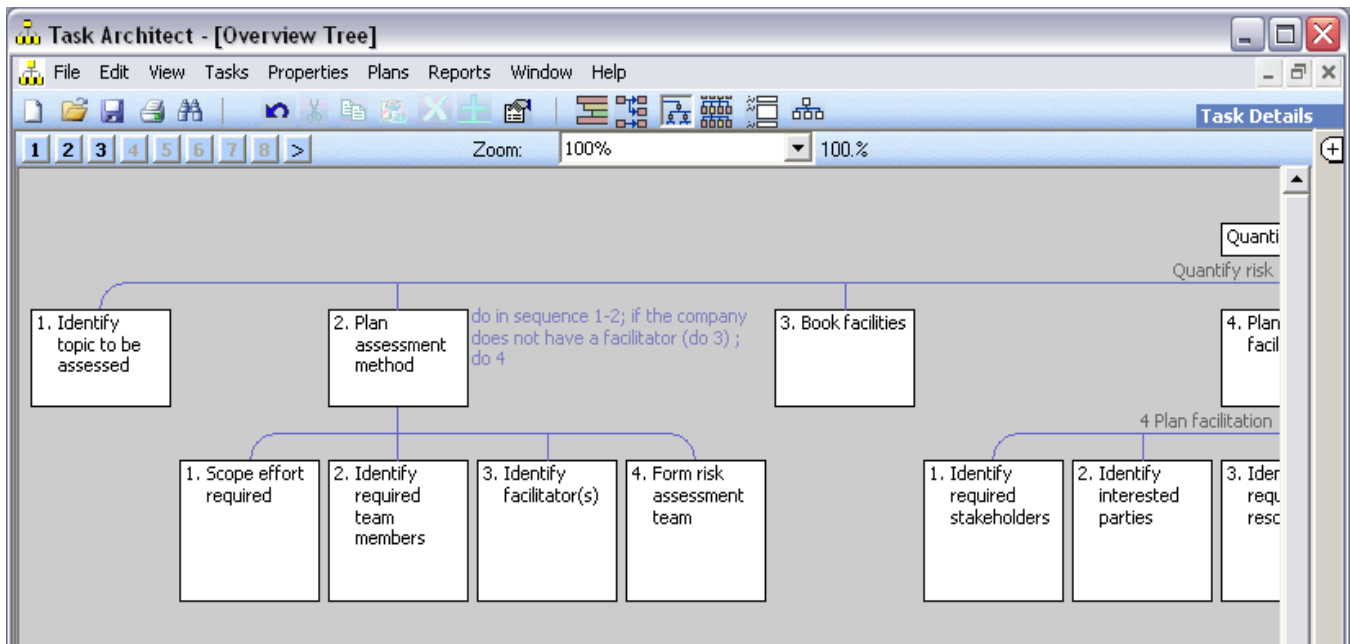


Figure 4 The Overview screen - a task diagram

levels of the hierarchy shown can be restricted in order to make the display more focused and easier to grasp (see Figure 4).

- Vertical slice: a vertical hierarchy showing only the parents, peers and siblings of the task that has been selected. This enables the analyst to focus on just one task and its context, rather than the whole hierarchy.
- Form View: a form of tabular display showing all of the parents, peers, siblings, plans, and detailed information about a single task in one screen.
- Single Task View: shows a single task and its siblings - another way of representing one particular part of the analysis.

A central aspect of the tool is the automatic layout of diagrams so that the user can focus their time on analysis rather than drawing package. However, we also recognize that there may be occasions where the analyst needs to create more one-off representations of the information. For instance, this might involve heavily annotating the diagram, circling key issues etc. TaskArchitect supports this by enabling task diagrams to be exported to Microsoft Office Visio.

Use of task properties to capture task details and support a variety of analysis styles

One of the key features of TaskArchitect is the ability to define properties of tasks. Properties are pieces of information that the analyst wants to record about the task – such as the tool used, the time taken to complete the activity or the type of user experience required. Task Properties can be shown in each view of the tasks; Figure 5 shows task

details in the List View There are different properties types - pre-defined lists of responses (single choice or multiple choice responses), numerical, text, picture type or a reference type. Picture properties enable the analyst to associate a graphics file with a particular task, such as a picture of the tool being used. Task references enable the analyst to point to other tasks in the hierarchy, such as those that are identical, saving the analyst from maintaining identical tasks in different parts of the hierarchy.

This flexibility in supporting task properties enables a variety of task analysis styles and descriptions required for task models to be supported within this single tool. For instance, if properties such as ‘perceptual effort’ and ‘mechanical effort’ are defined, the analyst can set up a cognitive task analysis framework. By defining a required set of properties in a template the analysts can set up a house style or favorite approach to analysis. For instance, the training needs analysis template includes the following properties:

- Task difficulty,
- Task Frequency,
- Task Importance,
- Competencies,
- Performance Measures, and
- Learning Tools.

Supporting a collaborative approach to analysis

Task analysis often requires collaboration. Many tasks are too large for a single person to analyze and manage. In addition, team reviews are a required part of the analytical

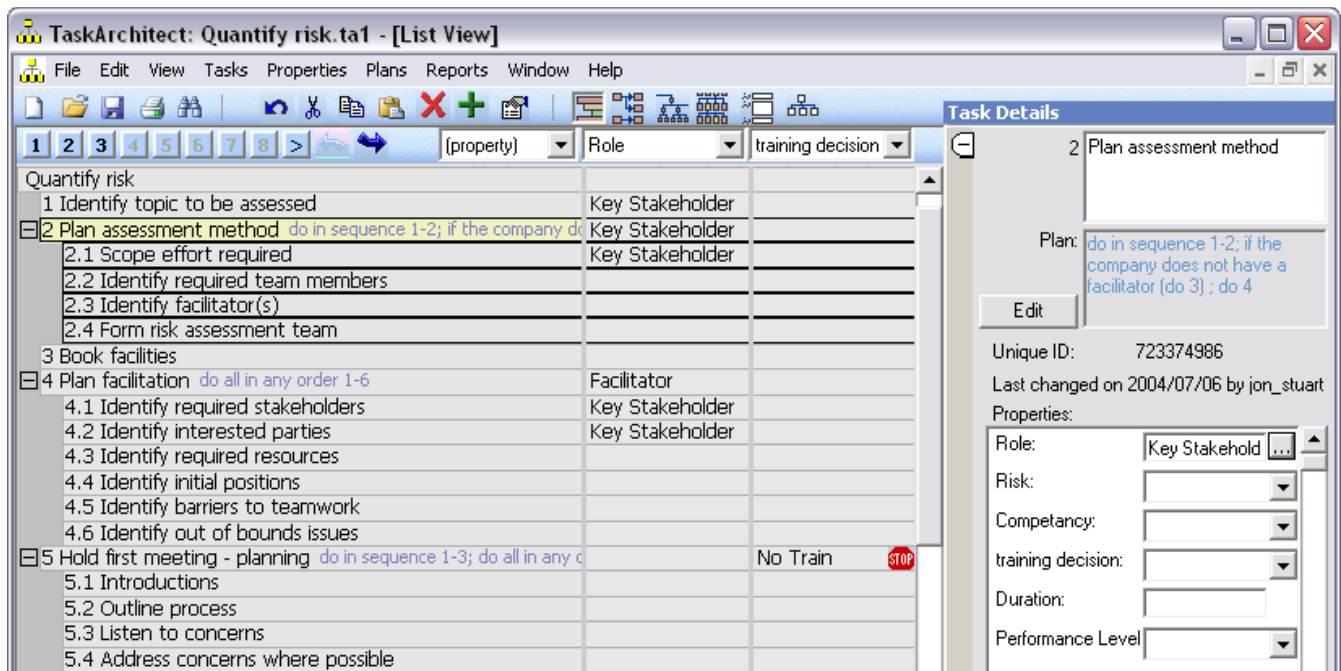


Figure 5 The List view showing task details and plans

process. TaskArchitect supports collaboration by tracking revisions, including the name of the person who last edited a task and the date of the last edit. In addition, standard task analysis features such as marking a part of the analysis as stopped or as completed are supported and shown in task views. Further, tasks can be detached from the current analysis for individual work by a team member then re-attached at a later date. While the tool will support very large analyses (4000 tasks), specific tasks can also be detached to make the display of the hierarchy more manageable.

Using the results of the analysis – Reports, Calculations and Export

In addition to the graphical views, a variety of report formats are provided to enable rapid communication of the analysis to other experts. Analysts can define properties whose values are based on the value of other task properties. For instance weighted sum calculations can be used for analyzing the importance of providing training based on task difficulty, importance and frequency.

The export function enables the analysis to be made available to other programs, in either XML, tab separated or comma separated formats. This allows easy integration with programs such as spreadsheets and word processors as well as more specialized tools such as databases, training needs analysis tools and cognitive modeling tools. This is a key feature of the tool – the flexibility of HTA allows data to be captured and organized quickly, the export function allows the results to be used in more complex modeling applications when required.

CONCLUSION

The creation of a computer-based tool that is specifically designed to support task analysis brings several major benefits [16, 24]:

- It is far faster than manual analysis,
- The support of the HTA syntax prevents many types of error in carrying out the analysis,
- The ease of editing tasks and creating reports supports rapid iteration of the analysis, and
- The removal of much of the overhead of carrying out the analysis makes it practical to carry out large analyses – the viscosity of the method is reduced [19].

TaskArchitect brings further benefits because it addresses a broad range of requirements from the many domains where task analysis is applied and the needs of different types of users, by supporting a generic method of analysis based on HTA.

- Through the use of task properties it enables many different types of information to be recorded about individual tasks in addition to the task name and plan.
- Different analysis approaches can be built on the framework of TaskArchitect by defining properties in accordance with the information requirements of the approach.
- HTA provides a very flexible, easy to use and communicative notation. While it does not include the coding of complex constructs such as temporality into a task model, it does enable the recording of a large amount of task information that can be easily exported into

modeling tools where complex constructs can be added in as required.

- The ease of capturing of data is likely to make the tool more acceptable to designers who find that more formal modeling tools are too difficult for the tasks they want to perform.
- House styles can be quickly implemented through the use of templates.
- The range of task visualizations, reports and export functions that are available, enable the analyst to quickly communicate the results of their work.
- Import from text files, and export to standard office tools, databases and to more specialized tools for training needs analysis and cognitive modeling enables the analyst to easily integrate task analysis into their existing toolset.
- Support for collaboration throughout the analysis process makes it easier to carry out large analyses and work within teams.

While TaskArchitect does not remove the need for experience in applying HTA in order to get the most effective results, it does make applying the technique far easier and speeds the development of this skill. By decreasing the time required to carry out a HTA, it is hoped that this will not only decrease the time pressure on an analyst [32] but will make HTA a tool that can be used in a wider variety of circumstances.

ACKNOWLEDGMENTS

We thank all of the people who provided feedback on early versions of TaskArchitect.

REFERENCES

1. Ainsworth, L. and Marshall, E. (1998) Issues of quality in task analysis: preliminary results from two surveys, *Ergonomics* Vol. 41, No. 11, 1607-1717.
2. Annett, J. (2000) Theoretical and Pragmatic influences on task analysis, In Schraagen, J. M. C, Chipman, S. F. & Shalin V. L., (Eds.), *Cognitive task analysis* Lawrence Erlbaum Associates, Mahwah, NJ, , pp 3–23.
3. Annett, J. (2003) Hierarchical Task Analysis, In Holnagel, E. (2003), *Handbook of Cognitive Task Design*, Chapter 2, pp17-35. Mahwah NJ: Lawrence Erlbaum.
4. Annett, J., and Duncan, K. D. (1967). Task analysis and training design. *Journal of Occupational Psychology*, Vol. 41, 211-221.
5. Annett, J., Duncan, K. D., Stammers, R. B., & Gray, M. J. (1971). *Task analysis*. Department of Employment Training Information Paper No. 6. London, UK: Her Majesty's Stationary Office (HMSO).
6. Annett, J. and Stanton, N. (Eds.) (2000) *Task Analysis*, London, Taylor & Francis.
7. Bass, A., Aspinall, J., Walters, N. & Stanton, N. (1995), *Applied Ergonomics* Vol. 26, No. 2, 147-151.
8. Baumeister, L/K., John, B.E. & Byrne, M.D. (2000) *A Comparison of Tools for Building GOMS models*, Proceedings of the SIGCHI conference on Human factors in computing systems, The Hague, The Netherlands, pp502 - 509
9. Beard, D.V., Smith, D.K. & Denelsbeck, K.M. (1996) QGOMS: A direct-manipulation tool for simple GOMS models. In *Proceedings of ACM conference on Human Factors in Computing Systems (CHI'96)* Vol 2, pp25-26, New York: ACM Press
10. Beevis, D. (2003) Ergonomics – Costs and Benefits Revisited, *Applied Ergonomics* Vol. 34, 491-496
11. Bomsdorf, B. and Szwillus, G. (1999) *Tool support for task-based user interface design*. A CHI'99 workshop. *SIGCHI bulletin*, 31(4), 40-42
12. Bonnie, B.E. and Kieras, D.E. (1996) *The GOMS Family of User Interface Analysis Techniques: Comparison and Contrast*, *ACM Transactions on Computer-Human Interaction (TOCHI)* Vol. 3 , Issue 4 p320 - 351
13. Bonnie, E.J., Prevas, K., Salvucci, D. & Koedinger, K. (2004) *Predictive Human Performance Modeling Made Easy*, Conference on Human Factors in Computing Systems 2004
14. Card, S., Moran, T. & Newell, A. (1983) *The Psychology of Human-Computer Interaction*, Hillsdale, NJ, Erlbaum.
15. Diaper, D. (1989) Task Observation for Human-Computer Interaction. In D. Diaper (Ed.), *Task Analysis for human computer interaction* (pp210-237). Ellis Horwood, Chichester, England.
16. Diaper, D. (2001) Task analysis for knowledge descriptions (TAKD): a requiem for a method, *Behaviour & Information Technology* Vol. 20 No.3, 199-212.
17. Diaper, D. and Johnson, P. (1989) Task analysis for knowledge descriptions: theory and applications in training. In J. Long and A. Whitefield (Eds.) *Cognitive ergonomics and human-computer interaction*. Cambridge University Press, 1989.
18. Diaper, D. & Stanton, N. (Eds.) (2004) *The Handbook of Task Analysis for Human-Computer Interaction*, London: Lawrence Erlbaum Associates
19. Green, T. R. G. (1989) Cognitive dimensions of notations. In A. Sutcliffe and L. Macaulay (Eds.) *People and Computers V*. Cambridge University Press, pp 443-460.
20. Kirwan, B. and Ainsworth, L.K. (Eds.) (1992) *A Guide to Task Analysis*, Taylor and Francis, London.

21. Lee, Y. (2004) Review of the Tools for the Cognitive Task Analysis, *Journal of Educational Technology & Society*, Vol. 7, No. 1, 130-139.
22. Limbourg, Q. and Vanderdonck, J. (2004) *Comparing Task Models for User Interface Design* In D. Diaper (Ed.), *Task Analysis for human computer interaction* (pp.210-237). Ellis Horwood, Chichester, England.
23. Mori, G., Paternò, F. & Santoro, C. (2002) *CTTE: Support for Developing and Analyzing Task Models for Interactive System Design*, *IEEE Transactions on Software Engineering*, Vol. 28, No. 9
24. Ormerod, T.C, Richardson, J., Shepherd, A. (1998) Enhancing the usability of a task analysis method: a notation and environment for requirements specification *Ergonomics* 1998 Vol. 41, No. 11, 1642-1663.
25. Rigney, J.W., Towne, D.M. (1969) Computer techniques for analysing the microstructure of serial-action work in industry, *Human Factors*, 11, 113-122.
26. Sebillotte, S. (1988) *Hierarchical planning as a method for task analysis: The example of office task analysis*, *Behavior and Information Technology*, 7, 275-293
27. Shepherd, A. (1998). HTA as a framework for task analysis, *Ergonomics*, 41(11), 1537-1552.
28. Shepherd, A. (2001), *Hierarchical Task Analysis*, Taylor and Francis, London.
29. Stammers, R.B. (1995) *Factors limiting the development of task analysis* *Ergonomics* Vol. 38, No. 3, 588-594.
30. Stammers, R. & Shepherd, A. (1990), *Task Analysis*, In *Evaluation of Human Work*, Wilson, J. R. & Corlett, E.N., Taylor and Francis, London 1990
31. Stanton, N. and Annett, J. (2000) Future directions for task analysis, In *Task Analysis*, Annett, J. & Stanton, N. (Eds.) London, Taylor & Francis
32. Stanton, N. A. and Barber, C. (1996) Factors affecting the selection of methods and techniques prior to conducting a usability evaluation. In *Usability Evaluation in Industry*, Jordan, P. W., Thomas, B; Weerdmeester, B. A. and McClelland, I. L. (Eds.). Taylor & Francis, London.
33. Stanton, N.A. and Young, M. (1998) Is utility in the eye of the beholder? A study of Ergonomics Methods, *Applied Ergonomics* Vol. 29, No. 1, 21-54
34. Stanton, N.A. and Young, M.S. (1999) *A guide to methodology in ergonomics*, Taylor and Francis, London.
35. van Welie, M., van der Veer, G.C., & Eliëns, A. (1998) *Euterpe - Tool support for analyzing cooperative environments*, *Proceedings of the Ninth European Conference on Cognitive Ergonomics*, Limerick, Ireland
36. van Welie, M., van der Veer, G.C. & Eliëns, A. (1998) *An ontology for task world models*, In *Proceedings of the fifth International Workshop on Design, Specification and Verification of Interactive Systems (DSV-IS '98)* (pp.57-70) vienna: Springer-Verlag
37. Wilson, J. (1995) A framework and context for ergonomics methodology. In *Evaluation of Human Work*, Wilson, J. and Corlett, N. (Eds.), 2nd Edition pp 1-39. Taylor and Francis, London