AFRL-HE-WP-TP-2005-0029

AIR FORCE RESEARCH LABORATORY

# Using Delegation as an Architecture for Adaptive Automation

Chris Miller
Smart Information Flow Technologies

December 2005

June 2005

Human Effectiveness Directorate
Warfighter Interface Division
Wright-Patterson AFB OH 45433

| REPORT DOCUMENTATION PAGE | Form Approved OMB No. 0704-0188 |
|---|---|

| | | |
|---|---|---|
| **1. REPORT DATE** *(DD-MM-YYYY)* Dec-2005 | **2. REPORT TYPE** Technical Paper | **3. DATES COVERED** *(From - To)* |
| **4. TITLE AND SUBTITLE** Using Delegation as an Architecture for Adaptive Automation | | **5a. CONTRACT NUMBER** |
| | | **5b. GRANT NUMBER** |
| | | **5c. PROGRAM ELEMENT NUMBER** |
| **6. AUTHOR(S)** *Chris Miller | | **5d. PROJECT NUMBER** 7184 |
| | | **5e. TASK NUMBER** 09 |
| | | **5f. WORK UNIT NUMBER** 72 |
| **7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) AND ADDRESS(ES)** *Smart Information Flow Technologies | | **8. PERFORMING ORGANIZATION REPORT NUMBER** AFRL-HE-WP-TP-2005-0029 |
| **9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)** Air Force Materiel Command<br>Air Force Research Laboratory<br>Human Effectiveness Directorate<br>Warfighter Interface Division<br>Wright-Patterson AFB OH 45433-7022 | | **10. SPONSOR/MONITOR'S ACRONYM(S)** AFRL/HECI |
| | | **11. SPONSOR/MONITOR'S REPORT NUMBER(S)** |
| **12. DISTRIBUTION / AVAILABILITY STATEMENT** Approved for public release; distribution is unlimited. | | |
| **13. SUPPLEMENTARY NOTES** This is a chapter to be included in a NATO-RTO technical report. Clearance: AFRL/WS-05-2504, 28 Oct 05. | | |

**14. ABSTRACT**

**Humans and automation can interact in a huge range of different ways, and the number of ways increases as computer technology enables automation to do more, and do it via new and different modalities. A "Level of Automation: framework, as we defined it above, is simply a convenient parsing of the myriad different ways humans and automation can interact into some convenient set of categories or levels. In such frameworks, an LOA labels a range of relatively homogenous, alternate human-automation relationships.**

**15. SUBJECT TERMS**
Adaptive Automation

| **16. SECURITY CLASSIFICATION OF: Unclassified** | | | **17. LIMITATION OF ABSTRACT** | **18. NUMBER OF PAGES** | **19a. NAME OF RESPONSIBLE PERSON** Mark Draper |
|---|---|---|---|---|---|
| **a. REPORT** UNC | **b. ABSTRACT** UNC | **c. THIS PAGE** UNC | SAR | 10 | **19b. TELEPHONE NUMBER** *(include area code)* (937)255-255-5779 |

# Using Delegation as an Architecture for Adaptive Automation

Prepared by: **Chris Miller, Smart Information Flow Technologies, LLC**

**cmiller@sift.info**

**File name:** **Miller-3D-LOAs.doc**

## 1. What is an LOA?

Humans and automation can interact in a huge range of different ways, and the number of ways increases as computer technology enables automation to do more, and do it via new and different modalities. A "Level of Automation" framework, as we defined it above, is simply a convenient parsing of the myriad different ways humans and automation can interact into some convenient set of categories or levels. In such frameworks, an LOA labels a range of relatively homogenous, alternate human-automation relationships.

A human + machine system has *more* automation (or, alternatively, a higher "level of automation") when any of three things is true:

1. The automation can be tasked at higher, more abstract levels—that is, it can be relied on to make more of the decisions about how to execute a broader, higher-level task. The system is making decisions that would otherwise have to be made by a human supervisor: decisions about how to achieve lower level goals. If I can tell a subordinate "Get me to LAX on Monday", that subordinate is more autonomous than if I have to tell him/her "Book me a flight to LAX on Monday; make sure you use Orbitz because they're cheaper; it'll be good if you can make it a non-stop; and make sure to get me a car and hotel, too".

2. The automation can perform whatever it does more independently, with more authority; it doesn't have to submit its decisions to me for approval or review. If my subordinate has to have approval of each decision about my travel plans, I'd say s/he is less autonomous than if I s/he can simply hand me a complete itinerary with tickets.

3. The automation controls more resources or assets. If I allow my subordinate to make $5000 decisions about my company's credit line (not to mention booking the corporate jet and limo fleet ☺), then s/he has more autonomy than if any decision over $100 has to be routed to me for approval.

These dimensions are not entirely independent. Instead, they provide three views into the delegation relationship between a supervisor and subordinate and they together define a "delegation space" within which achievable relationships can be described.

It is not surprising that prior efforts to characterize Levels of Automation express one or more of these dimensions. Sheridan's levels (as shown in Figure 1 above) say less about specifically what tasks or goals automation is performing, and more about the relationship between the human and the automation. That is, the levels describe the autonomy or *authority* relationship between human and automation. Does the automation have the authority to do whatever it is it's doing without prior approval? If so, then it operates at Sheridan's level 10. If it can do whatever it does under its own recognizance but must report what it has/is doing to the user, then it is at Sheridan's level 7, etc. This spectrum describes *how* human and automation relate to each other, but doesn't say much about specifically *what* each of them is doing. For convenience, we will refer to this characterization of authority or autonomy relationships between human and automation as the ***Level of Authority*** dimension.

By contrast, Parasuraman, et al.'s two-dimensional scheme adds some description of *what* the automation is doing, albeit in terms of four coarse-grained information processing categories. This dimension of what is being done is crossed, however, with the same sense of how the relation is characterized (the Level of Authority) above. Parasuraman, et al., imply that the "high" and "low" dimension for each of their information processing functions is related to Sheridan's initial 10 level spectrum. Thus, the Parasuraman, et al. framework subsumes Sheridan's initial framework and now defines a Level of Automation as a combination of a level of authority for each of the four information processing functions.

What is the dimension that Parasuraman, et al. have added? As we have noted above, it begins to describe what each of the actors is doing within a specified LOA—therefore it is a description of activity. We argue that the activity dimension can be subdivided into many finer categories and represented by a hierarchical task model. In other words, the four information processing stages are just a coarse aggregation and categorization of specific tasks. Instead of saying that "information acquisition automation is high" we could more precisely say that automation is deciding how to direct and configure sensors and reporting those decisions to the operator. For a more detailed elaboration of this argument, see Miller and Parasuraman, 2003; forthcoming.

Since we can use a hierarchical task model to characterize what human and automation are doing, we can refer to this dimension of automation activity or behavior as a ***Level of Abstraction***. Automation can have responsibility for higher- or lower-level tasks within the task hierarchy. Having responsibility for higher level tasks presumes responsibility for the tasks which fall below them (although the person doing the delegation retains the right to place constraints or stipulations on the range of decision possibilities about how to perform lower level tasks). So a "Level of Automation" is, therefore, even in Parasuraman, et al.'s model, a combination of a level of authority and a level of abstraction—automation has responsibility for one or more tasks at a given level of abstraction and with a given level of authority.

Are we done? Not quite. The Level of Authority x Level of Abstraction framework described above characterizes who is doing what and how they relate to each other. Especially in military domains, however authority relationships are frequently defined along resource lines as well as task or functional relationships. Hence, we define a third dimension along which to characterize delegation—a ***Level of Aggregation***[1]. The level of aggregation identifies how much (and/or which type) of resource each actor is authorized to use. When a supervisor delegates a task to a subordinate, that subordinate will be granted authority over some set of resources (including access to his or her own time and energies). The subordinate is expected to be able to either come up with a plan for completing the task within those resources or to tell the supervisor why s/he cannot. On the other hand, it is also frequently the case that there are some shared resources that the subordinate may have access to only after performing various coordination activities which enable their use. The extent of the subordinate's authority over resources can be categorized in a

[1] The choice of the term "aggregation" here to refer to the part-whole dimension of objects in a system, as well as the reference to a means-ends dimension of functions and sub-functions as an "abstraction" dimension, are conscious references to Vicente (1999) and Rassmusen's (1984) framework for Cognitive Work Analysis, which use the terms in a similar fashion. Plus, they provide nicely alliterative LOA abbreviations. We do not otherwise claim to be using the Cognitive Work Analysis framework here.

way similar to the way we have categorized the subordinate's authority over his/her own activities.

These three dimensions, Level of Authority, Level of Abstraction and Level of Aggregation, therefore define a ***Delegation Space*** of human-automation relationships within which delegation occurs and can be characterized. In short, ***delegation means giving to a subordinate the responsibility to perform a task (with its subtasks), along with some authority to decide how to perform that task and access to some resources with some authority to decide how to use them to perform the task.*** Thus, the three scales must be used to specify four variables which define the delegation space: the level of abstraction and the level of authority on it, and the level of aggregation and the level of authority on it.

Figure 3 illustrates these relationships. Imagine a set of tasks which can be performed in a domain, arranged in a hierarchical, abstraction relationship. A supervisor *controls* some portion of those tasks—which means s/he has *authority* over deciding when and how they need to be performed and when and how to use resources (which s/he also controls) to accomplish them. When the supervisor delegates some of those tasks, s/he is delegating that control—over the decision(s) about how and when they need to be performed, and over the decision(s) about how and when to use allocated resources to accomplish them.

The authority to make those decisions need not be complete. The supervisor can assert constraints on how the subordinate makes those decisions and/or can require the subordinate to perform various degrees of checking and request approval before the proceeding with a plan, but there must be *some* authority to make those decisions handed over if there is to be any benefit from delegation. A "Level of Automation" is, therefore, a combination of tasks delegated at some level of abstraction with some level of authority and resources delegated with some level of authority to be used to perform that (and perhaps other) task(s). The "level of automation" in a human-machine system increases if the level of abstraction, level of aggregation or level of authority (on either abstraction or aggregation) increases.

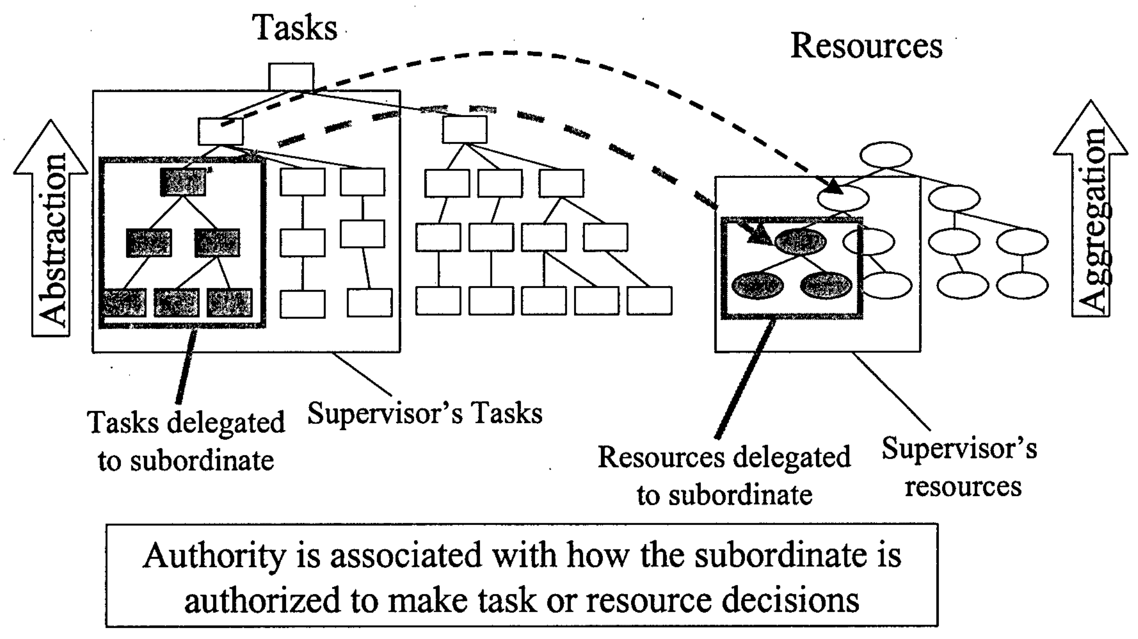
Tasks
Resources
Abstraction
Aggregation
Tasks delegated to subordinate
Supervisor's Tasks
Resources delegated to subordinate
Supervisor's resources
Authority is associated with how the subordinate is authorized to make task or resource decisions

Figure 3. Delegation relationships as characterized by abstraction, aggregation and authority dimensions.

In the next section, we will work through an extended example illustrating how these dimensions can be used to identify and characterize various human-automation relationships. In the following section, we will describe how they can serve as a framework, especially in conjunction with a Playbook interface, for configuring and exploring different human-automation relationships in experimentation. In the final section of this paper, we will describe how this framework could be used before or even during a mission by an operator to actively manage, control and maintain awareness of what his or her automation was doing.

## 2. An Example of Interactions in the Delegation Space

As a simple example of the delegation space and its utility in characterizing different human-automation relationships, let's explore the following set of tasks in a hierarchical relationship:

- Assume that the highest level task/function is something we might call "Overfly target" which, for this example, is defined as getting an aircraft, equipped with an appropriate sensor, to fly within an acceptable distance of a designated target and having it take some sensor imagery. This function is a top-level "play" in a Playbook.
- Overfly is decomposed as illustrated in Figure 4. Each level of decomposition contains the sub-tasks (or sub-plays) required to accomplish the parent task. It also contains alternate methods (alternate strings of sub-tasks) that could be used to accomplish the parent (though none are illustrated here). For example, the task "Achieve Airborne" could be accomplished by obtaining a plane and having it take off (via various methods) or by requisitioning one that is already airborne.
- The only task that is further decomposed to a third level is "Fly to Target" which is decomposed into a sequence of "Fly-to-Waypoint" tasks. Each of the other tasks at the second level could be further decomposed but has not been for simplicity. The loop around "Fly-to-Waypoint" indicates that it may need to be repeated a number of times.

### 2.1 The Abstraction Dimension

Figure 4 illustrates the Abstraction dimension. Overfly is a more abstract task than is Fly-to-Waypoint. This is the dimension that Playbook has traditionally manipulated. Operators with a Playbook can accomplish an Overfly mission by commanding "Overfly" (i.e., at a high level of abstraction) and letting Playbook's Planning and Analysis Component (PAC) figure out the best way it can to perform all of the sub-tasks in Figure 4, or (in principle) by issuing commands at the intermediate level of abstraction—saying, for example, "Achieve Airborne" (and having the PAC chose a likely aircraft), then saying "Fly-to-Target" (and having the PAC develop a route and

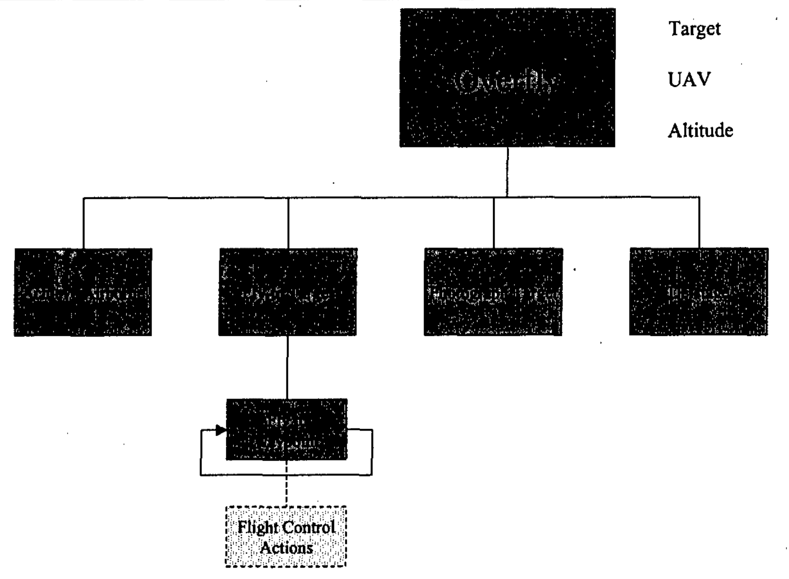

Figure 4. A simplified task decomposition for Overfly.

fly it) or by issuing a series of waypoint commands—each, essentially, a "Fly-to-Waypoint" command in it's own right. Thus, the operator with a Playbook can traverse the Abstraction dimension at will.

Does this ability for a Playbook to traverse levels in the Abstraction dimension make it an AAA? As with many things, it depends on how we define our terms. As one moves up in levels of abstraction (or tasks in a task hierarchy), one is referencing tasks at higher levels. If the automation is competent to handle the planning and execution at these levels, we enable operators to hand off higher-level tasks to automation, and we are generally saving them workload. This workload savings has always been a motivation of higher levels of automation. Thus, it seems entirely appropriate to call Playbook a type of Adaptive Automation Architecture.

On the other hand, as we saw above, Sheridan's Levels of Automation define a dimension of authority relationships that Playbook has largely ignored to date—what we characterized as the Levels of Authority dimension (especially as applied to task performance within the Abstraction dimension) above. We also saw that the Parasuraman, et al. model takes a two-dimensional approach to defining a level of automation—a level of authority in Sheridan's sense crossed with a type of task: an element of our abstraction dimension. Therefore, to qualify as an AAA in their terms, we must include this dimension of authority. Also, the set of resources controlled by automation is another useful and intuitive dimension to which an AAA ought to be sensitive. Thus, it will be important and useful to add these two new dimensions to the abstraction dimension that Playbook currently uses. In that way, we will provide a much richer vocabulary with which to understand and control adaptive automation.

## 2.2 The Authority Dimension

Sheridan's original (1987) Levels of Automation describe a dimension of authority—how much can automation do on its own and with what required coordination. This dimension has been largely held constant in our implementations of Playbook. Automation has generally been free, within our Playbook implementations, to develop any plan it could within the constraints stipulated by the human. After developing such a plan, the Playbook would submit it for approval, either prior to, or concurrently with, executing it.

But this has been a simplification on our part. In fact, delegation *inherently* involves some degree of authority—otherwise, there is no savings in workload. It seems a useful heuristic to say that, if a supervisor delegates a task, s/he is delegating the authority to make decisions about how to perform that task, although s/he retains the ability to impose constraints on the decisions that are made and the authority may not be complete—the subordinate may be required to check further.

So we should introduce an explicit, multi-level authority dimension into the Playbook architecture to extend our ability to mirror human-human delegation. Sheridan's levels form a nice spectrum, but we suggest the following distinct categories of authority relationships:

1. <u>*Full*</u>—The supervisor delegates full authority to the subordinate to decide how the task should be executed. The subordinate is charged to 'make it so' and is not required to have any further coordination, permissions or even information exchange with the supervisor.

2. <u>*Inform*</u>—the supervisor delegates full authority to the subordinate as above, but the subordinate is required to inform the supervisor of the decisions and execution actions taken. The supervisor retains no ability to override or approve those steps, however. It might make sense to inform human supervisors only after execution is complete if there is no

override or alteration capability. Thus, this level might be better thought of as "Report" than "Inform".

3. *Override*—the supervisor delegates the task, but expects information (as above), and additionally retains the right to step in and override any or all of the subordinate's plan. The subordinate can proceed with execution until and unless the supervisor intervenes. (There might be an agreed upon lag period which the subordinate must wait for an override from the superior, and there is no guarantee here that the supervisor's intervention might not produce instabilities in the system, or that human intervention can be accomplished within a timely fashion. These are challenges that might make this, or other, levels of authority unfeasible for some systems, and are a challenge that is not directly addressed by either Playbook or by this AAA—though it does point to the increased importance of the types of human-in-the-loop experiments that this architecture will support. That is left to the supervisor to manage ... and s/he might need some assistance in it).

4. *Approval*—the supervisor delegates decision/planning authority about how to accomplish the delegated task, but retains the right to explicitly approve actions before they are taken. The subordinate must submit the plan (or, perhaps, multiple plans) to the supervisor and cannot proceed until one of them has been approved for execution by the automation.

5. *Recommend*—the supervisor partially delegates planning authority alone; s/he retains execution authority. In other words, the supervisor authorizes the subordinate to recommend courses of action, but the supervisor will still make the final decision about which course to execute and who will do the executing. Recommend authority differs from Approval authority primarily in that the automation is doing the executing, after approval, in the latter case, while either the human or the automation (according to the human's choice) may do the execution in Recommend.

6. *Monitor*—the supervisor retains all decision and execution authority, but authorizes the subordinate to maintain awareness and to offer recommendations or critiques. In Monitor authority, the automation does not begin by providing a recommended plan; instead, the human supervisor begins planning and execution activities and the automation operates only by recognizing and offering critiques or improvement suggestions.

7. *None*—the supervisor delegates no authority for how to plan or execute the task. The task is not delegated. "None" delegation is the null or degenerate case and is included only for completeness. In practice, saying that one delegates with "No" or "None" authority is equivalent to not delegating.

Note that by defining the roles in the above authority spectrum as "supervisor" and "subordinate" we have avoided stipulating which role the human and automation play.

Whatever task (at whatever level in the abstraction hierarchy) the supervisor delegates, s/he may in principle delegate with any of these levels of authority. For example, if the supervisor delegates the "Fly-to-Target" task from Figure 4 above, s/he could do it with Full, Inform, Override, Approval, Recommend or Monitor Authority. Furthermore, while the default assumption may be that all the tasks under a parent task in the abstraction hierarchy should have the same level of authority, this doesn't have to be the case. It is entirely possible, for example, that the subordinate have Approval authority for the Achieve Airborne, Full Authority for Fly-to-Target, Recommend authority for Photograph Target and Inform Authority for Destage.

We may want to establish default authorities for specific kinds of tasks based on policy or where only certain authority levels are feasible or may make sense. A simple, feasibility example might be that the human + machine system does not afford the ability to inspect or approve/override specific flight control commands, therefore in that system those tasks would have to go to the automation with Full or Inform authority. Although there is no ability to modify that authority level, our approach still allows us to represent and reason about it. A more significant example, based on policy, might be the standing ROEs that weapons release must always be approved by a human operator—this means that for any task which involves weapons release, that task must be performed with Approval authority (at most) delegated to automation.

One particularly valuable use for default authority levels may be in assigning authority levels to resource usage, as we will see in the next section.

## 2.3 The Aggregation Dimension

Clearly, a system that controls more entities, or more subsystems has more (or a higher level of) automation. Although moving up in the abstraction dimension *usually* entails controlling more entities or subsystems, this is not always the case. In our work with Playbook, we have not explicitly broken out the set of entities or subsystems that automation is controlling as a separate dimension. Instead, we've relied on the simplifying assumption that automation could command any equipment to which it had access to to do anything it wanted, within the constraints imposed by the supervisor.

In the proposed AAA framework, however, we will explicitly represent the resources the subordinate has authority to control as the *Aggregation Dimension* since (as in the usage pioneered by Rasmussen, 1984 and Vicente, 1999) we are now referring to the subcomponents that, in aggregate, comprise the whole system. By contrast, in the abstraction dimension, we are identifying the sub-functions that, collectively, achieve higher-level (more abstract) functions. By explicitly representing the aggregation dimension we make it possible to handle situations in which the subordinate does not have complete access to all equipment all the time, but instead may need to ask permission or report usage.

For the moment, it will suffice to represent the Aggregation dimension by simply referencing the specific entities or subsystems in their natural groupings. For example, the specific UAVs that automation has access to, individually and in flights, squads, companies, etc. Subsystems can be represented similarly: for example, the (potentially multiple) sensor systems on board a given UAV and the specific sensors that comprise the overall sensor system.

As with the Abstraction dimension, aggregated resources must be delegated with a level of authority. For this purpose, we will use the same scheme of levels as for the abstraction dimension. For example, automation may be delegated the right to use a specific UAV (or the sensors on that UAV, or a whole squad of UAVs) with Full, Inform, Override, Approval, Recommend, or Monitor authority. Also, as for the Authority dimension above, we can designate default authorizations for the use of specific resources. For example, a subordinate may be authorized to use a specific UAV organic to company A with Full authority, but authorized to use the Battalion UAV only with Approval authority (i.e., ask permission first).

## 2.4 Summary of the 3D Model of Levels of Automation

Under this model, an act of delegation is a combination of a task, some resources to accomplish that task, and some level of authority to plan and execute that task with those resources. That is, a level of Authority (which may be heterogeneous over subtasks) to perform a task in the Abstraction dimension, combined with a level of Authority to make use of resources in the Aggregation dimension. A Level of Automation is the degree to which authority for performing that task with those resources has been delegated from the supervisor to the subordinate. But since the authority can vary along with the level of the Abstraction and Aggregation dimension, it is important to represent each of these dimensions to characterize the range of possibilities for human-automation interaction. Note that not all of these dimensions may be available or relevant to every system or every interaction, but the model needs to be rich enough to encompass them.

## 3. References

Miller, C. and R. Parasuraman (2003). "Beyond levels of automation: An architecture for more flexible human-automation collaboration," in *Proceedings of the 47th Annual Meeting of the Human Factors and Ergonomics Society*, (Denver, CO), Oct. 2003.

Miller, C. and Parasuraman, R. (in preparation). Designing for Flexible Interaction Between Humans and Automation: Playbook Delegation Interfaces for Supervisory Control. To be submitted to *Human Factors*.

C. Miller, "Delegation architectures: Playbooks & policy for keeping operators in charge," in *Proceedings of the NATO Research and Technology Organization Special Workshop on Uninhabited Military Vehicles*, (Leiden, Holland), June 2003.

Morrison, J. (1993). "The adaptive function allocation for intelligent cockpits (AFAIC) program: Interim research and guidelines for the application of adaptive automation," Technical Report NAWCADWAR-93031-60, Naval Air Warfare Center.

Parasuraman, R., T. Sheridan, and C. D. Wickens, (2000). "A model for types and levels of human interaction with automation," *IEEE Trans. Systems, Man and Cybernetics, vol. 30*, pp. 286–297.

Rasmussen, J., Pejtersen, A. and Goodstein, L. (1994). *Cognitive Systems Engineering*. New York; Wiley.

Sheridan, T. (1987). Supervisory Control. In G. Salvendy, (Ed.) *Handbook of Human Factors*. New York: John Wiley & Sons. 1244-1268.

Sheridan, T. and W. Verplank, (1978). "Human and computer control of underea teleoperators," Technical Report, MIT Man-Machine Systems Laboratory, Cambridge, MA, 1978.

Vicente, K. (1999). *Cognitive work analysis*. Mahwah, NJ; Erlbaum.