Sub-patterns for Human-Autonomy Teaming: Variations on a Delegation Theme

Christopher A. Miller(✉)

Smart Information Flow Technologies (SIFT),
319 First Ave. N., Minneapolis, MN 55401, USA
cmiller@sift.net

Abstract. Building on Shulte et al. [1], we deepen the concept of design patterns for human-autonomy teaming by introducing two distinctions. First, Patterns are composed of a Problem Pattern and a Solution Pattern, both of which should be described and linked so they can be recognized in design. Second, Patterns are hierarchically related in that SubPatterns capture more specific instances of their SuperPattern parents and, thus, can provide more specific design guidance. Both additions are explored within the general concept of supervisory control and specific instances from the Rotorcraft Pilot's Associate program are analyzed using the formalism developed in the paper.

Keywords: Supervisory control · Delegation · Design patterns · Human-autonomy teaming · Workload · Unpredictability · Competency

1 Introduction

The concept of a *design pattern* was introduced by Christopher Alexander—an architect—in his 1977 book *A Pattern Language* [2]. Alexander's concept, and its later adoption by software engineers [3], was meant to link recurrent problems and their similar solutions. Thus, the "pattern" is not just a description of a designed solution, but rather of a recurring process of similar problems with similar solutions. Alexander says "Each pattern describes a problem that occurs over and over again … and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice" [2]. One example in architecture is the recurring problem having rooms be sunny and yet also not overheat. A variety of architectural solutions have been derived—ranging from placing windows on sides of the room that don't get the sun during the hottest hours, to extended eaves, to sky lights, to window tints, to blinds, screens, drapes, etc.

The goal is to create a "pattern language" which not only characterizes the various elements and relationships of the solutions, but also the "conflicting forces" [2] and values in variations of the problem that give rise to preferences for specific solutions. In architecture this might mean characterizing:

- First, the *Solution Patterns* as described (e.g., windows not on room sides with the hot sun, etc.) along with a vocabulary of entities and relationships (e.g., windows, room sides, sun positions, 'hot hours', etc.),

D. Harris (Ed.): EPCE 2016, LNAI 9736, pp. 198–210, 2016.
DOI: 10.1007/978-3-319-40030-3_21

- As well as the attributes and relationships of the problem, the *Problem Patterns*, which the solutions are meant to resolve, along with their entities and relationships (e.g., rooms with sufficient light that are also do not overheat),
- And finally describing the values and priorities by which a solution pattern may be adopted in one instance or by one designer over others.

In the remainder of this paper, I will attempt to apply this "pattern" of patterns to some sub-patterns for the "super-pattern" of human-machine supervisory control [4].

2 Supervisory Control as a Design Pattern

Supervisory control [4] is a dominant paradigm for human-automation interaction (HAI) today, yet it is not the only approach. Is supervisory control a design pattern–"a link between recurrent problems and their similar or identical solutions"? It seems very reasonable to make this claim. In its initial formulation [5], Sheridan and Verplank were generalizing from a "pattern" in human supervisor-subordinate relationships and extending it to HAI. There are clear alternatives. This is important since a design pattern should not be all-encompassing, but rather appropriate in a subset of frequently encountered contexts. Sheridan [4] identified at least three other HAI approaches: direct control, indirect control through an interface (i.e., teleoperation), and computer aided control. There are also HAI relationships which are not supervisory. One characterization comes from Wiig [6] and involves 7 "Cs": Combat, Competition, Control, Cloistering, Coordination, Collaboration and Cooperation. Of these, Supervisory Control fits neatly into only the third category.

The entities and relationships that characterize supervisory control as a design pattern have already been specified in multiple sources, most recently in the paper by Schulte, Donath and Lange in this volume [1]. These consist, in their formalism, of:

- A *Work Process* consisting of a *Work Object* (i.e., a goal or mission) which takes place in an *Environment*, using *Material and Energy Supplies* and *Information*. The Work Process generates *Output* which alters the Environment in some way.
- The Work Process itself is performed by a *Work System* which is composed of *Worker(s)* and *Tool(s).* Workers understand and pursue the Work Object using some degree of initiative and thus can be either *human* or some forms of *advanced automated agents* capable of these higher cognitive functions. Workers alone are able to break down the Work Object into tasks and to assign those tasks to Tools. By contrast, Tools only perform tasks when used or told to do so by the Worker. Though they may be adaptive to the environment, and may also perform higher cognitive functions, they do so only within the roles and duties assigned to the by the Worker(s).
- Note that *Automation*, or *Autonomous Agents*, are a special case that be either Worker or Tool can, depending on their roles and capabilities.
- Two relationships are defined between Worker and Tool elements: a *Hierarchical Relationship* in which tasks are decomposed by one element (Worker only) and

delegated for performance by the Tools, and a *Heterarchical Relationship* in which multiple entities may cooperate to perform the Work Object, presumably in either Worker or Tool roles. Both imply bi-directional information flow but control differs between the two.

Supervisory Control is, then, characterized by a Worker (who may be Human or an Advanced Agent) in a hierarchical relationship with Tools to perform a Work Object. There may be more than one entity in both the Worker and Tool groups and relationships within those subgroups may be hierarchical or heterarchical. Tools themselves may range from "dumb" to highly advanced automation and intelligent agents but do not take initiative in assigning themselves tasks with an understanding of the Work Objective. This relationship may be depicted, using Shulte et al.'s notation, as shown in Fig. 1. Note that the multiple entities in shades of gray are meant to indicate options (i.e., one or more of each may be used) and the dashed-line circles are meant to convey groups of, in this case, possible entities. Thus, this figure connotes the "superpattern" of supervisory contro1[1]

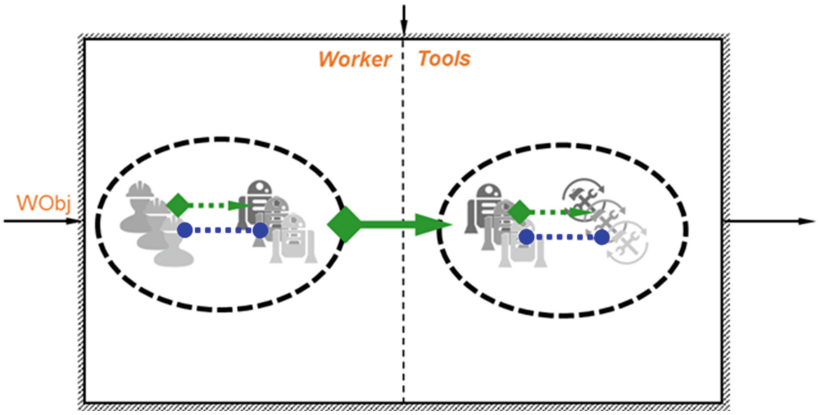

Fig. 1. A proposed "superpattern" for Supervisory Control.

If supervisory control is a design pattern, in the sense of [2], then it must be useful for resolving a set of conflicting forces. But this implies that, insofar as those force sets repeat in patterns of their own, it must admit sub-patterns which permit tradeoffs and differential valuation for different problem sets.

[1] I have avoided placing humans in the tool portion of the figure. While it is clear that there are supervisory relationships between humans and other humans, we might call this "supervision" rather than supervisory control. Schulte would, I think, argue that the human ability to understand the work objective and form intentions about how (and maybe whether) to pursue it, keeps them from fitting the "Tool" in his description.

3 Conflicting Forces: Problem Patterns and Their Solution Patterns

Are there conflicting forces that drive tradeoffs in the use of Supervisory Control and give rise to specific "subpatterns" as instances of it? If so, what is their nature and what aspects of context are pertinent to the application of the parent pattern?

At the highest level of abstraction, the conflicting forces characterizing the problem addressed by supervisory control can be stated as *one or more humans want to accomplish goals in specific ways without incurring the full workload and skills required to achieve the goals themselves*. This may be because they *cannot* incur that work due to physical, mental, locational, etc. reasons, or because they simply don't wish to, or because they are less competent. Thus, this forms the core set of conflicting forces and can be depicted as in Fig. 2. This figure introduces a "superpattern" depicted as a *Problem Pattern* which is solved by an abstract *Solution Pattern*. The Solution

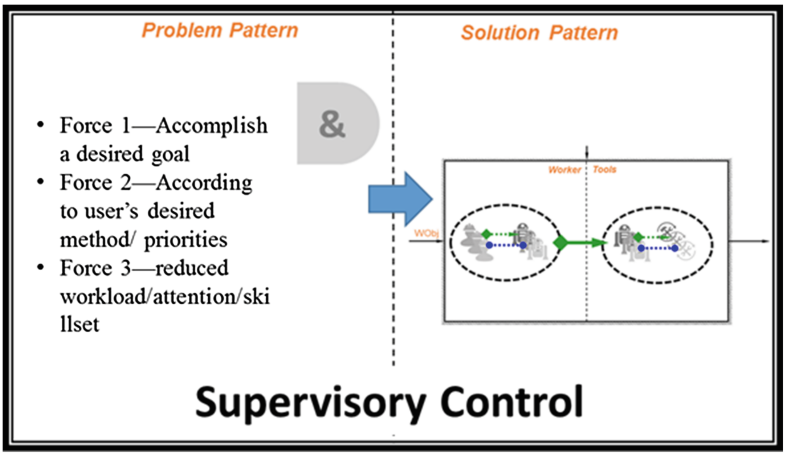

Fig. 2. Proposed Problem Pattern and Solution Pattern that link design decisions to use Supervisory Control.

Pattern is what perhaps more commonly comes to mind as a "design pattern" (cf. [3]), but design power comes from knowing the circumstances in which a solution is appropriate—that is, what problem it is a solution for. Problem Patterns are characterized by conflicting forces [2] and these should be captured in problem description.

I will use the convention of stating problem forces on the left side and the solution pattern on the right to imply that problems with these conflicting forces (i.e., problems of this type) are solved by solutions of this type. The forces themselves will be simply stated for now, with their relationships (notionally, Boolean AND, OR, XOR, NOT) illustrated. In examples below, I will introduce an added notation for prioritization or values which discriminate between those forces.

There is much more to be said about variations in the problem(s) which supervisory control may or may not solve but, as will be seen below, these are appropriately the focus of "sub-patterns"—that is, alternative patterns which are themselves types of supervisory control but which impose additional constraints on the solution pattern and which are intended and "good for" solving different types of problem patterns.

4 SubPatterns of Supervisory Control

Supervisory Control can be seen as a design or Solution Pattern which is good at resolving the conflicting forces that characterize its Problem Pattern. But this gives us little traction by itself, since one always has to apply and interpret Supervisory Control in a more specific setting. If all we know is the high level, comparatively abstract Problem-Solution pattern expressed above, we are left with no way to perform the valuable work of designing specific instances of supervisory control for specific problems. In short, what is needed are "Sub-Patterns" which will be more specifically-defined for more constrained Problem Patterns with more constrained and specific Solution Patterns. If we can characterize these more specific pairings, then we will have come some way toward providing effective design guidance. Each subpattern will have a structure that repeats that shown above—that is, it will have a sub-Problem Pattern and a sub-Solution Pattern. In all cases, the subpatterns will be refinements of, but will still adhere to, the general Supervisory Control pattern stated above.

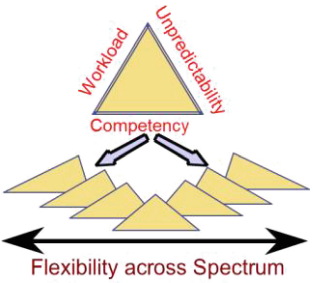

Fig. 3. A tradeoff between three interrelated parameters for human-automation interaction (after [7]).

4.1 Problem Pattern Variations—Finer Grained Conflicting Forces

What finer-grained sets of conflicting forces can we identify within the broad set identified for Supervisory Control? We characterized one set (each embodying or aggregating many subforces) in Miller and Parasuraman (2007) [7]: the *workload* (particularly cognitive workload) the human experiences in attempting to control a system, the *unpredictability* of the system to the human and the *competence* of the overall Workers + Tools system under the expected configuration and environmental conditions. We have drawn this relationship as an adjustable triangle to indicate that there is a fundamental tradeoff between these three dimensions (see Fig. 3).

Performance to a given level of competency can only be achieved through some mix of workload and unpredictability. A user's workload can be reduced by allocating some functions to subordinates (human or automated), but only with increased unpredictability to the user (at least for those functions); conversely, reducing unpredictability by having the user perform functions increases workload. It is sometimes

possible to reduce both workload and unpredictability through better design—corresponding to shortening the height of the triangle.

Another implication is that these approaches are endpoints of a spectrum with many alternatives between, each representing a different tradeoff between workload, unpredictability and competency and each a different mix of human and automation roles. The range of alternatives and exactly how they are implemented may be constrained, but an alternative must be selected. This selection is the process of *designing*, and it is exactly where selecting among solution patterns would be appropriate.

Other dimensions are certainly relevant to supervisory control, but many can be compressed into those three (workload, competency, unpredictability). A list of potential other factors and their relationship is presented in Fig. 4. This list is undoubtedly incomplete and its elements are not orthogonal. Indeed, much work in human factors has addressed their complex relationships for the past several years without complete success. I am attempting a preliminary list of elements which contribute to conflicting forces pertinent to designing supervisory control implementations to enlarge the vocabulary of entities in the Problem Pattern description. From these finer-grained forces we can define more specific sub problems classes. For example: how is supervisory control accomplished with agents/tools of varying competencies? With supervisors of varying competencies? When missions are of high vs. low risk? When the supervisor is not co-located and can't communicate with the subordinate? Etc.

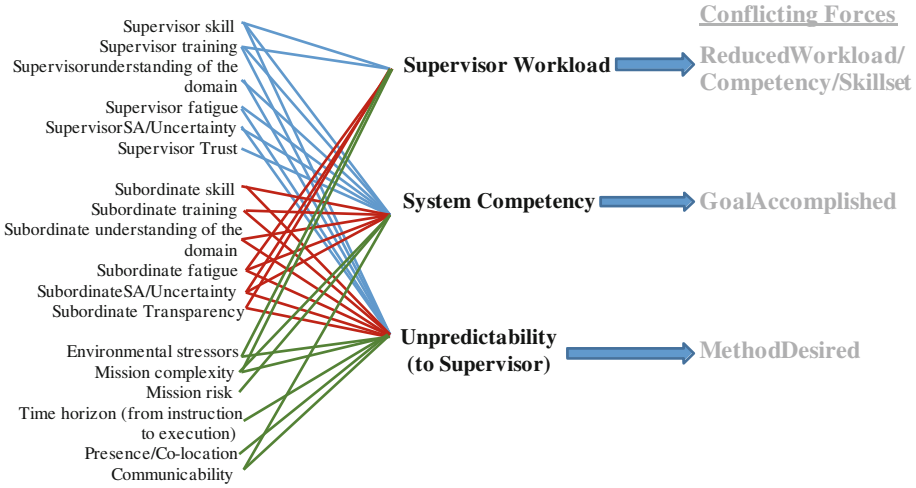

Fig. 4. Some finer-grained dimensions that affect the conflicting forces for the Supervisory Control SuperPattern and its SubPatterns.

4.2 Solution Pattern Variations—Finer Grained Entities and Relationships

Since Sheridan and Verplank defined the "supervisory control" relationship in 1978 [5], Google Scholar says there have been almost 50,000 published articles using that

phrase in their title. In principle, each of these represents or discusses a separate instance of the SuperPattern "Supervisory Control"… but surely there are useful Solution subpatterns within that set—ideas that restrict the broad class of supervisory control to subclasses designed to solve more specific problems. The ability of a pattern representation scheme to account for these variations will be key to its descriptive power. Just as a map may be designed to afford broad international navigation, but not to get a driver from one street address to another, so too a pattern description may need to be augmented or limited to function at different levels of description.

A full categorization of this space is well beyond the scope of this paper, but some dimensions along which supervisory control systems differ can be suggested. Below, I will emphasize the delegation or instruction (cf. [7]) act which is crucial to the Hierarchical Relationship and which transfers control authority from supervisor to subordinate. Important variations in the attributes, number and capabilities of both supervisors and subordinates also exist and should be examined, but have not been yet.

- **Variable**: ***Delegation Method*** — We have previously proposed [8] that delegation can be performed through various mixtures of stated Goals, Plans, Constraints/Stipulations, and Values or Policies. The common or appropriate mixture of these elements for differing conditions may, itself, constitute a sub-pattern within the general delegation pattern.
  - *Relation to Problem Forces*–We know [9] that instruction which emphasizes goals provides the subordinate more autonomy in achieving them while plan-based instruction (sets of actions) reduces unpredictability yet is more brittle. Hence, Problem Patterns prioritizing reduced unpredictability and/or in which upsets and goal failure are rare or acceptable may find plan-based delegation Solution Patterns more acceptable. Similarly, Priority/Value-based delegation seems to be preferred when the supervisor does not know enough to dictate specific goals or plans in a context.
  - *Depiction*–Visually, adapting Schulte's conventions, we can depict Delegation Method as a tag on Hierarchical Relationship as shown in Fig. 5.
- **Variable:** ***Separation/Presence*** *(temporal and physical)*—an act of delegation necessarily precedes performance and is never entirely co-located with it, but the span of time and/or space between the supervisor's instruction and the subordinate's performance may vary from seconds to years, inches to astronomical distances. This distance affects the supervisor's situation awareness (SA) and thus, trust and confidence in the subordinates' understanding. This SA is, ultimately, the more important variable and anything which causes the supervisor to believe that s/he may not know best how to instruct the subordinate in the context of task performance will likely have the effects described below.
  - *Relation to Problem Forces*–Separation affects information flows and frequently mandates more a priori negotiation producing, I hypothesize, a rough bell curve for instruction detail. At very low separation, supervisor intervention ability may drive down the perceived need for substantial instruction. As separation increases, confidence in instructional ability declines, yielding higher levels of delegated autonomy and on delegation at higher task and authority levels or using value-based delegation methods.

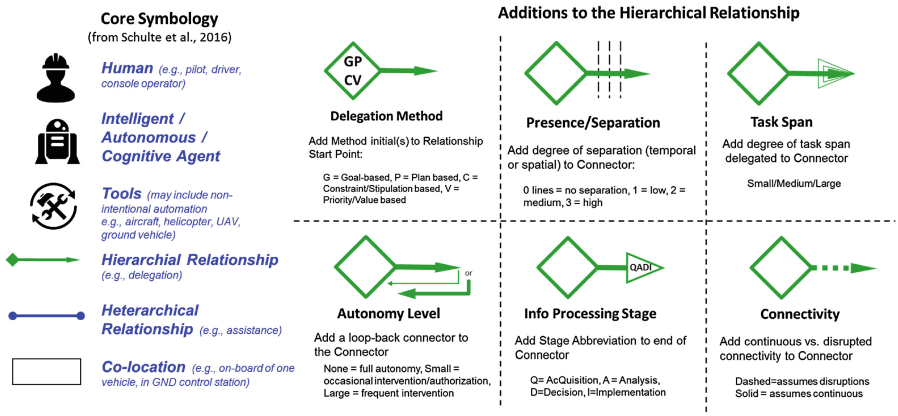

Fig. 5. Proposed depiction additions to enable presentation of additional, finer-grained aspects of the Hierarchical Relationship and, therefore, to permit sub-Patterns to be represented and discussed.

  - *Depiction*— We will depict a relative degree of separation by means of inserted lines on the Hierarchical Relationship connector as in Fig. 5.
- **Variable: *Task Span***—an act of delegation conveys the responsibility to perform a task or function, but that function may be "larger" or "smaller" in terms of the number and variety of subfunctions it includes. The size of this span is related to information flows and the need for a priori negotiation.
  - *Relation to Problem Forces*— Obviously, delegating "larger" tasks to subordinates does a better job (insofar as they are performed adequately) of reducing the supervisor's workload and skill requirements. It can also, though, make performance more unpredictable and recovery from error worse still because it effectively hides errors from the operator [10].
  - *Depiction*— We depict relative degree of task span via the size of the end arrow in the Hierarchical Relationship as shown in Fig. 5.
- **Variable: *Autonomy Level***—a delegation act conveys some responsibility and authority to select and perform alternatives from among the sub-functions which could accomplish the delegated task, but this authority need not be absolute; the agent may need to ask further permission for certain functions or resources. Further, autonomy might not be complete if plans or actions need to be reviewed and approved by the supervisor before they are enacted. Yet another reason is if the Supervisor can intervene and seize control of the tools directly rather than managing them through the subordinate.
  - *Relation to Problem Forces*—As with Task Span, to which it is related, delegating more autonomy to the subordinate reduces workload and skill requirements from the supervisor to the degree that the subordinate is competent (and trusted). But increased autonomy for the subordinate means increased risks that tasks may not be performed exactly as the supervisor wishes—and hence can drive the need for intervention and/or "check-ins" and subsequent authorization.

  - *Depiction*—Autonomy level can be depicted via the presence or absence of a looping arrow on the Hierarchical Relationship along with, if present, its relative size to depict more intervention/authorization requirements from the supervisory (and, hence, less autonomy) as shown in Fig. 5.
- **Variable:** ***Information Processing Stage***—Parasuraman et al. [11] proposed that a function delegated to autonomy be characterized both by the authority level of the delegation act and the information processing stage at which the function occurs: Information Acquisition or Analysis, Decision Selection and Action Implementation. Autonomy can easily occupy these stages in various combinations.
  - *Relation to Problem Forces*—Generally, autonomy which processes information and/or recommends decisions requires greater information interaction with the human supervisor than does autonomy which only executes a human-specified action. This implies greater workload, but more predictability for the supervisor. Of course, full autonomy (see Autonomy Level above) requires autonomy over these stages for performance of the task assigned. As Galster et al. have shown [12], autonomy at the later sequential stages (decision selection and implementation) either requires greater attention to earlier stages from the supervisor or risks "out of the loop" errors and associated time loss as the operator retrieves SA.
  - *Depiction*—We can depict the primary information processing stage(s) in which the subordinate is tasked to act via an abbreviation added to the end arrow of the Hierarchical Relationship—see Fig. 5.
- **Variable:** ***Connectivity***—some subordinate relationships are designed assuming active monitoring and continuous control inputs by the supervisor. These subordinates either fail, or enter a "fail-safe" state when those conditions are violated. Others are designed to operate with occasional sustained communications loss—usually by relying on a more "covering" delegation act that occurred in an earlier time span. The use of additional "layers" of delegation in the form of back up plans or general policies is also common.
  - *Relation to Problem Forces*—Connectivity is clearly related to Autonomy level in that lower levels of autonomy require connectivity. Thus, its impact on the problem's conflicting forces is similar: design for subordinate action during lost connectivity implies reduced workload on the part of the supervisor during non-communicative periods, but this increases unpredictability. The net result is frequently either design for increased autonomy or greater effort to pre-mission planning and authorization stages.
  - *Depiction*— We depict presumption of connectivity via the use of solid vs. dashed lines on the Hierarchical Relationship Connector in Fig. 5.

These are a few "sub-pattern descriptor elements" which, I believe, can be profitably used to depict specific variations under the general heading of the Supervisory Control superpattern. There are clearly many other elements which could be added, and I have admittedly focused only on the hierarchical relationship. Below, I will explore

whether these elements can help us discriminate between Solution and Problem Patterns for different types of Supervisory Control implementations.

5 Some Instances of Supervisory Control Sub-patterns

As a simple example using the concepts developed above, consider the Rotorcraft Pilots' Associate (RPA)—a system created to aid the pilots of an advanced attack/scout rotorcraft [13, 14]. RPA was an intelligent cognitive aiding system which was aware of the intent of the two pilots and of the overall mission, thus it is a Worker in Schulte's sense. The pilots and the Associate worked with a suite of advanced Tools—sensors, communications, weapons and defensive systems, as well as decision aids which processed and integrated results. While the overall relationship between the pilots and the Associate was largely hierarchical and an instance of supervisory control, there were cooperative relationship elements as well. But over the many functions the humans and systems in RPA performed, there were many different interaction styles and relationships. Thus, it is a reasonable source of alternate supervisory control examples. We will examine two such subsystems/functions to illustrate the above representational scheme and the similarities and differences it highlights.

One RPA function was recommending cover locations during an enemy contact. When this event was detected, the Associate determined whether pilots needed and wanted to react. If so, it tasked less intent-aware Tools to analyze terrain and known threats to prepare a set of "cover locations"—locations the aircraft could get to rapidly where it would be safe from fire and visibility. The Associate then determined whether these locations had adequate priority to be presented on the pilot's displays.

Figure 6 presents this relationship pattern, which I have tentatively labelled "Prepare High Criticality Input," as a sub-instance of the general Supervisory Control Pattern (as shown by the expansion from the small image in the upper left). Otherwise, this subPattern uses the same structure as its parent: a Problem Pattern characterized by conflicting forces and a Solution Pattern showing participating elements and their relationships. I have introduced a rough priority to the Problem Pattern forces. Up and down arrows indicate higher or lower priority, and no arrow indicates neutral priority. Chief importance goes to the time critical and high criticality aspects here—failure to get useful cover information quickly enough can result in the loss of the ship and crew.

To the right of this figure is the Solution Pattern. Two humans interact with an intent-aware agent (the Associate) to interpret the WObj for this task. They permit/instruct the Associate to prepare cover location recommendations via pre-mission authorizations. The Associate tasks intelligent, but not intent-aware Tools (decision aids, sensor processors) to develop recommendations when a threat is detected. The relationship from Workers to Tools is Hierarchical, but more detail is provided about it via the enlarged green arrow. The Delegation Method is primarily goal-based (to have cover recommendations) but there are also constraints and values on what would constitute a good location. There is essentially no Separation since all Workers are co-located with the Tools and there is no significant time lag between the request and the task execution period. The Task Span is very narrow– just computing recommendations and presenting them, not executing them, much less additional

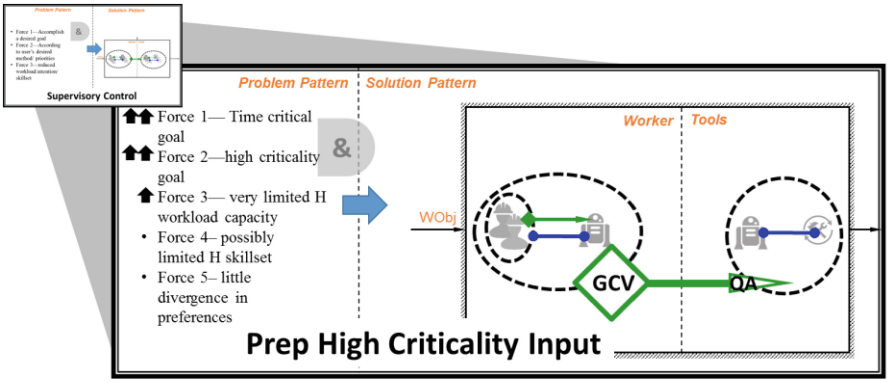

Fig. 6. A Problem and Solution Pattern for a particular subclass of Supervisory Control relationships characterized as Preparing High Criticality Inputs.

deployment or evasion tasks– and falls into the acquisition and analysis Information Processing stages (not decision or implementation). The Autonomy Level, for this task is very high, however, since there is no ability for the Workers to override or interrupt the recommendation of locations. There is a presumption of continuous Connectivity.

Contrast this with preparing a SPOT report upon enemy contact. Such reports are supposed to be sent back to aid in higher echelon and theater SA and coordination. They are important, but their criticality is not nearly as high as deploying to cover upon enemy contact. The conflicting forces on the left side of Fig. 7 reflect this.

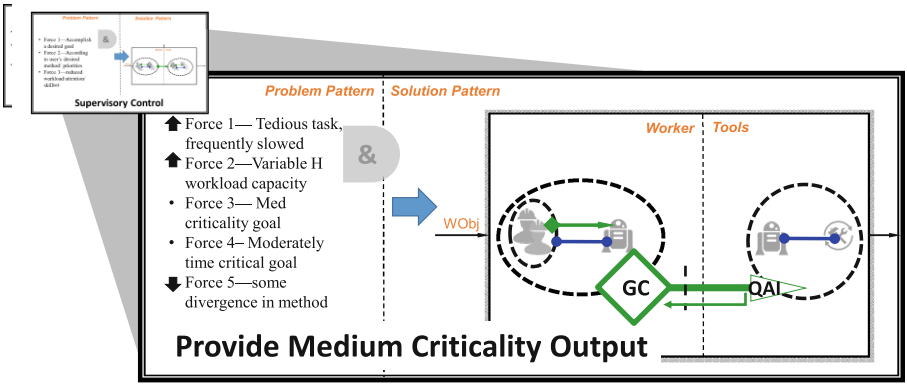

Fig. 7. A Problem and Solution Pattern for another subclass of Supervisory Control relationships characterized as Provide Medium Criticality Output.

The right side of Fig. 7 shows the Solution Pattern. It is similar to Fig. 6 above, as expected for alternate implementations of a superpattern, but there are differences. The delegation method from Workers to Tools is primarily goal-based (to have the report

prepared and sent) but there are also some constraints on when it can be sent without explicit authorization (e.g., user workload levels). There is some temporal separation since although all Workers are co-located with the Tools, authorization for when the report can be sent autonomously is done pre-mission. The Task Span is fairly narrow–just processing information for input to the report, presenting it and sending it if the user agrees (or evaluating workload levels and permission and sending it autonomously if previously authorized), though this is somewhat larger than for coverage locations above. Tasks fall into the Information Acquisition, Analysis stages and Implementation stages. The Autonomy Level, for this task, however, is intermediate since the system is usually supposed to submit its prepared report to the Human for review and editing before sending. There is a presumption of continuous connectivity.

6 Conclusions

I have argued for broadening of the perspective on a design pattern from a strict focus on the solution to, as I believe was Alexander's [2] intent, linking Problem Patterns to Solution Patterns into SuperPatterns and then decomposing them into SubPatterns for more design power. I have explored these relationships in the context of a proposed Supervisory Control "SuperPattern" with both a Problem Pattern (defined by conflicting forces) and a Solution Pattern. I showed how such a pattern can be decomposed into subpatterns with more specific problem and solution patterns. I suggested initial approaches to characterizing conflicting forces for supervisory control subpatterns, as well as specific dimensions (with methods for depicting them) along which supervisory control implementations (especially the Hierarchical relationship) are achieved.

This approach was illustrated via two different functions from the RPA system. I showed that the formalism was capable of identifying differences in both the Problem Pattern and the Solution Pattern for these different functions. I was even able to suggest SubPatterns associated with high criticality inputs vs. medium criticality outputs. This hierarchical approach to linking Problem and Solution patterns and then decomposing them seems both viable and to provide more design power because it identifies more specific characteristics which help to discriminate when (in what circumstances) one instance or implementation approach to supervisory control works vs. others.

Acknowledgments. I am indebted to Axel Schulte, Diana Donath & Doug Lange for the study on which this paper is based, as well as the NATO RTO-HFM-247 Technical Team, particularly Jay Shively, for encouraging discussion on the topic.

References

1. Schulte, A., Donath, D., Lange, D.S.: Design patterns for human-cognitive agent teaming. In: Harris, D. (ed.) EPCE 2016. LNAI, vol. 9736, pp. 231–243. Springer, Heidelberg (2016)
2. Alexander, C.: A Pattern Language: Towns, Buildings, Construction. Oxford University Press, Oxford (1977)

3. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns. Design Patterns. Addison-Wesley, Boston (1995)
4. Sheridan, T.: Supervisory Control. In: Salvendy, G. (ed.) Handbook of Human Factors, pp. 1244–1268. Wiley, New York (1987)
5. Sheridan, T., Verplank, W.: Human and computer control of undersea teleoperators. MIT Man-Machine Systems Laboratory, Cambridge. Technical report (1978)
6. Wiig, K.: Knowledge Management Methods: Practical Approaches to Managing Knowledge. Schema Press, Arlington (1995)
7. Miller, C., Parasuraman, R.: Designing for flexible interaction between humans and automation. Hum. Factors **49**(1), 57–75 (2007)
8. Miller, C.: Delegation for single pilot operation. In: 2014 HCI-Aero ACM, New York (2014)
9. Vicente, K.: Cognitive Work Analysis. Erlbaum, Mahwah (1999)
10. Miller, C., Shaw, T., Emfield, A., Hamell, J., Parasuraman, R., Parasuraman, R., Musliner, D.: Delegating to automation: performance, complacency and bias effects under non-optimal conditions. 2011 HFES **55**(1), 95–99 (2011). SAGE, Los Angeles
11. Parasuraman, R., Sheridan, T., Wickens, C.: A model for types and levels of human interaction with automation. IEEE SMC A Syst. Hum. **30**, 286–297 (2000)
12. Galster S.: An examination of complex human-machine system performance under multiple levels and stages of automation. Dissertation for Catholic University of America (2003)
13. Dornheim, M.: Apache tests power of new cockpit tool. Aviat. Week Space Technol. **151**, 46–49 (1999)
14. Miller, C., Hannen, M.: The rotorcraft pilot's associate: design and evaluation of an intelligent user interface for cockpit information management. Knowl. Based Syst. **12**, 443–456 (1999)