Joanne Mowat
The Herridge Group Inc.

Cognitive Walkthroughs:

Where they came from, what they have become, and their application to EPSS design

Introduction

Usability testing has long been used in systems development. Over time new permutations of this process have emerged. In this paper we will take a look at traditional usability testing, at variations on that theme, and on one of the variations which is particularly appropriate for application to EPSS projects; cognitive walkthroughs. Cognitive walkthroughs use a detailed review of sequences of actions to evaluate the effectiveness of an interface without formal training. It is a simplified methodology which places usability testing earlier in the design phase when interface problems are still fixable at minimal cost.

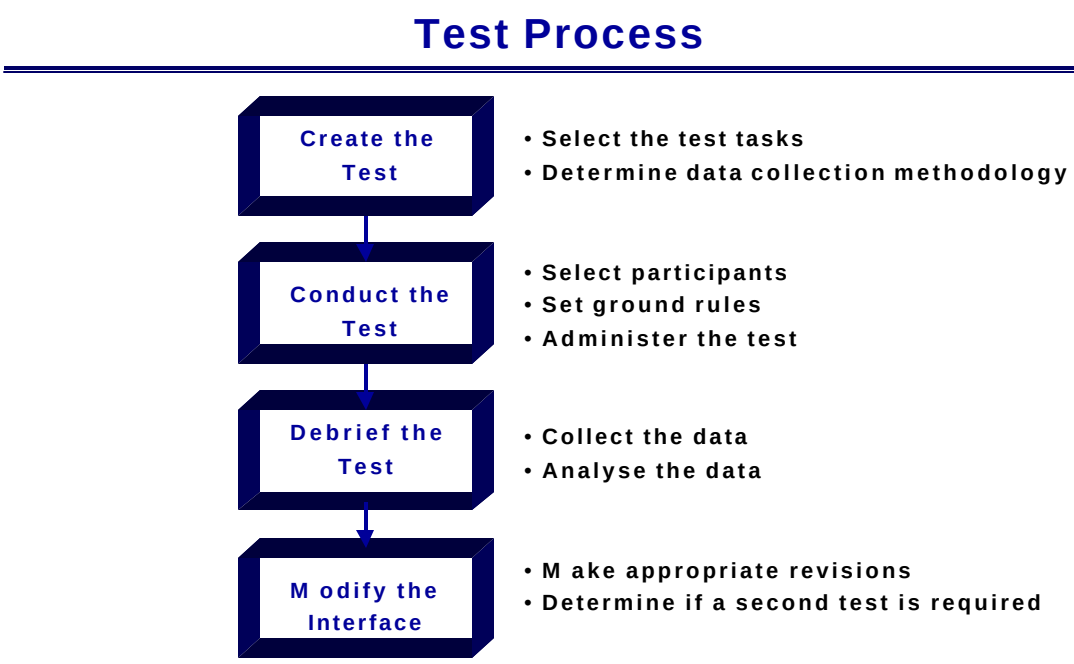
Designing and developing electronic performance support systems (EPSS) is an expensive process in terms of time, money, and resources. To ensure the greatest return on investment we can learn many lessons from the computer system world. Approaches and tools being employed to create today's easy-to-use systems can be applied to EPSS projects.

It must be stated that cognitive walkthroughs are a tool for developing an interface, not for validating it. One should go into a cognitive walkthrough expecting and welcoming changes and improvements. Also, these walkthroughs do not replace good design; nor are they an answer on their own. However, by integrating such techniques or approaches into a solid design and development methodology, significant benefits can be realized.

Usability Testing

Usability testing was developed to allow computer system programmers to study an interface under real-world or controlled conditions in order to observe how well the situated interface supports the user's work interface. Problems with the interface are identified and, hopefully, corrected before implementation. At one time usability testing was regarded as anathema by system designers and used only when they were faced with large problems. New technologies including joint requirements planning (JRP) sessions, joint application design (JAD) sessions, CASE tools, structured design and programming, information modeling, and prototyping have completely changed the way systems are designed and have brought an increased desire on the part of the systems design team to perform usability testing (Yourdon, 1989). This coupled with a more customer-centered design approach, based on a recognition of who is paying for the product, has made quality essential to the systems design process.

Diagram #1



Usability testing has traditionally been carried out once the system has reached an alpha test state. This means that the design and coding are complete and bugs are being identified and corrected. It is intended as a pre-pilot test employing a small but representative sample of end users to ensure that:

- End users can intuitively and self-sufficiently use the system to meet business needs and objectives.
- There are no hidden points of confusion.
- Training time is minimal

The benefits of usability testing include:

- Improved quality
- Minimization of post-pilot revisions (every $1.00 spent in usability testing avoids $2.00 in later costs).
- Ensuring that you can deliver something the end users want and can use. (Villachica & Moore, 1997)

However, since traditional usability testing is held after coding is completed, problems raised at this point are often deemed too costly to fix. Thus a training problem is identified and system intuitiveness is redefined as the users being able to use the system after training. The cost of building a user interface that is complete enough to reveal major problems, testing it, then making changes to it is unacceptably high. (Lewis & Rieman, 1994)

The following are some of the problems related to traditional usability testing:

- The people with the experience to structure and facilitate usability testing are scarce and in high demand.
- Usability testing requires the participation of real end users whose time is neither free nor an unlimited resource.
- The design needs to be relatively bug-free before using valuable user time
- A tendency to focus on task-based behaviours rather than cognitive models, perpetuates the myth that a compliant interface is enough.

Despite the problems with the traditional model, the reasons for doing usability testing are still compelling. The challenge is to take a structured approach to discovering the strengths and weakness of an interface; before building it. To minimize or overcome the drawbacks to usability testing, Villachica & Moore (1997) recommend:

- Holding the usability test as early as possible in the design phase and planning for it accordingly.
- Using only real end-users.
- Keeping the data collection simple, unobtrusive, and as efficient as possible.
- Getting users to verbalize actions.
- Focussing on: the errors the users make; and, broad areas of consensus.

Getting the users involved as early as possible in the design is critical. However, there is also a need to evaluate the evolving design without users so that, before they are involved, the major and obvious problems have been identified and corrected. This not only respects user time restrictions but also helps maintain credibility. Users will quickly come to distrust a system that is seriously and obviously flawed and to distrust the design team that created it.

Several approaches are being used by different groups to conduct usability testing. Individual walkthroughs, team walkthroughs, empirical testing, simulation models, and structured walkthroughs (to name but a few) are all in

regular use today. Team walkthroughs have been shown to be more costly but also more effective than individual ones (Karat, Campbell & Fiegel, 1992). The use of scenarios rather than individual free form testing has also proven more efficient. Empirical testing takes half the time to identify the same problems but requires that, at a minimum, a fully functioning prototype with a supporting database be in place.

Structured walkthroughs focus on the completeness, integrity, correctness, and quality of a user interface but are still process and task focussed. They deal with whether the system interface accomplishes what it was designed for rather than whether or not the design supports the work.

Cognitive Walkthroughs

The cognitive walkthrough method of usability testing combines software walkthroughs with cognitive models of learning by exploration. It is a theoretically structured evaluation process in the form of a set of questions that focus the designers' attention on individual aspects of an interface and that make explicit important design decisions made in creating the interface and the implications of these decisions for the problem-solving process. This methodology stresses that usability testing should take place as early as possible in the design phase, optimally in conjunction with early prototypes. This allows

for the evaluation of early mock-ups quickly and supports developers in the upstream activities of identifying and refining requirements and specifications.

The methodology is different from other forms of usability testing in that:

- it does not need an executable simulation of the interface. Level 1 prototypes of the interface (often screen diagrams with no functionality) are used. This means that cognitive walkthroughs can be held early in design while it is still cheap and easy to revise the interface.
- the focus is on the mental operations of the user rather than on the characteristics of the interface per se.
- problems are identified in relation to specific, key, representative user tasks.
- it links the users' cognitive processes directly to the features of the interface.
- it is derived from a theoretical model of mental processes rather than from a system design methodology. (Polson, Lewis, Rieman & Wharton, 1992)

This methodology has been evolving since its introduction in 1990 (John & Packer, 1995). Developers of an interface walk through level 1 prototypes of the interface in the context of core tasks a typical user will need to accomplish. Actions and feedback of the interface are compared to user goals and knowledge. Discrepancies between user expectations and the steps required by the interface are noted. However, since system development is still anchored,

for the most part, in data-centric information systems groups continuing assumptions such as:

- People who know the work will use the software
- Training and on-going support are inevitable
- The future is simply an extension of the past (Gery, 1997)

interfere with the identification of design flaws that can cause the system to fail to support performance-centered requirements and which do not embed business and software learning resources in the system. These assumptions are particularly dangerous when using this technique to evaluate an EPSS interface.

Cognitive walkthroughs are based on the theory of exploratory learning. A list of theoretically motivated questions about the user interface are generated with focus on the interaction between the design and the user attempting to perform a specific task. Positive responses to the individual questions support the inference that the interface will be easily learned. Negative responses highlight those steps in an operating procedure that may be difficult to learn. (Lewis, Polson, Wharton, Rieman, 1990)

There is some concern that cognitive walkthroughs may identify fewer real flaws in the interface than issues related to the test itself. They may result in recommendations that are sub-optimal or even erroneous. Some feel it is fair at best in finding recurring general problems and studies have shown that cognitive

walkthroughs may consistently detect only 50% of the problems. (Jeffries, Melba, Wharton, Uyeda, 1991) There is a real need to carefully define what tasks are included in the walkthrough, and to ensure that they are representative and sufficient to test all major interactions. The tasks should exercise key system functionality. This may mean selecting tasks that comprise multiple core functions of the interface. This will allow for the evaluation of the elements, their combination, and any necessary transitions among the sub-tasks. (Wharton, Bradford, Jeffries & Franzke, 1992) The degree of realism / complexity must be traded-off against the number of tasks that can be covered in the walkthrough, assuming fixed and limited resources. Some questions which arise when planning the walkthrough are: where should the boundaries of the walkthrough be drawn, how many tasks are enough, and at what level of granularity should be evaluation be carried out.

The cognitive walkthrough process, as it has evolved, flows as follows:

1. The author of the design presents the proposed design to a group of peers.

2. The peers evaluate the design using an explicit set of criteria appropriate to the particular class of design issues in order to evaluate the ease with which users can perform tasks with little or no formal instruction. These criteria focus on the cognitive process needed to perform the tasks with the system as designed.

- Inputs to this part of the process include:
  - A detailed design description of the user interface.
  - A task scenario (or scenarios for several tasks).
  - An outline of the explicit assumptions about the user population and the context of the system use.
  - The sequence of actions which a user could use to successfully complete each task.

3. The peer reviewers step through the actions while considering the behaviour of the interface and its effect on the users. They attempt to identify actions that would be difficult for an average member of the target population. Any claims that a given step will not cause problems to the target population must be supported by theoretical arguments, empirical data, or relevant experience and the common sense of the team members. Claims that any step will cause difficulty to the target population must be similarly supported. (Polson, Lewis, Rieman & Wharton, 1992)

   The reviewers need to answer four questions about each step they take:

   - Will the users be trying to produce whatever effect the action has?
   - Will the users be able to notice that the correct action is available? This relates to whether the button, menu, etc. is visible.

- Once users find the correct action will they know that it is the right one for the effect they are trying to produce? This relates to whether the labeling, icons, etc. are clear and logical to the users.
- Will the users be able to understand the feedback they get from the system? Will they know they have taken a correct or incorrect action?

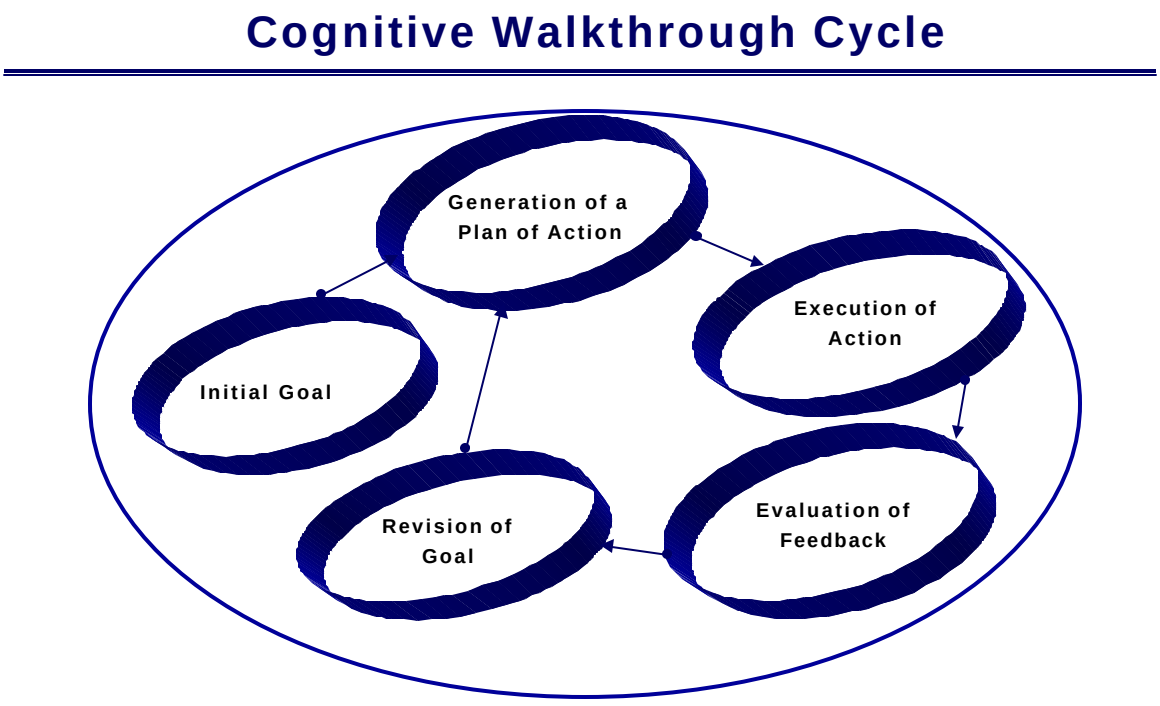
4. The reviewers evaluate whether the users will be able to formulate a new goal for the next action or detect that the task is complete based on the system response. Will they be able to select each of the correct actions along the solution path.

Some basic design features help determine whether an action will be apparent to a first-time user. These include: a user will try label-guided actions first (menu items, buttons, etc.); a well-labeled action will be especially salient; if labeling cannot be provided, ensure there are a limited number of options in the search set; set effects may prevent users from trying untypical actions; and, users are reluctant to extend their search beyond readily available menus and controls. (Rieman, Franzke & Redmiles, 1994)

The process rests on a foundation of a theory of learning by exploration proposed by Polson and Lewis in 1990 which purports that users will guess what to do by using the cues provided to them by the system. The user's initial goal

leads to the generation of a plan of action. The action is executed, feedback is evaluated, the goals are revised, and the cycle continues.

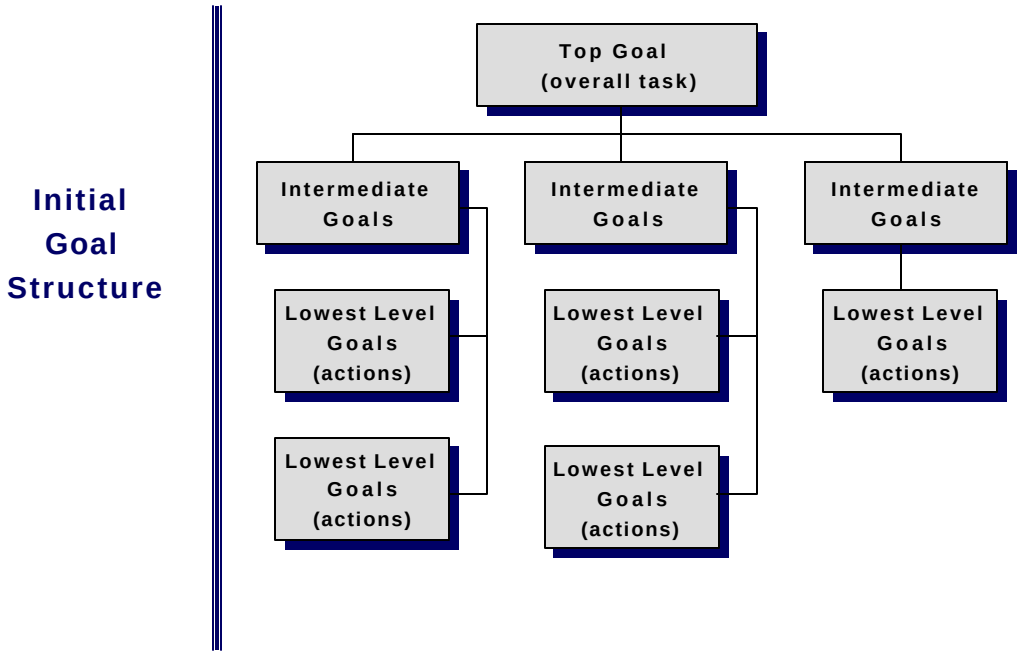
Diagram #2



The methodology also includes an aspect of constructionism in that it assumes a process by which users integrate the new text, data, etc. with their background knowledge and construct the action plan that will enable them to accomplish their goal. (Polson, Lewis, Rieman & Wharton, 1992). Goals are represented by propositions which are linked to other goals, linked to their background knowledge, and linked to the objects seen in the interface. Action flows from the

top goal along the links to the action. Once the action has been performed, any response by the system is noted and interpreted. This leads to the deactivation of the initial goal and the building of new goals.

Diagram #3



For any action to happen, there has to be a clear link or path of associated connections between the goal and the action. In other words, during the cognitive walkthrough the evaluators must be able to clearly show that it is reasonable to assume that the users will make the connections and construct the knowledge in a way that will result in the action required. For example, if there is a button with the word Print below it or a picture of a printer on it, it is

reasonable to assume that a particular target population would make the connection that they can use that button to print. If the button is poorly labelled or diagramed, or if it is not easy to find on the screen, then it is reasonable to assume that the users may not necessarily make the connection and so will be unable to fulfill their goal.

So, the cognitive walkthrough can be seen as a specified procedure for simulating a user's cognitive processes as the user interacts with the interface. The value of this procedure, as mentioned earlier, depends on choosing the correct set of tasks to be evaluated. These tasks must be things users will actually want to do and should include tasks made up of a sequence of basic tasks in order to evaluate the links between these tasks. Tasks must be described from the point of view of a first time user and must be described in realistic terms, including any specific assumptions on the state of the system.

Problems identified by cognitive walkthroughs may relate to goal problems: the user is trying to do the wrong thing; or, action problems: the user is trying to do the appropriate thing but cannot figure out how to do it. These problems can be addressed by providing training to overcome the system flaws, supplying prompting information as the user moves through the interface, or most effectively, changing the system so training and prompting are not required.

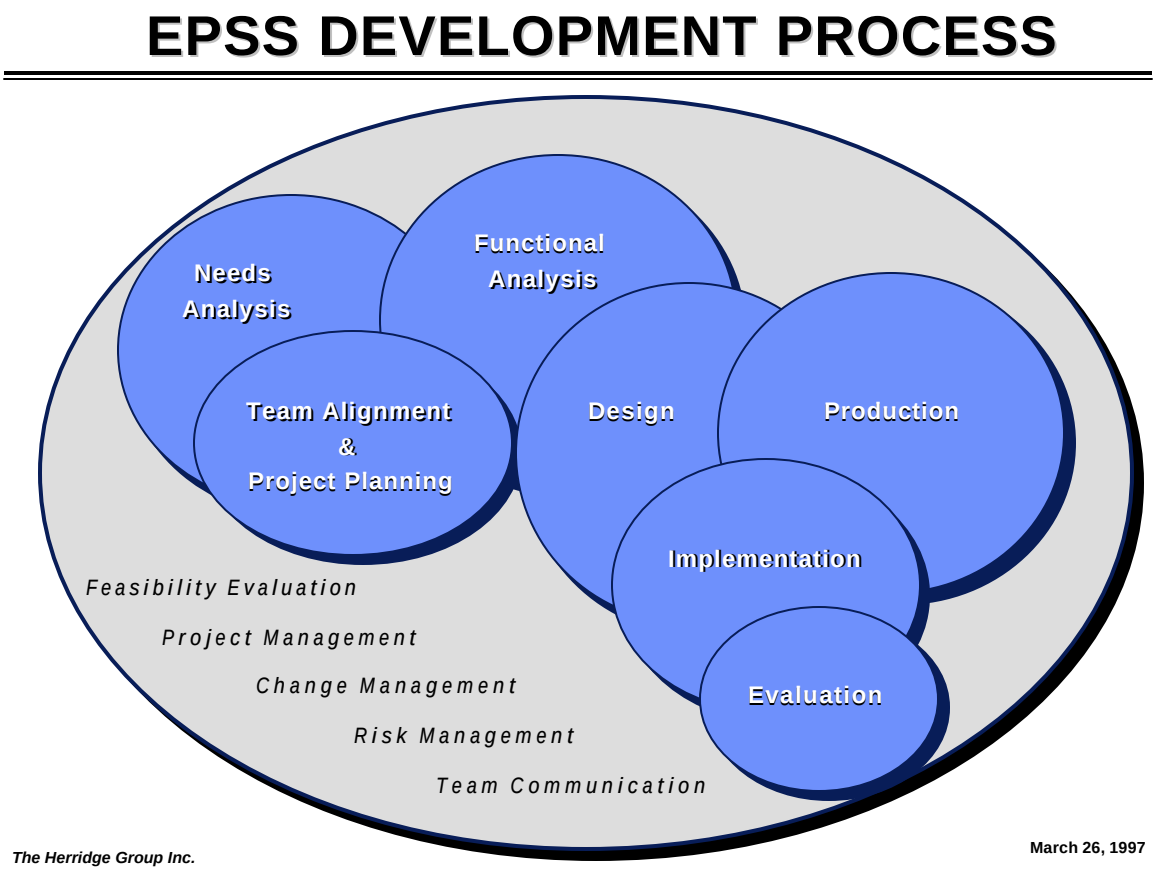
Applying Cognitive Walkthroughs to the Design and Development of Electronic Performance Support Systems

The aim of cognitive walkthroughs is to produce systems that are easy to use and learn since there are many situations where users will want to make use of a system without having to invest much time or take training. Here is where an obvious dove-tail between the evaluation method and the purpose of the EPSS. When cognitive walkthroughs are applied to EPSS projects, they can identify not only the system interface problems, but also issues and concerns surrounding the integrated support features. At times, by eliminating the interface problem, the need for the support feature is eliminated.

To get the greatest benefit from cognitive walkthroughs in EPSS designs, they should be used in conjunction with joint application design sessions, iterative prototyping, and strong project management. If an EPSS design and development process like the one shown in Diagram #4 is used, cognitive walkthroughs could be incorporated in several phases. The process shown is an iterative rather than linear one in which the phases are heavily overlapped and in which formative evaluation is constant; largely handled through prototyping. In this process, the first cognitive walkthrough would be held during analysis, as soon as the first prototype is developed. (See Diagram #5) By doing several walkthroughs, linked with the iterations of the prototypes, problems will be identified early and corrected immediately. If the walkthrough can be held before the prototype is shown to the end users, valuable end user time will be

conserved and the credibility of the design team preserved since the users will be enjoying an improved product.

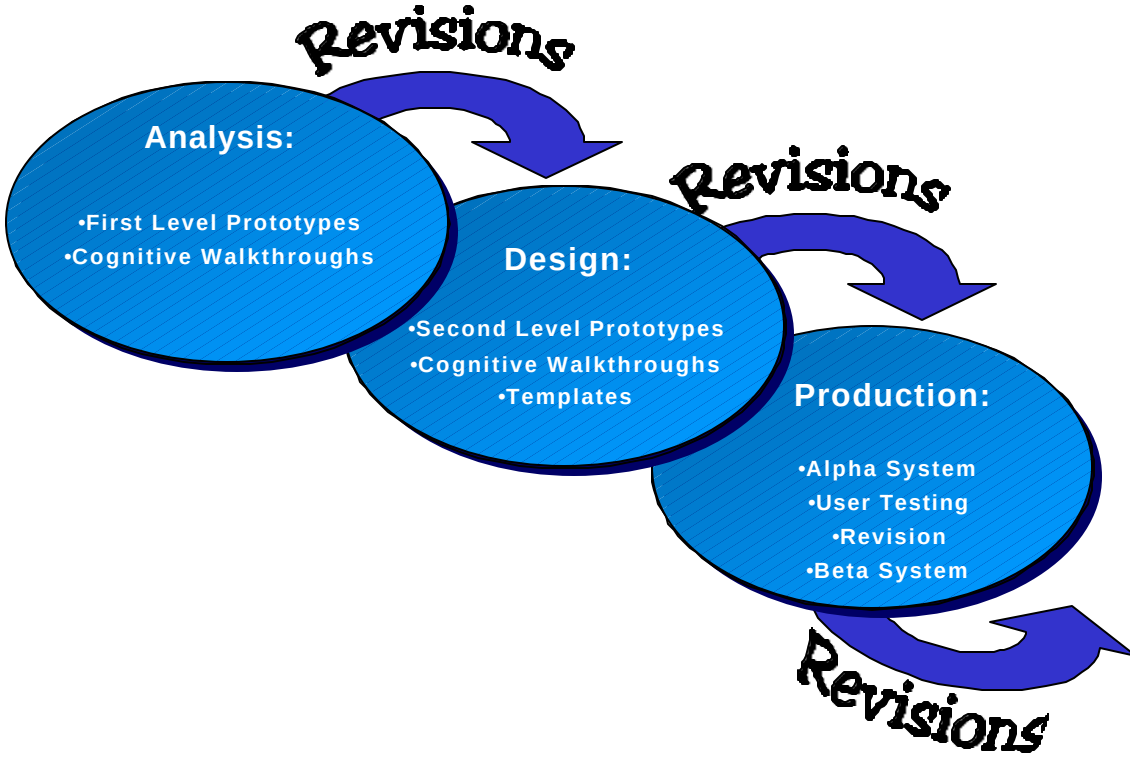
Diagram #4



Cognitive walkthroughs are carried out by the designer and the designer's peers and as such cannot replace user testing. As well as reviewing and critiquing the prototypes, real users must test the EPSS before it is implemented. What walkthroughs do accomplish is a reduction in the number of revisions that result from the user testing of the alpha product. If each type of interaction in each component of the EPSS has been prototyped and if each set of prototypes has

been through a cognitive walkthrough process, the templates used in production will have been effective and accurate thus minimizing the changes coming out of user testing.

Diagram #5



In the best case scenario, the EPSS is being developed as a completely integrated part of the computer system or job process it is intended to support. Cognitive walkthroughs will serve, in these cases, to point out issues related to system design or work process that can be corrected, thereby eliminating or greatly reducing the amount and depth of support required. If an EPSS is being built to support an already existing system or job process, the walkthroughs will ensure that relevant and efficient support will be provided.

Even if, as mentioned by Jeffries, Melba, Wharton, Uyeda in 1991, cognitive walkthroughs detect only 50% of the problems, doing several walkthroughs on the iterations of prototypes and then having the revised prototypes reviewed by users will ensure that the bulk of the problems will be identified and corrected while it is still cost effective to do so.

Earlier in this paper, concerns were raised about the data-centric nature of cognitive walkthroughs due largely to their being controlled by the systems development groups. By moving control of this methodology over into the instructional designers bailiwick, performance-centered requirements can be defined and designed for. Business and learning resources can be embedded in the system or the support tool to help ensure optimal performance as close as possible to day one.

Conclusion

Many lessons can be and have been learned from system design methodologies that can be directly applied to the instructional design of both paper-based and computer-mediated performance and learning interventions. While borrowing and re-purposing the approaches and tools we should exploit opportunities to combine and refine them so that they can be embedded in our instructional design processes in such a way as to reap the greatest benefit. Just as the system methodologies borrowed from cognitive learning models in developing the cognitive walkthrough approach to usability testing, we can borrow from the time and resource saving system approaches to develop new and better ways to develop and support performance.

Bibliography

1. Franzke, M. (1995). Turning Research into Practice: Characteristics of Display-Based Interaction. Proceedings CHIU95. *http://www.acm.org/sigchi/chi95/Electronic/documnts /papers/mf_bdy.html.*

2. Gery, G. (1997). Performance Support: Performance Centered Design. *ISPI 1997 Conference Notes*. Anaheim, California.

3. Jeffries, R., Miller, J.R., Wharton, C., & Uyeda, K.M. (1991). User Interface Evaluation in the Real World: A Comparison of Four Techniques. *Proceedings CHIU91*. New Orleans, Louisiana, ACM, NY. pp. 119-124.

4. John, B.E., & Packer, H. (1995). Learning and Using the Cognitive Walkthrough Method: A Case Study Approach. Proceedings CHIU95. *http://www.acm.org/sigchi/chi95/Electronic/documnts /papers/bej1bdy.html.*

5. Karat, C.M., Campbell, R. & Fiegel, T. (1992). Comparison of Empirical Testing and Walkthrough Methods in User Interface Evaluation. *Proceedings CHIU92*. Monterey, CA, ACM, NY. pp. 397-404.

6. Lewis, C., Polson, P., Wharton, C. & Rieman, J. (1990). Testing a Walkthrough Methodology for Theory-Based Design of Walk-Up-and-Use Interfaces. *Proceedings CHIU90*. Seattle, Washington, ACM, NY. pp. 235-242.

7. Lewis, C., & Rieman, J. (1994). Task-Centered User Interface Design A Practical Introduction. *http://www.cs.ut.ee/~jaanus/hcibook/chap-1.v-1.*

8. Polson, P.G., Lewis, C., Rieman, J., & Wharton, C. (1992). Cognitive walkthroughs: A method for theory-based evaluation of user interfaces. *International Journal of Man-Machine Studies 36,* p.p. 741-773.

9. Rieman, J., Franzke, M., & Redmiles, D. (1994). Usability Evaluation with the Cognitive Walkthrough. Proceedings CHIU95. *http://www.acm.org/sigchi/chi95/Electronic/documnts /tutors/jr_bdy.html.*

10. Rowley, D.E. & Rhoades, D.G. (1992). The Cognitive Jogthrough: A Fast-Paced User Interface Evaluation Procedure. Proceedings CHIU92. Monterey, CA, ACM, NY. pp. 389-395.

11. Spaulding, V., Wilson, T., Najarian, A., & Stephenson D. (1996). Traveler's Aide. *http://c2000.gatechedu/c2000/cs6751_96_fall_demo/projects /aida/part2.html#prototype.*

12. Wharton, C., Bradford, J., Jeffries, R., & Franzke, M., (1992). Applying cognitive walkthroughs to more complex user interfaces: Experiences, issues, and recommendations. *Proceedings CHIU92*. Monterey, CA, ACM, NY. pp. 381-388.

13. Yourdon, E. (1989). *Structured Walk-Throughs*. Englewood Cliffs, NJ: Prentice Hall.

14. Villachica, S. & Moore, A. (1997). You Want it When? *ISPI 1997 Conference Notes.* Anaheim, California.