Usability Issues and Guidance for Flexible Execution of Procedural Work

Dorrit Billman

San Jose State University @ NASA Ames Research Center   
dorrit.billman@nasa.gov

Debra Schreckenghost

TRACLabs   
schreck@traclabs.com

Abstract

We believe user experience in complex work domains is shaped by the effectiveness of technology in jointly accomplishing work goals. Function allocation between humans and smart technology is an important part of effectiveness, in turn, an important contributor to user experience. We study operation of equipment for the International Space Station, using procedure automation with flexible function allocation. We discuss automation goals, their impact on suitable function allocation, and the role of flexible function allocation. We offer examples and propose guidelines.

Bios & Research Background

Dorrit Billman is a cognitive scientist with background in learning and in human factors of tools supporting complex cognitive work. She has researched tools for ecological modeling, collaborative intelligence, planning by Mission Control operators, and most recently equipment operation, in collaboration with Debra Schreckenghost. Debra Schreckenghost is a Senior Scientist at TRACLabs. She has conducted research in the areas of adjustable autonomy, human interaction with automation, and real-time performance assessment of robots and automation.

Our research investigates design of procedures, procedure automation software, and particularly, how users interact through software to operate complex equipment. From observing users working with a variety of function allocation policies, we have identified a number of issues around teaming with smart technology.

Source Domain

Our domain concerns user-automation interaction in accomplishing technical work, specifically operation of habitat systems for the International Space Station. Currently, this work is carried out manually by astronauts following written procedures with support from Mission Control. Future operations are expected to rely much more on automation, to increase the autonomy of crew and decrease dependence on Mission Control. Our studies investigated users carrying out (simulated) operations using procedure automation software [1]–[3].

In work domains and particularly in safety critical, technical domains, the key elements of user experience are effectiveness and efficiency in accomplishing the work. Contributions of the technology to these goals are more typically described as usefulness and usability. However, the centrality of these aspects to “user experience” cannot be over-emphasized; people’s experience is deeply shaped by their ability to accomplish work goals effectively. Despite the domain differences, we believe that ability to use “smart technology,” whether labeled as automation or AI, to accomplish goals is important in many domains. We expect there will be important cross-domain commonalities, particularly in the design issues and trade-spaces, while appropriate solutions will vary to meet domain needs. We discuss the issues and design trade-spaces for how functions should be coordinated and distributed across human and artificial entities. We invite discussion of domain similarities and differences.

Distribution of Work in Procedural Work Domains

Our research addresses work in procedural domains. Procedural work domains are those where work is meaningfully organized around individuated, discrete actions, and correct ordering of the actions is important and somewhat standard. Typically the importance of sequencing comes from physical constraints, but it could also be driven by human benefits from establishing a standard (set of) sequence(s). Procedural work contrasts with continuous control, where the work is not primarily segmented into discrete units. It may also contrast with domains where work is composed of discrete actions, but there is little importance to sequence or good sequences are not standard; creative work might provide examples or work domains where sequences are highly situation dependent. We do not imply that all work in “procedural domains” involves procedures, but that procedures can centrally contribute to accomplishing work goals. Here we focus on mixed initiative procedure execution, involving one person who is working with automation.

Supporting procedural work involves several design considerations. The foundational design guidance is derived from an analysis of the work domain, including characteristics of the work goals, users, technology, and work context. Without a sound understanding of the work needs, procedure automation software is unlikely to be helpful. A second source of design guidance and constraint comes from the design of procedures appropriate to the domain. The high-level goal of a procedure is typically to improve productivity, or to mitigate the effects of anomalies. Additional goals may include safety and risk management, efficiency, consistency, and appropriateness for execution by available human operators with available technology. Where automation is used, it is a valuable strategy to use procedures to organize how the automation contributes. A third source of design guidance comes from the capabilities of the entities available to execute these procedures and the reasons for using automation to execute the work, if that is to be included. In procedural domains automation is often introduced to execute component actions originally executed only by humans; in such cases, automation is introduced into an existing work process. Alternatively, automation may be used for tasks that people cannot themselves do directly. If both the human and the automation are capable of carrying out certain units of work, whether a whole procedure or one action within it, then there are choices in how work should be distributed among humans and technology and a function allocation policy is needed.

Design of a function allocation policy should be informed by the overall purpose of automating. Design of function allocation needs to specify the units of work that can be allocated (e.g. an action, a whole procedure, certain types of actions, etc.). In addition, for relevant procedural units, allocation may concern executing the whole unit of work (e.g. manually plugging a device in vs automatically switching on current), or may concern initiation of an action (e.g. a user triggers a command to automatically change a component). If there are function allocation choices, there may also be flexibility in which allocation is chosen. This may serve a domain need for flexibility in the work more generally.

Goals Served by Function Allocation

Function allocation, or distribution of task-work across team members, should be guided by the objectives important in the specific domain. Function allocation can serve multiple, sometimes conflicting objectives, beyond the overall goal of the specific procedure.

Goals favoring maximal allocation to automation:

-Reduce human error in initiation (picked wrong time or wrong action, failed to follow procedure when applicable);

-Reduce human error in execution (confirms something not true, takes action on incorrect object;)

-Protect the human,(from effects of error, and particularly from operation in a hazardous environment).

Goals favoring maximal allocation to humans:

-Reduce automation error in initiation (applies specified procedure action when inappropriate, typically, because conditions not anticipated at time of procedure design)

-Reduce automation error in execution (attempted action fails due to hardware problem, data connectivity problem, lack of needed resources)

- Humans have unique skills that are hard or costly to replicate with automation

Complex goals favoring mixed allocation:.

-Maximize assurance from dual involvement on task components (both must approve; command and verification splitting; analogies to pilot/co-pilot checklist use).

-Maximize execution speed (automation can be faster in simple conditions but when mixed execution is required, speed is affected by added team work, e.g., at handovers,)

Goals likely to favor flexible allocation:

-Improve or sustain human monitoring, learning, skills, engagement, or understanding of system operation.

-Maximize usable human time for other tasks.

-Adaptation for changing conditions, resources, or goals.

Frequently multiple goals, possibly in competition, are in play. In our domain, a key goal is effective use of human time. Astronauts have high workload and limited time, so freeing time is an important goal for automation use.

We are particularly interested in options for *flexible function allocation*. Effective operations can require flexibility in what is done and in who does it. For example, in current practice changes to procedures may be required when equipment ages or is modified. Flexible allocation would allow a shift from automatic to manual control for this adaptation. In addition, certainty that a procedure is appropriate and will have the intended effect may drop if a new component is installed or systems are configured in a nonstandard way. In such cases it may be particularly valuable to have a person closely monitoring and manually initiating procedure actions. Conversely, as equipment, context, and procedure become well-established, automatic execution may be most effective. State of the human operator will also influence the relative benefit of automated vs manual execution.

Allocation Goal: Freeing Human Time

In our studies we were particularly interested in how design of function allocation affects availability of useful human time. While improved quality of work is a reason for human or human-automation teaming, greater efficiency is also a motivation. With the types of procedures executed in our domain, much of the execution time is a function of the equipment operated, not the operator actions. When the person does not need to be actively involved, this can free up operator time for manual work in parallel activity.

***Team-work costs vs task-work benefits***. As with human collaboration [4], time spent coordinating with team members (team-work) can subtract from time available for task work. In general, added team-work should be less than the reduction in task-work. In our domain, identifiable teamwork time-costs occurred at handovers. There may be other costs, such as awareness or workload.

Measures: direct measures of total operator time; number of operator actions; workload; awareness of system state.

Examples: Users sometimes deviated from the stipulated allocation plan when it seemed to impose a high team-work cost. When a small block of automated actions occurred between actions to be done manually, users sometimes did these to-be-automated actions manually, apparently to avoid the team-work cost of task switching.

Guidance: Primary costs in human time come at task handoff, where this team-work function adds time above all-manual or all-automated execution. Primary benefits to human time come from continuous stretches of automatable actions. Proportion or number of automated actions may be less consequential than their distribution. There may be a tradeoff between maximizing the amount of automated work and minimizing handoffs.

One allocation strategy is to allocate commands to a person and verification of command effects to automation. This can increase team-work costs due to frequent handoffs between humans and automation. When such policies are required, the team-work cost of handoff should be minimized in other ways, e.g., design the automation to automatically resume when manual actions are complete.

***Handoff frequency***. Minimize the frequency of handoffs and maximize the time span of automated actions between them. The longer the user has between manual actions, the larger the block of time available for other work. Reducing handoffs reduces time on team-work.

Measures: longest, average, and shortest automation sequence in actions and in time; hands-off blocks of useful length. Number of handoffs.

Examples: When users encounter short stretches of automatable actions between manual ones, they often did the automatable actions manually to eliminate handoff costs.

Guidance: When condition dependencies permit, the actions in procedures intended for human-automation execution should be grouped and ordered into Manual Only action sets and Automatable action sets. This can reduce the frequency of required human-automation handoffs. Note that this order principle may compete with others, such as a grouping actions together that accomplish the same goal.

***Interruption.*** Interruptions disrupt human work, when automation requires human input, at handoffs or procedure completions. Interruption from team-work should minimize impact on task-work. This may be mitigated by interface and information design. For example, information about importance and urgency can be provided.

Measures. Disruption as time for user to initiate/resume action given a usable time-block. Impact on second, manual task.

Example. The automation pauses when a manual only action is reached, an automated action fails, or the procedure completes. While the user should be made aware of all these pauses, there usually is more urgency in responding to a manual action or the failure of an automated action.

Guidance. The automation interface should notify the operator when it stops doing task work and indicate why it stopped. Reasons for stopping include reaching a planned handoff, an interaction between procedures or users that requires user acknowledgement, or halting abnormally. The saliency of these notices should be designed to balance the urgency and importance of information with the potential for unnecessary distraction*.*

***Situation awareness with automation***. The goal of reducing human time interacting with automation performing procedural work must be balanced against the need for the human to maintain awareness of both the domain systems and automation behavior. The design of procedures that are performed with less human involvement can require providing additional information about both the system states and the task completion to orient the user when intervention is required.

Example. The use of the procedure as the basis of automation is intended to make the automated actions more transparent to the user. We found it useful to provide a user interface that annotated the text of the procedure with information about what actions had been taken, what system states had been changed, and what action the automation is currently performing.

Guidance. Situation awareness and problem solving with procedures may be improved by including in the procedure information about what system states or environmental conditions the procedure actions are intended to change and what system states or environmental pre-conditions are assumed to be true before actions are executed..

In addition, an automation system might have information on activity in multiple tasks. This could form “situation awareness” about probable workload and could guide changes in order or timing of demands for manual actions. Prioritizing actions across tasks is an active research issue in aviation autoflight systems.

Allocation Goal: Flexibility

If procedure designers could perfectly predict the conditions and goals in force when a procedure would be executed, it might in principle be possible to specify an optimal allocation policy. However, in any complex domain this is unlikely to be true, just as it is unlikely for every possible contingency to be anticipated in the design of a procedure. Note that while we describe flexible allocation in terms of assignment to automation or to one operator, many of the same factors and issues apply to mixtures of multiple operators and/or automated systems. Several factors affect the value of flexible allocation, as indicated below:

1) More valuable: Domain is more complex, less understood, or has lower degrees of predictability, often typical of technical operations in dynamic environments.

2) More valuable: Normal operations include high variability in system and user behavior. Even anticipated variation can have a high planning cost.

3) Less valuable: In highly predictable domains, standardization may be possible and valuable, e.g., in aiding handoffs, or for collecting data about system performance during operations.

***Scope and nature of flexible allocation*.** Function allocation can be done by the designer when a procedure is written. In addition, the operator may be able to allocate functions in advance of procedure execution (an allocation plan), or during procedure execution (reactive allocation). Flexible allocation is only possible for actions that can be done both manually and automatically. If these are a small number or small proportion of the procedure actions, it is unlikely to be worth the overhead of supporting flexible allocation for such procedures. Of course, when all actions can only be done manually or only automatically, reallocation is not possible.

Example. We designed in a need for reallocation and assessed methods for doing this. For example some tasks can require reallocation when components used in the task have been replaced or repaired, and the user should execute relevant actions manually to check for normal operation.

Measures. Benefits and costs may be measured separately or in terms of overall performance. Benefit might be modeled or estimated in the abstract from the size or nature of possible allocation plans, coupled with estimates of how conditions are likely to change. Once a running prototype exists, performance can be compared with the prototype or prototype variants, when operator allocation is or is not allowed.

Guidance. Prior to designing for flexible allocation, the cost benefits should be assessed, relative to the goals. Flexible allocation adds complexity for the operator as well as development costs, so there should be commensurate benefit. Avoid adding complexity without benefit.

***Methods for changing function allocation***. We suspect that there are many situations where reactive as well as planned reallocation by the operator is valuable. Where function reallocation is beneficial, methods for both planned and reactive allocation need to be provided to the operator, which fit within in the operational context.

Example. These two styles of allocation are suited to different circumstances. For example, an allocation plan might be used to standardize routine allocations, such as automating all verify instructions. Reactive allocation can be used to ‘short cut’ a local inefficiency, such as doing a couple actions manually rather than shifting to automation, given the overhead of that shift. It also is important to consider what improvement is anticipated by the change in function allocation. For example, our study of flexible allocation focused on reducing human workload by allocating functions to automation to increase time available for humans to work on other tasks.

Methods for reactive and planned reallocation will differ. For example, for reactive reallocation, we provided a simple method to stop automation at the end of the currently executing command. Multiple methods were provided for planned reallocation, including reassignment of actions throughout a procedure. (Recall that actions include cognitive actions such as verifications as well as commands to change the controlled system.)

Guidance. In many situations it should be possible to revise the allocation plan while in use. This requires informing the operator which actions and procedures are done and which can be re-allocated. Changing the allocation of functions during operations is a type of team-work that takes human resource away from task-work. It is important to consider how the changed allocations might actually improve performance. Determining the goodness of flexible allocation of functions requires understanding the intended benefit and assessing whether the intended benefit was actually incurred.

If freeing human time is the goal, it is important to consider whether the increase in human team-work to change allocations is greater than the reduction in human task-work. Other goals, such as improving human safety, require other evaluation criteria.

***Transparency of function allocation***. Function allocation assigns units of work to manual or automatic execution. The types of work units that can be assigned should be clear. In our case, the assignable units of work could be a procedure as a whole, a procedure step, and a procedure action. In our case, any unit could be executed manually, but only some could be executed automatically. Thus, if the user selected automatic execution for a whole procedure, automation stopped when a manual-only action was encountered. It is important to display explicitly what the current assignment of a work unit is and also display what alternative assignment is possible. In addition to support for the assignment process, the handoff during execution should be clear. That is, when executing, it should be clear to the user when the automation will act next, and when further action from the user is expected.

Example. One procedure interface required the user to infer which actions could be automated (by recognizing that automatable lines required a telemetry read-out or command button). A number of users had difficulty in determining which actions were automatable. After a redesign, each procedure action was explicitly marked as manual only (M) or automatable (A). Using this new interface, very few users had difficulty in knowing which procedure actions would execute automatically, which actions must be done manually, and on which line automation would pause. As a second example, after a manual action the automation might resume on its own, or require an explicit user action to resume.

Guidance. The procedure automation interface for executing flexible allocation plans should inform users which actions are marked for automated execution and which actions are marked for manual execution. The automation interface should make clear what allocation choices are available to the user during allocation; from the execution perspective, it should also make clear when the system requires user response and when automation will act.

***Coordinating interactions among procedures and agents*.** In most domains work activities may draw on multiple procedures, in sequence and also with overlapping execution. Interaction among procedures might result from effects of working on multiple activities, as from multiple operators, or when multiple procedures are needed for one activity. Procedures may interact, as when a state required by one procedure is controlled by another. Flexible task allocation can permit the operator to designate points in the procedure where the automation will pause until the operator resumes it. Pausing to coordinate can be used to ensure that the state required by one procedure but controlled by another is achieved before continuing. Pausing can also coordinate interaction between operator and automation. For example, an action might need confirmation by the operator for automated execution before the automation can continue i.e., shared authority. The confirmed actions are not performed by the person but rather are reviewed and approved before automation resumes.

Example. A user might automate the activation of the carbon dioxide removal system (CDRS) while she manually activates the active thermal control system (ATCS) needed to cool the CDRS. Because cooling water is not needed until later in the CDRS activation procedure, the user might mark a coordination point in the CDRS activation procedure that would pause automation prior to the point where cooling water is needed. Once the user completes ATCS activation, she could direct automation of the CDRS activation to resume.

Guidance. The user should be able to mark points for coordination or confirmation between human and automation task work when allocating procedural tasks.

Conclusion

In technical work domains, we suggest that a core driver of positive user experience is effectiveness in accomplishing the work. We discuss the issues that emerge in designing ‘team-work’ as well as ‘task-work’ for mixed initiative operation of equipment. A key element is function allocation. The goals that automation was intended to serve influence design of function allocation, and particularly the role of flexible function allocation. Many of the guidelines we suggest note the importance of methods for transparency. We believe that flexible function allocation, as well as representing a fine-granularity of work that can be (re)allocated, will be important for effective human-technology teaming. Flexible coordination of goal-directed activity is likely to be important for user experience across multiple domains.

Acknowledgements

The work on flexible allocation to automation was supported by the National Space Biomedical Research Institute (NSBRI) through NASA NCC 9-58.

References

[1] D. Billman, D. Schreckenghost, and P. Miri, “Assessment of Alternative Interfaces for Manual Commanding of Spacecraft Systems: Compatibility with Flexible Allocation Policies,” *Proc. Hum. Factors Ergon. Soc. Annu. Meet.*, vol. 58, no. 1, pp. 365–369, Sep. 2014.

[2] D. Billman, D. Schreckenghost, and S. Billinghurst, “Problem Solving when Procedure Conditions Are Not Met: Using Procedure-Automation Software for Support,” in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Los Angeles, 2015, vol. 59, pp. 294–298.

[3] D. Schreckenghost, T. Milam, and D. Billman, “Human performance with procedure automation to manage spacecraft systems,” 2014, pp. 1–16.

[4] J. E. McGrath, “A Conceptual Framework for the Study of Groups,” in *Groups: Interaction and Performance*, Englewood Cliffs, NJ: Prentice-Hall, 1984schre.