TOWARD AN ANTIPHONY FRAMEWORK FOR DIVIDING TASKS INTO SUBTASKS

Benjamin D. Sawyer, Bruce Mehler, & Bryan Reimer
The Massachusetts Institute of Technology, AgeLab, Cambridge, MA, USA
Email: bsawyer@mit.edu

Summary: Task analysis is a staple of ergonomics, neuroergonomics, human factors, and experimental psychology inquiry, and often benefits from granularity beyond the task level to the subtask level. The concept and challenge of identifying the subcomponents of tasks are neither new, nor solved. Practitioners routinely identify individually internally consistent and yet conflicting subdivisions. The challenge of producing reliable, valid subtask data across efforts recommends a unified framework for identifying consistent subtask divisions within tasks. A framework is here forwarded, based upon universal "antiphony" turn-taking behavior in human-human interaction, but adapted to address the highly scripted vocabulary of human-machine interaction. Practical application to a real-world vehicle interface is demonstrated, an example discussed in the light of research design, applied use, and future improvement.

INTRODUCTION

The study of work, and the human factors and ergonomics that underpin successes and failures in work, has long relied upon analysis of highly variable tasks. Human Machine Interaction (HMI) inquiry likewise relies upon analysis of highly variable tasks within interface. Such efforts can provide vital insight. In surface transport research, for example, such efforts have shown the driving task to be highly variable in terms of workload, at times complex enough to tax and exceed the information processing capacity of the human operator (Senders et al., 1967). Indeed, analysis of driving performed concurrently with other tasks has shown evidence that such multitasking reduces the overall capacity available to the human operator, hindering their ability to respond appropriately (see Strayer, Drews, & Johnston, 2003 and Sawyer et al., 2014 for focused examples; see Hancock & Warm, 1982 for underlying theory). Results from this literature have been useful in establishing both design guidelines and policy toward safety on the road. However, the treatment of a complex task as a single unit limits the granularity of results, and so the depth of understanding of such inquiry. For example, task level data allows researchers to say that drivers are less likely to appropriately respond while engaging in a secondary task, but does little to identify design strategies to improve user interface design to mitigate this detriment.

Consider the challenge of comparing competing HMIs toward designing greater on-road safety in a given task. To weigh the contribution of each design element, researchers must explore the constellation of subtasks within the task. For example, a substantial body of research into driving while messaging, either through SMS or digital apps, analyzes epochs of vehicle control or attentional correlates. Through this work, the messaging task has been shown to cause detriment to control of the vehicle, as well as reduced operator ability to identify emergent threats on the roadway (Senders et al., 1967; Strayer, Drews, & Johnston, 2003; Sawyer et al., 2014). However, it is certainly the case that "messaging" is not a homogeneous task, and in fact is composed of subtasks as diverse as manual haptic button pressing, reading, route planning, and composition of

language (Sawyer et al., 2014; Sawyer & Clegg, 2010). How might such complexity be unpacked and analyzed? How do we identify major contributors to driving detriment?
The concept and challenge of identifying the subcomponents of tasks are neither new, nor solved. Both have long been addressed by work on keystroke level data (KLM, Card, Moran, & Newell, 1980), which includes the supposition that expert user behavior with computer systems can be predicted through a combination of modeling low-level operations such as keystrokes and heuristic rules for predicting mental effort. Such ideas play an important role in the GOMS model (Card, Moran, & Newell, 1983), and the many variants of that theoretical construct that have followed its development (for an overview of the family, see John & Kieras, 1996). There are, however, no accepted broadly applicable guidelines for the identification of the dividing lines between such low-level operations. Conventions for some common, homogeneously designed interfaces, such as the computer keyboard, do exist (John & Kieras, 1996). Complex, diverse, and so relatively unique interfaces rarely have accepted conventions for identifying low-level operators. The GOMS family of frameworks in many cases spring from adaptation of an existing model to the special cases of the interfaces at hand (see John & Kieras, 1996). More special cases mean less ability to generalize, and so recent technological trends toward more complex and diverse interface mean logical, universal, and generalizable conventions in subdividing tasks have become a more difficult goal.

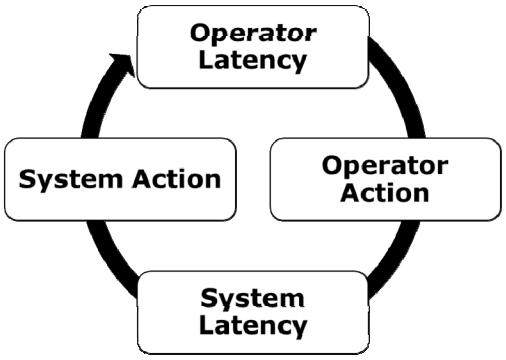
What is the correct way to identify the low-level subcomponents of a novel task? Certainly, there is more than one defensible answer. Efforts to model low-level operators in tasks within operationally diverse interface generally involve identifying logical divisions to delineate the subtask epochs that constitute that task. What is logical to one practitioner may not be to another, and conflicting and yet individually internally consistent subtask structures inevitably exist for any given task. While these inconsistent approaches may not be detrimental to any one effort considered in isolation, the need to compare efforts from disparate locations and times does exist. Many excellent task-analysis efforts are simply not comparable, despite the advantages to research, profit margins, and public good that such comparison might generate.

In the hunt for a framework to curry agreement in our own subtask analysis efforts, our team has turned for inspiration to a fundamentally human behavior: turn-taking in language. While subject to cultural, gender-based, educational, and other variations, such conventions are strikingly universal (Sidnell, 2007). Turn-taking is well studied in the linguistic literature, and described by theories notably including conversation analysis (Sacks, Schgegloff, & Jefferson, 1974), which describes the organization of turn-taking and turn allocation. Of course, discourse between operator and system presently lacks the richness and flexibility of human-to-human discourse. As such, turn-taking in human machine interaction may be better conceptualized as a limited vocabulary of available cues, some available to the human, some to the machine. This highly scripted responsive alternation is reminiscent of the scripted, and also scored, call and response of choral music, or *antiphony*. In the present work, we will use the same term to refer to call and response turn-taking between operator and human. An *antiphony framework* for dividing tasks into subtasks will be described, and examples provided.

The Antiphony Framework

A framework for division of task epochs into subtask epochs must be interpretable by practitioners. It must allow individuals separated by time and geography to come to comparable conclusions about subtask structure, the subdivision of task epochs into subtask epochs. Our antiphony framework must be usable by an increasingly globally diverse practitioner community, as turn-taking conventions do vary somewhat by language, among other factors (Sidnell, 2007). As such, the present work seeks to establish conventions for identifying the turn-taking in human-machine interaction. This antiphony framework will not presently be extended to teams of humans, or to groups of systems, although such elaboration is foreseen. Instead, we now focus upon the identification of subtask epochs over the course of one interaction (the task epoch) based upon identifying the responsive alternation between a single human (the operator) and a single interface (the system).

Fundamentally, any antiphony task epoch may be broken down around the stages of a cycle, itself composed of 1) operator latency, 2) operator action, 3) system latency, and 4) system action. Operator latency refers to delay due to perception, cognitive processing, and decision making. Operator action refers to production of any cues which direct the system. System latency refers to delay due to processing constraints. System action refers to the production of multimodal cues to the operator, or modifications to the environment. As an example, in a task broken into subtasks around the antiphony cycle, a driver observing the roadway might perceive and upcoming traffic jam and make a decision to make a phone call informing a friend of late arrival. This chain of events, from the start point of the observation of the traffic, would consider time to take the first action as 1) operator latency. Subsequently speaking out loud the activation phrase for a voice-activated phone would be described as 2) operator action. The time from the end of the operator's voice command to an acknowledgment from the system would be described as 3) system latency. An acknowledging audio response by the system, would be described as 4) system action. Here the onus of interaction would fall again to the human, who after potential 1) operator latency would perform an 2) operator action interpreted at the potential cost of 3) system latency and followed by a 4) system action.



I. Operators And Systems Process In Parallel
- I. Latency May Have Zero Duration

II. Inaction Is A Form Of Action
- I. Action May Have Zero Duration

III. All Actions Are Interruptible
- I. Subtask Times May Be Forshortened

IV. Subdividing tasks may start from any step
- I. Consider initiation & transparency

Figure 1. The four-stage antiphony cycle (left), and rules associated with its application to identifying subtask epoch divisions (right). Note that latency always precedes action, but may not be externally measurable.

Emergent aspects of this four-stage antiphony cycle (Figure 1) are in need of discussion. First, note that as both operators and systems are capable of processing the actions of the other in

parallel with the production of those actions, both operator and system latency subtask epochs may have a duration of zero. That is to say, for example, that a voice-activated system will begin buffering and analyzing voice data as it is spoken. Likewise, a human is well able to process many actions of the system without a subsequent pause, or indeed even waiting for their completion. Second, inaction is a form of action, and so both operator and system action subtask epochs may have a duration of zero. Both operators and systems may forgo available actions, either terminating the interaction or leaving the other party with the onus of continued interaction. Third, system and operator actions are interruptible, and so seemingly static epoch lengths, such as the time for a system to play a message, may be foreshortened. For example, operators may interrupt systems in anticipation of their full actions. Likewise, although more rarely, systems may interrupt operators, for example to deliver higher priority information or in anticipation of failure to correctly classify the operator's task. Fourth, the cycle may be measured from any step, depending upon task characteristics. Tasks are ideally measured using a precipitating event as a start point, to determine operator or system latency in response. However, while system-initiated tasks often have known start points, the start-points of human-initiated tasks are far more difficult determine. Specifically, systems can be made transparent, while human cognition largely remains a black box. Note, however, that evaluations of non-transparent interface may render systems similarly opaque in terms of latency, while neuroergonomic advances increasingly promise windows into human cognition (Sawyer et al., 2016). Ultimately, it may be possible to measure the beginning and end of all subtask epochs.

Practical Antiphony Subtask Identification

A practical implementation of an antiphony framework for subtask division of tasks must fulfill several important criteria. It must be interpretable by practitioners, straightforward to implement, and the result extensible to diverse, and potentially yet unimagined, interface. Here, and in support of these goals, we present steps for moving from a task, to understanding of the call and response turn-taking within that task, to identifying a list of subtasks. We further present an example drawn from AgeLab experiences with a real-world interface: address entry for voice navigation in a 2014 Mercedes CLA (Mehler et al., 2015). This interface includes interruptability and nonlinearity. It nonetheless can be considered only a moderately complex task in terms of difficulty to reduce to subtask epochs.

First, a script for operator-system interaction must be obtained. Specifically, this should be a list of 1) all actions the system will take, including error response actions, and 2) all *expected* actions from the operator. Designers of systems should have easy access to information on system actions, but those evaluating systems designed by others may need to draw such a script from records of user interaction. It is important in this case to use more than one interaction as a template, as error handling and nonlinearity within systems can lead to a variety of different paths that an operator might take to a goal. Conceptually, this can be considered a list of turns that might be taken, with an eye to subdividing at any point where the onus of continuing the "conversation" passes from operator to system or vice versa. Note that it is impossible to map all unexpected responses from a human operator, but it is presently possible to map all system responses. For example, in providing an address a system might have separate error handling for an ambiguous street number as compared to an ambiguous city. It is useful to represent such nonlinearity in a linear fashion, moving temporally through all the possible interaction paths a

user might take, including redundant paths. An example of such a "raw script", in this case drawn from records of user interactions, is presented on the left side of Figure 2.

Second, a basic script of the nuclear components necessary to navigate through the interface to the task goal should be constructed. If the task can end in multiple goal types, and such a basic script becomes complex and multi-faceted, consider the possibility that it might better be broken into two tasks. An example of such a "basic subtask script" is presented at the top right of Figure 2. It is worth noting that such a script can be the endpoint of the antiphony framework, without moving on to the third step suggested here.

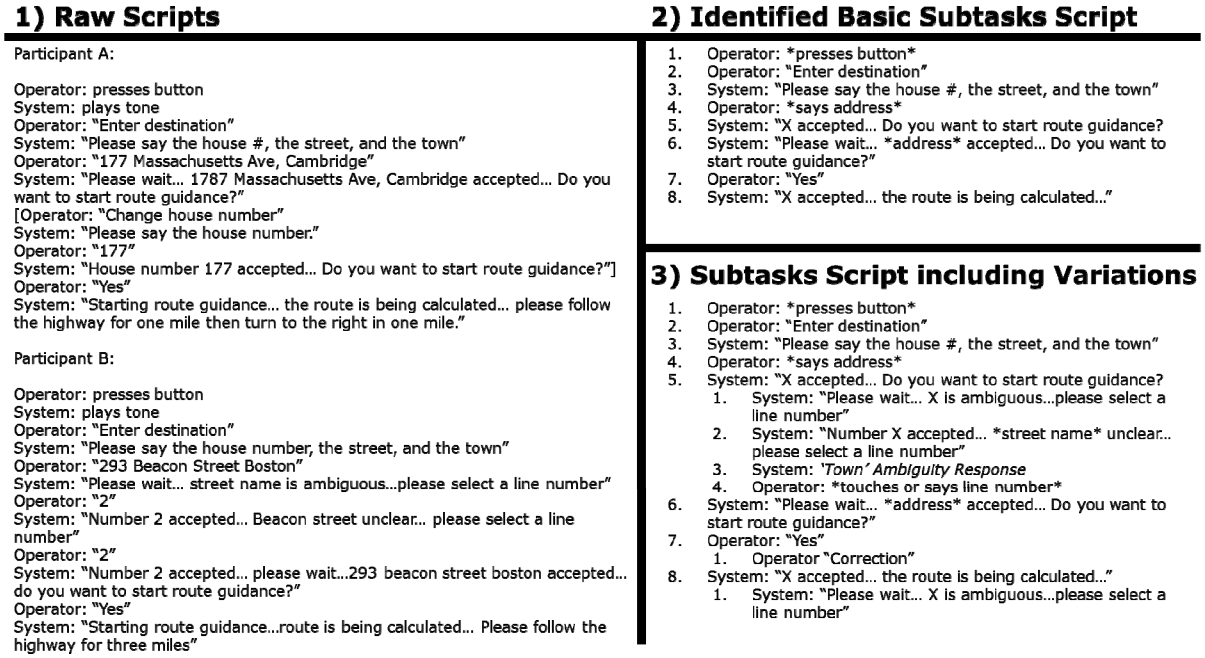


Figure 2. Starting with raw scripts (1), a basic subtask script (2) has here been identified, and delineates each step which must be passed through in order to achieve the goal. This basic subtask script is generated with an eye toward the turn-taking between the operator and the system. Optionally, variations in error handling (3) can be identified. In example 1, such variations are denoted by parentheses, while in example 3, they have been grouped under the basic subtasks from example 2.

Third, and dependent upon research design needs, variations and error handling loops may be included as subcomponents of a basic subtask script. This will likely involve a level of redundancy, for example if the response "yes" is included in several error handling loops. However, if subtasks and variations are dutifully coded, the additional granularity may lead to insights that are useful in both the research and design setting. Ultimately, such decisions should be based on a cost-benefit analysis.

Toward analysis of subtask level data, and specifically epoch lengths, the above subtask structure provides a definition from which to code start and stop times for both operator and system actions. These action times are enough to define the entire cycle, as both operator and system latency are implicitly defined as the times between, the "gap". Operator latency is defined as the gap before the operator action, while system latency is defined as the gap before system action.

Discussion and Next Steps

The antiphony framework, here demonstrated, is based upon call and response turn-taking in human-to-human communication and adapted to the uniquely scripted interaction presently seen between humans and machines. Although the examples presented here involve voice interface, this framework can be used in all manner of multimodal interaction. It is worth noting that the present work does not delve deeply into the research design considerations, or coding strategies, that one might encounter in inquiry focused on subtask level data. Briefly, and with understanding that it should be covered in more depth in future works, we will address each.

The coding of subtask data can be accomplished in a number of ways. Some efforts collect timestamps directly from the system being interacted with, a strategy which has been called device status reporting (DSR, see Sawyer et al., 2015). While this approach is only available in efforts where authorized or unauthorized low-level software and/or hardware access is available (as in Sawyer et al., 2014), and requires initial effort, it has the advantage of providing coding of subtask epochs in a thereafter largely automatic manner. DSR allows subtask information to be gathered alongside other experimental efforts, or potentially while users naturalistically interact with the system away from the confines of the laboratory. In situations where low-level access to a system is not available, digital recording and subsequent human and/or machine coding strategies are available. This approach is more time intensive, and suffers from the entropy that human decision-making and response time may inject. Double coding, and use of interrater reliability scores is highly advised (as explored in the context of glance coding in Reimer et al., 2014). Subtask data can, in some cases, involve subsecond epoch lengths, and it is important to consider the synchronicity of clocks responsible for coding various aspects of human and machine performance (for an expanded discussion, see Sawyer et al., 2015). The granularity associated with the present examples may be ill advised in certain projects, and cost-benefit analysis is encouraged. Specifically, overall coding time, especially when such coding is being done by humans, may be reduced in several ways. First, work may be reduced by half by coding only the beginning of operator and system actions, and can be reduced again by half by coding the beginning of *either* operator or system actions. Delineating a difference between operators and systems, and between latency and action provides potentially valuable information, both from the standpoint of a researcher and a practitioner. That said, using this framework to forgo delineating such a difference is also a valid use of the framework. The choice of which divisions to code and analyze are left to the judgement of the researcher.

Next steps in this effort will involve tailoring its use among a more diverse field of researchers and practitioners, and, if necessary, refining the methods to better achieve the stated goal. It is further expected that this antiphony framework can be expanded to multiple operator or multiple system scenarios. Such an expansion would be necessary to address the complex emerging realities of human-machine teaming (Cuevas et al., 2007; Caldwell, 2005). Finally, there is a need for recommendations for automated collection of subtask data through device status reporting (DSR), as well as best practices for achieving sub-second accuracy in the absence of real-time operating systems (Sawyer et al., 2015). Please, as you use the antiphony framework, feel free to reach out to our corresponding author with comments, constructive criticism and suggestions. Please share the outcomes of the framework's application in your own effort. The concept and challenge of identifying the subcomponents of tasks is neither new, nor here fully

solved, but we do hope that the present work provides a scaffolding to the benefit of those on the front lines of subtask analysis.

ACKNOWLEDGMENTS

Data utilized were drawn from a study supported by Toyota's Collaborative Safety Research Center and US DOT's Region I New England University Transportation Center at MIT (NEUTC). Support for this analysis and publication development was provided by the Toyota Class Action Settlement Safety Research and Education Program. The views and conclusions being expressed are those of the authors, and have not been sponsored, approved, or endorsed by Toyota or plaintiffs' class counsel. Thank you also to Michael E. Sawyer, for many excellent and relevant conversations, including those on antiphony, turn-taking, and rhythm and cadence of language. They were appreciated.

REFERENCES

Card, S. K., Moran, T. P., & Newell, A. (1980). The keystroke-level model for user performance time with interactive systems. *Communications of the ACM*, *23*(7), 396-410.

Caldwell, B. (2005). Multi-team dynamics and distributed expertise in mission operations. *Aviation, Space, and Environmental Medicine*, *76*(6), B145-B153.

Cuevas, H. M., Fiore, S. M., Caldwell, B. S., & Strater, L. (2007). Augmenting team cognition in human-automation teams performing in complex operational environments. *Aviation, Space, and Environmental Medicine*, *78*(5), B63-B70.

John, B. E., & Kieras, D. E. (1996). The GOMS family of user interface analysis techniques: Comparison and contrast. *ACM Transactions on Computer-Human Interaction (TOCHI)*, *3*(4), 320-351.

Hancock, P. A., Warm, J. (1989). A dynamic model of stress and sustained attention. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *31*(5), 519-537.

Mehler, B., Reimer, B., McAnulty, H., Dobres, J., Lee, J., & Coughlin, J. F. (2015). *Assessing the Demands of Voice Based In-Vehicle Interfaces-Phase II Experiment 2-2014 Mercedes CLA (2014t)*. MIT AgeLab Technical Report 2015-8. Massachusetts Institute of Technology, Cambridge, MA.

Reimer, B., Mehler, B., Dobres, J., McAnulty, H., Mehler, A., Munger, D., & Rumpold, A. (2014). Effects of an 'Expert Mode' voice command system on task performance, glance behavior & driver physiology. *In Proceedings of the 6th International Conference on Automotive User Interfaces and Interactive Vehicular Applications,* 1-9. ACM.

Sacks, H., Schegloff, E. A., & Jefferson, G. (1974). A simplest systematics for the organization of turn-taking for conversation. *Language*, 696-735.

Sawyer, B. D**.**, Finomore, V. S., Calvo, A. A., & Hancock, P. A. (2014). Google glass: A driver distraction cause or cure? *Human Factors*, *56*(7), 1307-1321.

Sawyer, B. D., Karwowski, W., Xanthopoulos, P., & Hancock, P. A. (2016). Detection of error-related negativity in complex visual stimuli: a new neuroergonomic arrow in the practitioner's quiver. *Ergonomics*, 1-7.

Senders, J. W., Kristofferson, A. B., Levison, W. H., Dietrich, C. W., & Ward, J. L. (1967). The attentional demand of automobile driving. *Highway Research Record*, (195).

Sidnell, J. (2007). Comparative studies in conversation analysis. *Annu. Rev. Anthropol.*, *36*, 229-244.

Strayer, D. L., Drews, F. A., & Johnston, W. A. (2003). Cell phone-induced failures of visual attention during simulated driving. *Journal of Experimental Psychology: Applied*, *9*(1), 23.