

Question 1	<p>a) In the Proceedings of UIST '99, Hudson and John presented a tool for automatically assigning KLM operators to a GOMS decomposition of a 2-dimensional GUI. Explain what you consider to be the pros and cons of such a design tool if it were extended beyond the initial work of Hudson and John and wrapped into a commercial interactive development environment.</p> <p>b) More generally, explain why the GOMS family of modeling and evaluation techniques have not been used extensively in the practice of HCI design.</p> <p>c) Briefly explain another model of human cognition and how it could be used to describe human behavior with an interactive application.</p>
------------	--

Part a)

NOTE: I am assuming that “if it were extended beyond the initial work” means that if such a design tool as CRITIQUE were to automate the full power of GOMS modeling (one of Hudson and John’s future goals) and does not include adding support of other evaluation techniques.

In the Proceedings of UIST '99, Hudson and John presented CRITIQUE, a tool for automatically assigning KLM operators to a GOMS decomposition of a 2-D GUI. If this work were extended to support more automatic goal hierarchy, method, and selection rule creation...there would be both pros and cons if it were integrated into a commercial IDE.

In their exchange with Newell and Card in 1986, Carroll and Campbell pointed out 4 faults with GOMS work. Two of those faults (GOMS arrives too late to influence design and is too difficult to apply) can be overcome by having such a tool as CRITIQUE integrated into a commercial IDE. If it is immediately available during the prototyping phase by being part of the tools the designer uses, then in fact an interface is not needed first before GOMS can be performed (but now it can be done in parallel). Furthermore, it is no longer hard to apply if the creation of a GOMS decomposition is automated.

The automated creation of a GOMS decomposition would also mean there is no need for an analyst/specialist. GOMS decompositions are subjective when done by people. The granularity of decomposition and determination of what is a unit task is often dependent on what the analyst deems. It is a pro to have this bias removed and to have consistency across all decomposition (which an automated decomposition could provide). Not needing an analyst would also mean not having to pay for this service. If there are any reservations about using a usability technique because of financial reasons, this tool should address that concern.

Finally, the obvious pro here is that if it is done right, then any evaluation is better than none. If it means that an obvious design flaw is detected and removed, this would obviously lead to better results. The best example of this is the use of GOMS to save NYNEX millions of dollars by correctly predicting that new workstations would not lead to performance as efficient as old workstations did.

Some of the cons with this idea however have to do with GOMS assuming expert users who make no errors. This is a serious problem because while GOMS can be used to predict expert use of a system and can inform optimization to a design, it may mislead designers into expecting that this kind of optimization will lead to efficient use of a system. But yet, as pointed out by the Olson’s, there is still no account for such things as a novice user, learning time, fatigue, mental load, etc. Thus the results of having an automated GOMS may not take into account these intangible factors that will be important in the use of the system.

Another con that has to be taken into consideration is the difference between usability and usefulness. It’s unknown how efficiently such a tool works (in terms of how much time it affects the development cycle). Should developers really devote a large amount of effort (if in fact it is time consuming to have this information and then go through several modifications to the system) when they are prototyping a system and it is still questionable whether or not a system is useful at all?

These would seem to be some of the pros and cons for having such a tool integrated into a commercial IDE. Of course, it too should also be evaluated before we can fully say what all the pros and cons are.

Part b)

There are a few reasons why GOMS has not been used more extensively in the practice of HCI design. In the well noted exchange between Carroll and Campbell with Newell and Card, four faults with the GOMS work was discussed. GOMS was questioned to be: too low level, too limited, arrives too late in the development cycle to influence the design, and is too difficult to apply. From this exchange, it was also explained that because a new canonical interface will come and dominate the scene at about every 5 years, it is hard to keep pace with the generation predictive models for all different possible interface designs.

Other limitations with GOMS, pointed out by the Olson's, can also explain why GOMS has not been used extensively in HCI design practices. GOMS assumes expert users who make no errors. There is no account for novice and intermediate users as well as the learning time for novice users. By presuming only non-novice users, GOMS fails to take into consideration the importance of individual differences in HCI. Furthermore, users are human, but GOMS does not address mental workload and fatigue. It also is common for slips to happen, so the assumption of humans who make no errors is flawed. The family of GOMS techniques typically focuses on low-level sensorimotor performance and touch a little on cognition, but GOMS ignores the broader work context.

Finally, no technique is in as widespread use as we would like for them to be. There are many reasons that affect this including financial, organizational and social ones as well as the merits of any particular technique.

Part c)

Another model of human cognition is Distributed Cognition. The main premise behind distributed cognition is that knowledge does not exist just within the head. It can exist also through external relationships with things in the world and with other people. The collection of actors, computer systems and other technology (often referred to as cognitive artifacts) and their relations to each other in the environmental setting in which they are situated forms the functional system. The system itself has cognitive properties differing from the individual parts involved. Within the system, it is expected that the member's knowledge are highly variable and redundant. The most important property of distributed cognition is the distributed access to the information within the system.

Distributed cognition effects the way we look at human behavior and interactive systems. The concept of mediating representations means that we no longer look at artifacts as a means of communication, but as something that embodies a piece of knowledge itself. Thus when we interact with an object, we are getting knowledge from the world...and sometimes creating new knowledge and placing it back into the system. In terms of human behavior, this model demonstrates that it is important for systems to be designed to convey the appropriate information. Keys to a design will include feedback, affordance, and visibility. As a human acts within a system, it is not necessary for all knowledge to exist in their head. Instead an application can be designed to take some of the load. They can recognize certain information and at other times they will sometime look to artefacts for clues to help them recall information.