This paper presents a three-phase computational methodology for making informed design decisions when determining the allocation of work and the interaction modes for human-robot teams. The methodology highlights the necessity to consider constraints and dependencies in the work and the work environment as a basis for team design, particularly those dependencies that arise within the dynamics of the team’s collective activities. These constraints and dependencies form natural clusters in the team’s work, which drive the team’s performance and behavior. The proposed methodology employs network visualization and computational simulation of work models to identify dependencies resulting from the interplay of taskwork distributed between teammates, teamwork, and the work environment. Results from these analyses provide insight into not only team efficiency and performance, but also quantified measures of required teamwork, communication, and physical interaction. The paper describes each phase of the methodology in detail and demonstrates each phase with a case study examining the allocation of work in a human-robot team for space operations.

Keywords: computational modeling, function allocation, human robot interaction, methods

INTRODUCTION

In many complex work domains, technology is changing from serving as a tool to acting as embodied agents that work alongside humans. Effectively, technological agents can now be viewed as teammates, although their ability to interact within the team is often limited to a few modes of interaction. This introduction of a limited teammate brings new challenges to the design of sociotechnical systems. Specifically, some of the concerns that need to be addressed in early, conceptual design phases include what is the impact of allocating among teammates the authority to execute different actions, and how robots and humans will need to coordinate work given the limited interaction modes the robots can support.

As noted in this paper’s review of existing methods (detailed in the next section), there is to date no methodology that provides a comprehensive assessment of human-robot teaming in dynamic work environments that estimates not only performance but also the human-robot interaction within multiagent teams consisting of multiple humans working with multiple robots. Decisions determining the allocation of work and modes for human-robot interaction are instead typically made implicitly as a response to the capabilities of the robots, and supporting work analyses are typically made on qualitative, static representations of the work environment. As a result, adverse effects of these design decisions may only be identified later, during field testing or human-in-the-loop experiments, when substantial changes in team design would be prohibitively expensive.

This paper proposes a methodology based on computational work modeling for human-robot teams as a means for making more informed design decisions related to the allocation of work and the interaction modes for human-robot teams. The paper starts by reviewing important
factors in the design of human-robot teams, particularly those identified by work modeling. To address these factors, the methodology consists of three phases, each described in its own section: (1) presimulation analysis, focusing on a static analysis of dependencies and constraints of the work and the agents; (2) simulation, predicting a human-robot team’s behavior through time to identify factors not apparent in static analyses; and (3) postsimulation analysis. Throughout the description of the methodology, a case study illustrates each phase examining an on-orbit maintenance mission on a manned spaceflight mission.

**BACKGROUND**

Salas, Dickinson, Converse, and Tannenbaum (1992) define a team as a collection of (two or more) individuals working together interdependently to achieve a common goal. A team is involved in joint activity defined as “an extended set of actions that are carried out by an ensemble of people who are coordinating with each other” (Klein, Woods, Bradshaw, Hoffman, & Feltovich, 2004, p. 91). With this joint activity, the team members are interdependent and working toward the same goal. Recent perspectives on the design of sociotechnical systems have strived for considering technology as a “team player” (Klein, Woods, et al., 2004). In such a perspective, instead of technology as a tool or prosthesis, it is viewed as possessing “agency,” either amorphous (e.g., software agents) or embodied (robotic agents). Thus, as opposed to “persons” that comprise a team, the term agents will be used to account for both humans and technology that are considered to have agency.

Organizational or team design seeks to determine important aspects of the team, including who owns resources, who takes actions, who uses information, who coordinates with whom, the tasks about which they coordinate, who communicates with whom, who is responsible for what, and who shall provide backup to whom (Szilagyi & Wallace, 1990). Methods such as computational organization theory have established quantitative models that apply to fully trained organizations facing tasks within their operating procedures and knowledge base. However, other important factors such as team response to high workload and time pressure are not as well captured, and conflicting results are found in this literature when examining specific features such as, for example, particular forms of flat versus hierarchical teams or the attributes of good team communication (see review in Schraagen & Rasker, 2003). Thus, the work environment is also increasingly recognized as an important factor in the performance and behavior of teams. To this end, the field of organization design has moved to an open-systems viewpoint that emphasizes the characteristics of the environment and that views a team as a complex set of dynamically intertwined and interconnected elements that are highly responsive to their environment.

The importance of the work environment—that is, the need for an ecological perspective—is also recognized in the literature underlying cognitive work analysis (CWA; Bisantz & Roth, 2007; Rasmussen, Pejtersen, & Goodstein, 1994; Vicente, 1999). From this perspective, the taskwork that the team must perform is driven by dependencies and constraints in the work and the work environment, as captured in a work domain analysis (WDA). Established methods for WDA include an abstraction-decomposition space (ADS), defined by two dimensions. The horizontal dimension reflects a parts-whole decomposition, defining the domain in progressively greater degrees of detail. The vertical dimension as generally applied in CWA qualitatively provides a structural means-end decomposition of the intrinsic constraints and information requirements in the work environment, and can be used to identify the work activities required to regulate inherent dynamics in the work environment. Relative to our intended task of team design, the abstraction hierarchy’s multilevel modeling illustrates how an agent (and the designer when making work allocation decisions) may be able to view work at different levels of abstraction, ranging from detailed descriptions of specific work activities to succinct descriptions of higher-level functions and their relationship to mission goals.

However, we have argued previously that although the representations created by CWA provide a useful visual and/or qualitative description of the work, their qualitative and static presentation limits them in several ways relative to the more predictive capabilities sought here via
computational dynamic simulation (Feigh & Pritchett, 2014; Pritchett, Kim, & Feigh, 2014b). First, although these models have been used to guide the design of potentially innovative or novel designs of sociotechnical systems, they are not able to provide detailed predictions of the dynamics of the work that will result. Second, their effective interpretation requires analysts with significant expertise in CWA, rather than representing the work activities—collective and of individual agents—in terms of dynamic processes and outputs that are of direct concern to (and modeled within) the broader engineering design process, including those applied by the designers of concepts of operation and of robot/machine systems. Finally, the models provided by CWA do not have the same types of explicit mechanism for validation that they are correct and complete as is found with computational models, whose collective representation of the taskwork required of the team can be simulated and compared with known/desired task outcomes. Thus, we will propose in the next section a form of work models that is modified from the qualitative form generated by traditional WDA such that it can be computationally simulated to predict work dynamics.

Similarly, CWA also provides a method for social, organization and cooperation analysis (SOCA), in which at least two aspects of team design are examined: content (referring to the division of the taskwork) and form (referring to structures such as authority and responsibility; Vicente, 1999; see also Naikar, Pearce, Drumm, & Sanderson, 2003). However, this method has been generally used in domains with largely independent agents that act upon different regions within the abstraction hierarchy; for example, Vicente (1999) demonstrated this method in a surgical context in which the surgeon and anesthesiologist acted primarily upon different parts of the work domain, isolating their interaction to a few key factors. Similarly, Jenkins, Stanton, Salmon, Walker, and Young (2008) color-coded the ADS of a military work domain to indicate which areas are associated with the activities of each agent. In contrast, many other human-robot teams instead have teammates performing tightly coordinated activities upon the same parts of the work domain; Pritchett et al. (2014b), for example, noted a case where one pilot is controlling aircraft pitch with elevator while another pilot—or an autopilot—may be controlling speed with throttle, activities which are tightly coupled in the physical reaction of the aircraft. Thus, the methods for SOCA to date serve the purpose of highlighting which aspects of the work domain are (or may be) allocated to specific agents, but then requires further analysis to fully understand how work dynamics will require the agents’ activities to be coordinated, and how this coordination may be performed.

Thus, the coordination of the taskwork also needs to be explicitly modeled. Klein (2001) defines coordination as “the attempt by multiple entities to act in concert in order to achieve a common goal by carrying out a script they all understand” (p. 70), with Klein, Feltovich, Bradshaw, and Woods (2004) describing coordination in close relation to joint activity, in the context of “how the activities of the parties interweave and interact” (p. 145). In some cases, one agent might need to wait until another agent has completed its actions. In other cases, there might be opportunities for parallelization of work. Thus, dependencies and constraints in the taskwork and the work environment drive the team’s behavior in terms of what taskwork can be performed in parallel or where agents might need to wait on each other, and how often and what the agents need to communicate.

In this paper, we refer to coordination activities as teamwork. The form of the teamwork required within the team is driven by two factors. The first factor is driven by the work domain in terms of the need for coordination of the taskwork, given the inherent dynamics within this taskwork as noted above. Recent extensions to CWA to support the design of teams include the work of Burns, Torenvliet, Chalmers, and Scott (2009), which noted the need to analyze the work domain for its implications for coordination; they proposed the comparison of the WDA of three positions within a team to identify shared situation awareness requirements. Similarly, Ashoori and Burns (2010, 2013), noting the historical lack of tools within CWA, proposed extending the WDA and control task analysis method to look at how
agents might coordinate specific activities. These analyses were, however, isolated to specific control tasks, rather than looking at broader coordination patterns across all the team’s required taskwork, some of which may not be captured well by control task analysis. The second factor is driven by the agents in the team in terms of their abilities. One important aspect of their abilities is the degree to which one agent in a team—often the case with machines—is authorized to execute an action, but another agent—such as a human operator—remains vested with the responsibility for the outcome of its actions, a situation termed the “responsibility-authority double-bind” (Woods, 1985). This situation unfortunately establishes a teamwork requirement where the human operator must invest significant effort in monitoring and, as necessary, intervening in the taskwork performed by the machine.

Another important aspect of agent capabilities driving the teamwork is the modes of interaction that they are capable of. For human teams, these interaction modes can be limited by the mechanisms by which they communicate; for an extreme example relevant to the case study later in this paper, Fischer, Mosier, and Orasanu (2013) examined the impact of transmission delays on the behavior of a team comprising astronauts in space and mission controllers on Earth.

Once machine agents are included as teammates, their (usually limited) modes of interaction will also strongly determine the form of the teamwork within the team, particularly the behaviors required of the human teammates. Some methods for examining this teamwork have been extended from the social sciences and studies of teams to human-automation interaction (see, for example, Bass & Pritchett, 2008; Muir, 1994; Muir & Moray, 1996; Woods, 1985; Woods & Hollnagel, 2006). However, these studies have also noted that automation does not have the same teamwork skills that humans naturally have. The automation does not have a sense of responsibility for its outcomes (Sarter & Woods, 1997). Automation does not experience shame or embarrassment, and cannot be assessed for attributes such as loyalty, benevolence, and agreement in values (Lee & See, 2004; Pitt, 2004). When placed outside its intended operating conditions, it may discontinue operations, unlike a human team member who will generally strive for effective performance in unfamiliar circumstances (Feigh & Pritchett, 2014). Examining communication activities in particular, in many teams, agents must be able to anticipate each other’s information needs and provide information at useful, noninterruptive times (Entin & Entin, 2001; Hollenbeck et al., 1995; Hutchins, 1995). The timing of this communication is also important: interleaved tasks can leave one agent waiting on another, and poorly timed communication can disrupt performance. Unfortunately, too often machine agents are “clumsy”: they unduly interrupt its human team members because, whereas humans can implicitly sense information about whether other team members would benefit from an interruption, automation historically cannot (Christoffersen & Woods, 2002).

Given these difficulties, the current state of the art in considering automation as an agent within a team is to examine specific modes of interaction. These modes of interaction can be structured across the distribution of taskwork with associated teamwork functions, such as the “playbook metaphor” proposed by Miller and Parasuraman (2007). Johnson (2014) considered human-robot “interdependence” as a central theme to the design of human-robot teams, wherein by combining of “capacities” of human and robotic agents, multiple team members can contribute to the same taskwork actions.

In a broader sense, several different interaction “modes” are associated with human-robot interaction. First, as noted before, when authority and responsibility are allocated to different agents (e.g., a manager delegates work to a subordinate, but remains accountable for the successful completion of the work), there is an implicit need for the responsible agent to verify the correct execution of the work (Woods, 1985). This required verification demands some form of teamwork through the monitoring of the authorized agent in real-time, or the confirmation of the authorized agent’s work after completion. A second aspect of interaction modes result when robotic agents require a human operator to provide directions to begin and complete actions.
need to (tele)operate a robot throughout the execution of a task, or a human operator might need to provide commands to a robot prior to the robot executing a task, which is often referred to as a command sequencing operator mode (Fong, Zumbado, Currie, Mishkin, & Akin, 2013). This added teamwork creates additional dependencies within the collective work of the team, including precedence relationships (e.g., when an action needs to be confirmed by the responsible agent before the authorized agent is allowed to continue onto its next action) and information dependencies (e.g., where monitoring by the responsible agent demands information about the inputs to, and interim progress of, the authorized agent’s actions). Earlier work on distributed teams in air traffic management highlighted the potentially profound degree of teamwork that can be created by differences in the allocation of authority and responsibility, and the significant information exchange this monitoring can require (Pritchett, Bhattacharyya, & IJtsma, 2016).

These modes of interaction define broad repeatable, predictable instances of teamwork that can be identified during simulation of the work dynamics. The specific times at which these teamwork actions are required will depend on how and when their associated taskwork actions are to be completed; in turn, they may delay or impede the taskwork until both agents are available to interact. Thus, in the next section, we propose incorporating, within computational simulations of a team’s work, the ability to identify where the allocation of taskwork creates the need for teamwork.

This focus on taskwork, and the teamwork necessitated by different team designs, differs from many historic approaches to human-automation function allocation. For example, Roth and Pritchett (2018), in their preface in this journal’s special issue on models of human-automation interaction, noted the historical significance of levels of automation (LOA) as first defined by Sheridan and Verplank (1978) for a single decision-making task. In the same special issue, Smith (2018) and Miller (2018) argued for a richer understanding of the needs of designers beyond “what LOA should I use?” particularly given the many different teamwork tasks that might be allocated across the team for which a one-size-fits-all model does not apply, and given the different interaction modes that might be applied within each. Furthermore, Johnson, Bradshaw, and Feltovich (2018) proposed a design approach that, rather than focusing on the machine functions, explicitly represents the interdependencies of all agents in the team. These ideas expand upon the challenge first raised by Woods and colleagues (Dekker & Woods, 2002; Woods & Hollnagel, 2006) of exploring how to get humans and automation to “work together” rather than focusing exclusively on “who does what.”

In this context, we propose that an effective methodology for assessing designs of multiagent teams, particularly when the teams include machine agents such as robots, should be able to (1) dynamically predict the constraints and dependencies within the taskwork performed by a team and (2) explicitly identify and quantify the teamwork necessitated by the allocation of taskwork within the team. The following sections describe a methodology created with these requirements in mind.

**Presimulation: Modified WDA**

The first phase of the methodology is the modeling of the taskwork that is to be performed by the human-robot team. While inspired by the methods described in the CWA literature, the work model formed here is extended so that it can be computationally simulated. Thus, the first step is creating an ADS for its comprehensive identification of characteristics of the work domain at different levels of abstraction (Bisantz & Roth, 2007; Vicente, 1999). A fundamental requirement of this step is that the ADS capture the constraints on the team’s taskwork as defined by the work environment, with the dynamics within the work domain captured by means-ends relationships between the different levels of abstraction. The higher levels of abstraction are the same as commonly used in WDA, progressing down from the highest “functional purpose” (or “goals”) level, to “abstract functions” (or “priorities and values”), and thence to “generalized functions.”

The work model is extended by adding more requirements to the bottom two levels to expand on the notion of “physical functions” and
resources” beyond what is commonly defined in qualitative, static models. The “resources” level captures two aspects of the work domain: information describing the state of the work domain and environment, and physical resources that are used within the environment.

At the physical functions level, the model defines elemental aspects of taskwork, termed in this paper as actions. Each action must be represented as some activity on the environment and in response to the environment—rather than being defined by task or normative procedural models, each action model is informed by getting information resources about the state of the work domain, uses physical resources, and setting information resources to represent its activity upon the work domain. So that these actions can be passed to agent models during computational simulation, they are defined as being sufficiently distinct that they are each performed within a distinct time window and each performed by a single agent; coordinated work is thus modeled as involving multiple actions performed by different agents.

With these extensions to the work model, dependencies in the taskwork can be captured in several ways. Precedence relationships between activities (e.g., action B needs to follow action A) are defined when one action needs to get or use resources that first need to be established by an earlier action. Likewise, actions may be linked in terms of information (e.g., two actions work on the same information) and physical attributes (e.g., actions taking place in the same locations, or needing the same physical resources). Finally, constraints in the work environment might include how certain resources are only available in a certain location or at a certain time, or how traversing from one location to another takes a certain amount of time.

Once established, the work model can then be examined for groupings or clusters of actions based on similar characteristics. Identifying such clusters can help determining effective candidate work allocations, as these clusters reflect inherent constraints in the work. By identifying what information resources each action requires and manipulates, a directed graph of actions can be determined based on constraints on their sequencing and timing. Likewise, actions can be based on the physical resources they use, again noting constraints on their sequencing and timing. Similarly, actions can be grouped according to the location in which they will be performed, an analysis useful when the translation between locations is costly in some way and thus should be minimized.

Identifying Candidate Work Allocations and the Teamwork They Require

Given the work analysis just noted, candidate work allocations can be identified and then analyzed for their teamwork requirements. Of course, these candidates should consider any limitations on which actions can only be performed by specific agents. Taking a broader view than limited categorizations such as the “Men-are-better-at/Machines-are-better-at” (MABA-MABA list; Fitts et al., 1951), these limitations may arise from agent capabilities (e.g., only specifically trained teammates may perform specific inspection actions), but they may also arise from other aspects of the team design (e.g., only the commander of a space vehicle may be authorized to make mission-critical decisions). Similarly, agent location or the information available to agents may constrain which actions they can perform.

Beyond these basic constraints, we propose identifying candidate work allocations to also examine the work dynamics as captured by inherent clustering of actions in terms of information resources, physical resources, location, and dynamic precedence relationships. For example, in situations where traversal between locations is costly, it is sensible to allocate actions that need to be performed in the same location to a single or a small number of agents. In other situations, it may be useful to allocate actions that use the same physical resources to specific agents.

Each candidate allocation of taskwork will also necessitate teamwork actions to coordinate taskwork activities that are distributed between agents. The form of this teamwork is shaped by the interaction modes that the agents are capable of, notably including human-robot interaction modes. This teamwork can arise from several factors. First, when agents may need to jointly perform actions required to achieve an outcome,
their interdependent work can be analyzed according to levels of interdependency noted by Johnson et al. (2014) and by Johnson (2014), including (1) the executing agent can execute this action without outside support, (2) support by a second agent can help to improve the reliability when executing this action, (3) support by a second agent is required to execute this action. These pairings can be defined on a per-taskwork-action basis, based not only on agent capability, but also on the teamwork implications for task performance. For example, the second level noted here, involving a second agent, is common in safety-critical domains as a backup and form of error-checking.

Second, teamwork is required when there is a mismatch between authority and responsibility for any taskwork action (Feigh & Pritchett, 2014; Woods, 1985). This teamwork can be defined in the form of continuous monitoring by the responsible agent during the authorized agent’s execution of a taskwork action, or in the form of subsequent confirmation of the outcome; the confirmation can be further distinguished according to whether it must be completed before the authorized agent can continue to another action versus allowing the authorized agent to continue immediately and only requiring the responsible agent to confirm within some future time frame. Again, these desired forms of teamwork can be defined for each taskwork action—although initial scoping analysis may benefit from first letting the computational modeling define the implications of broad application of different interaction modes. Furthermore, in choosing the desired teamwork modes, the team designer also needs to account for factors such as whether the agents involved in the teamwork are distributed or co-located, and what communication mechanisms are available (real-time vs. time-delayed, bandwidth, etc.). For example, in the face of significant communication delays from a spacecraft to a mission control room, an Earth-based responsible agent is limited in verifying the robotic operation, as real-time control or monitoring is not physically possible.

**Example: On-Orbit Maintenance**

Communication delays associated with deep space will demand a shift in the operations of manned spaceflight missions. Whereas astronauts now rely heavily on real-time ground support, future astronaut crews will need to operate autonomously from Earth. To alleviate some of the burden that this shift will pose to astronauts, National Aeronautics and Space Administration (NASA) and other space agencies envision the use of robots (Bualat, Barlow, Fong, Provencher, & Smith, 2015; Diftler et al., 2011; Hiltz, Rice, Boyle, & Allison, 2001). Although automation and robots have a long history in space operations, these technologies currently have the role of an aid or a tool for astronauts; often, the robotic systems are teleoperated from mission control on Earth. For future space operations, however, communication delays will not allow for remote teleoperation; instead, more capable robots are envisioned working side-by-side with astronauts.

In this case study, the goal of the mission is to inspect and repair one or more exterior components of the spacecraft that are suspected to need repair or replacement. These components include a panel, a filter, and wiring. Should the inspection find that the condition of a component is below the required specification, there are different procedures for resolving each component, where the panel and the filter both need to be replaced by a new one, and the wiring can be either repaired or replaced.

For this example we will consider a team consisting of an extra-vehicular (EV) astronaut (i.e., outside the spacecraft), an intra-vehicular (IV) astronaut (i.e., inside the spacecraft), a humanoid robot (e.g., Robonaut 2; Diftler et al., 2011), a remote manipulator system (RMS; for example, Canadarm; Hiltz et al., 2001), and a free-flying robot (e.g., Astrobees; Bualat et al., 2015).

In an ADS of the taskwork, shown in Figure 1, the upper-level mission objectives are split into three goals and priorities, followed by six generalized functions at the middle level: dock interaction, locomotion, inspection, panel replacement, filter replacement, and wiring repair. Within these functions, the work is broken down into physical functions and, at the lowest levels, the physical and information resources. The physical functions here were defined as elemental actions, reflecting the
activities each occurring at a distinct time by single agents. More (or less) detailed break-downs can be used; in fact, a less detailed work analysis for this scenario was used in earlier versions of this case study, see IJtsma, Pritchett, Ma, and Feigh (2017).

The physical resources include the tools needed for inspection of panels, filters and wiring, and toolkits for replacement or repair of these components. In addition, there are backup components when a complete replacement is required. For the panels and filters, there are screws of different types which are represented as physical resources. Information resources include attributes of the physical resources (they each have a location, components have a condition, and screws have an attribute specifying whether they are fixed or loose). More abstract information resources include the results of inspections (i.e., a list of the components that need to be replaced or repaired) and the location of agents. Figure 1 also shows a subset of the linkages between actions and physical/information resources.

Within this work model, several clusters of work can be identified. The most obvious clusters follow directly from how the generalized functions are broken down into actions: the actions can be grouped by which generalized function they achieve. This is a grouping down the abstraction decomposition.

A second type of clustering is created by going up in the abstraction decomposition. Here, actions that are linked to the same physical or information resources can be grouped together. Figure 2 shows these clusters in network form. Figure 2a shows the actions (color and no outline, numbered) and physical resources (gray and black outline, letters) as nodes, and the edges represent linkages in that these resources are required to complete the actions. The numbers and letters correspond to actions and resources in Figure 1. The highlighted groups are actions belonging to the same generalized function. Thus, based on there not being any overlapping parts of the shading, one can see that there are no shared resources between the generalized functions.

However, broader clusters associated with information resources do go across functional boundaries, as shown in Figure 2b. Again, the nodes now represent actions (color and no outline, numbered) and information resources (gray and black outline, letters). The highlighted

---

**Figure 1. Abstraction-decomposition space for the on-orbit maintenance scenario.**
groups again are the actions belonging to the same generalized functions. It is clear that many functions share the same information resources. For example, the inspection of a panel and panel replacement share information resources associated with which panel needs to be replaced. Other functions are standalone, such as locomotion which includes fetching. These actions set information resources related to locations of physical resources and agents, but these resources are not required to perform any of the other work.

Other clusters of interest include actions that are performed at the same location, or actions that are linked through precedence relationships. In this scenario, clusters based on location are dependent on how the work evolves: The result of an inspection determines what actions will need to be performed at a specific location. For example, as inspection of the panel returns that a panel needs to be replaced, the inspection action and the panel replacement actions form a cluster. In terms of clusters based on precedence relationships, the inspection actions form a cluster of preparing tools, applying them and storing the tools. Likewise, replacement or repair of components (panels, filters, and wiring) requires sequences of preparation of tools, applying in some manner and storing them again, which could be grouped together into a cluster.

Based on an analysis of these different clusters within the scenario’s work model, three work allocations were identified that each value a different type of coherency (see Figures 3–5). Color-coding and hashing indicate which agent has been allocated the authority for an action. Any blank blocks are shared between the agents, such as traverse and fetching actions, which are performed

---

**Figure 2.** Clusters of actions that are associated with the same resources: (a) Physical resource clusters and (b) informational resource clusters.
by all agents. It is emphasized that in a full application of the methodology, many more work allocations can be defined, based on a broader analysis of the clusters. However, for this demonstration, we limit ourselves to just three work allocations, specifically chosen to highlight different forms of coherency in the work clusters.

The first allocation, in Figure 3, strives for coherency in terms of abstracting the work according to the sequences of work providing generalized functions, such that each high-level function is allocated to a single agent. Thus, the EV astronaut does inspection, the RMS does panel replacement, the humanoid does filter replacement, and the free-flying robot fixes the wiring.

The second allocation, in Figure 4, considers cluster of physical resources and information resources. Whereas, in the first allocation, some information resources are shared between agents, this second allocation makes sure that actions that use the same physical or information resource are allocated to one agent only. For example, all actions that use the same toolset (preparing, applying to remove and fasten screws, and storing the toolset) are allocated to the EV astronaut, reducing the extent that agents’ actions must be coordinated around the handoff of physical resources. For coherency in information resources, the agent that inspects a component is also involved in the replacement or repair of that component, minimizing the extent that team performance is dependent on communication. For example, the free-flying robot inspects the wiring and also does the complete repair of that component, rather than needing to communicate the result of the inspection in a manner sufficient to ensure another agent repairs it appropriately. The panel repair is the only exception: Because the panel is sufficiently large that it needs to be moved by both the RMS and EV astronaut, this activity is distributed over these two agents and the information resource “panel inspection results” is shared between them. Of note here is that by striving for coherency at the lowest level of abstraction, higher levels have become incoherent by breaking up functions and distributing the work over multiple agents.

The third allocation, in Figure 5, explicitly strives for parallel activity, whenever the work allows, to minimize mission duration and all this implies in terms of reduced risk due to EV time outside the vehicle, minimized consumables, and quicker resolution of any failed components. Thus, generalized functions are broken up and parts allocated to two or three different agents. There is still coherency in terms of physical resources (when the goal is parallel
work, it would be inefficient to frequently translate and transfer physical resources between agents). For example, actions that all use panels (get new panel, lift panel, set down panel, dispose of broken panel) have all been allocated to a single agent, but other repair actions have been allocated to other agents such that processes can occur in parallel. In terms of both information resources and generalized functions, this allocation is very incoherent.
It is assumed that neither robots in this scenario can be held accountable for the outcome of an action, so the human agents are allocated responsibility for outcome of the robotic actions. Through this mismatch between authority and responsibility, there is thus a need for additional teamwork actions for the responsible agent to verify the robotic operations, which can be either through real-time monitoring of the robot or through confirmation of the robot’s status after each robotic action. This teamwork can be added to the ADS to identify its dependencies with the taskwork, or, when assuming predefined patterns of teamwork and its dependencies, can be automatically accounted for as described in the next section (the latter option being useful when the analysis is aimed at quickly exploring different work allocations and their teamwork requirements). The allocation of responsibility in the three work allocations includes the EV astronaut being responsible for the RMS’s operation, the IV astronaut being responsible for the Humanoid’s operation, and mission control center (MCC) responsible for the free-flying robot. Both EV and IV have real-time communication links with the robots; therefore, they can monitor these robots. MCC is at a slight communication delay (10 s), sufficiently large that the MCC can only confirm each of the free-flying robot’s actions.

In addition, the current state of space robotic technology requires human input. For the three allocations, we assume a fixed set of interaction modes: The RMS is manually operated, and the humanoid and free-flying robot work through command sequencing. To demonstrate the effects of different interaction modes, the first allocation is also tested with the RMS being commanded and confirmed by the EV astronaut (from here on denoted as Work Allocation 1b). The agents operating the robots are the humans responsible for the outcomes.

**SIMULATION: IDENTIFYING EMERGENT BEHAVIORS WITHIN THE TEAM**

The second phase of the methodology consists of a dynamic analysis through computational simulation of the work model created in the first phase. This phase serves to identify the emergent behavior created by interactions between the taskwork in the work model, the teamwork associated with a specific allocation of work among the agents, and the work environment, accounting for the dependencies and constraints that affect the timing of the work.

For this dynamic analysis, the actions identified in the previous phase, and their relationship to information and physical resources in the work environment, are modeled in a computational form. The simulation framework Work Models that Compute (WMC) provides the basis for the modeling and simulation to perform dynamic analysis of human-robot teams. A brief description of the simulation framework is provided here. For a detailed discussion, the reader is referred to Pritchett, Feigh, Kim, and Kannan (2014) and IJtsma, Ma, Feigh, and Pritchett (2019).

WMC relies on much the same modeling constructs discussed in the previous section: actions, physical and information resources, and agents. The computational form of the actions each specifies which information resources an action gets (reading their value), some elemental calculation or conditional statements upon these representations of environment state, and which information resources the action sets (changing their value). Likewise, the model defines which physical resources an action needs to perform the work. For example, the code for an inspection action gets the information resource “panel condition” to retrieve information on the panel’s condition. The action model contains conditional statements to discriminate between two potential environmental conditions: (1) the panel is broken and needs repair, in which case this action sets the information resource “panel inspection results,” or (2) the panel is in fine condition, and no further action is needed. Furthermore, this action uses the physical resource “panel inspection tool,” and thus during this action this physical resource is unavailable to other actions.

At their most basic, action models may employ very simple representations of how they use the resources they get to perform their work, that is, set resources. Where the actions operate on dynamic environments, their models can also include more involved internal representations
of dynamics. For example, a simple action model moving a robotic arm might just set the associated information resource to the final position; however, where the dynamics of the arm might be continuously controlled or monitored by a human, or might obstruct the work of others as the arm passes through a location, the action model might instead include a dynamic model of how its position and velocity varies through time in response to command inputs. For example, in earlier work in the air traffic management domain, aircraft dynamics were simulated as pilots interacted with their own and other aircraft in the airspace (Pritchett et al., 2016).

Some dependencies within the collective work of the team arise out of the taskwork once it is executed within a specific scenario and work environment. For example, during simulations, the simulation framework recognizes when an agent’s allocated actions require the agent to traverse from one location to another, or fetch a physical resource required for an upcoming action. In these cases, the simulation will automatically engender “traversal” and “fetch” actions to describe additional actions arising within the immediate situation. The simulation can also be scripted to insert faults or trigger failures by any agent—including the robots—to perform their actions correctly, where these events then require detection and resolution within the team.

Other dependencies arise due to the candidate work allocations and the desired teamwork modes, which are specified as inputs to a simulation run. These definitions link authority and responsibility for each of the action models to their associated agent models for that run. Based on any requirements for teamwork, WMC automatically engenders teamwork actions needed to monitor or confirm actions in the case of authority-responsibility mismatches and needed to command or jointly conduct interdependent activities. Thus, the teamwork definitions from the first phase of the methodology are now translated into teamwork actions with different dependencies with the taskwork. Figure 6 shows teamwork actions that are currently supported in WMC and the temporal dependencies with their parent action. For example, monitoring and manual teleoperation of a robot (shown on the left) need to occur in parallel with the parent action. Command sequencing and confirmation (shown in the middle figure), however, occur prior and after the parent action, respectively. Any combination of these teamwork actions is possible, as supported by the interaction modes described in the previous section.

These teamwork actions themselves can demand all or some of the information and physical resources used by the taskwork actions they support. Furthermore, the teamwork may also require new information resources to be created. For example, when an action performed by a robot needs command inputs, the simulation engenders both a command action and a command (information) resource. Whenever the robot performs its action, the simulation will first require a human agent to execute the command action, which sets the command resource. This command resource is then gotten by the robot’s action. Thus, the teamwork actions, and their linkages to information and physical resources, add to the dependencies within the team’s collective work as it emerges during each simulation run.

The automatically engendered teamwork actions are, by default, fairly simple. Although they can be extended by the modeler based on the goal of the analysis, the case study shown next will demonstrate how these basic models—
automatically engendered within the simulation and thus without requiring extensive computational model development of each—capture many important dependencies within the collective work of the team.

When the simulation is started, the simulation’s core action list contains the list of taskwork actions that is sorted by their respective predicted update times. The simulation proceeds by stepping through this list. As an action moves to the top of the action list, it is examined as to whether the conditions in the work environment do trigger this action to be performed at this time as predicted; the action may be moved to a different time in response to conditions in the environment. Once an action is required, it is examined for whether it needs to engender other taskwork and teamwork actions as just noted. Furthermore, additional actions may be needed due to constraints within the work. These include precedence relationships, in which one action needs to be completed before another can be started. For example, for a robot that requires control inputs, its assigned taskwork action can be started only once the information resource “commands” has been given a value, which itself requires a command action first be completed by the proper human agent. Likewise, an action may need to be delayed due to unavailability of physical resources (e.g., a different action is currently using a required tool). All of these considerations can add new actions to the action list and change the sequence and timing of the actions’ possible update time. Thus, the action list changes dynamically as the simulation is run.

Once its dependences and constraints are met, the action at the top of the action list can be executed. For this, the simulation framework passes it to the agent model allocated authority for its execution. A perfect agent model will complete all actions, starting immediately upon their receipt and lasting for an expected duration. Easy extensions to the agents’ models can further capture the dynamics they bring to the team collective behavior. For example, an agent may be flagged as only being able to perform one action (or a given number of actions) at the same time; in this case, an agent model can further delay when it executes its assigned actions according to which it prioritizes to do first.

These many potential dependencies and constraints inherent to a team’s collective work are difficult to reliably predict without simulation. Existing methods for allocation of work, however, most often only use a static analysis, risking that emergent effects will only become apparent in detailed testing once technologies, team compositions, and procedures are sufficiently established that they are difficult to change. Thus, the benefit of using computational simulation is that the designer can identify how these dependencies and constraints interact to create emergent effects in the various metrics of interest, providing objective, quantitative data on how these might affect the performance of a human-robot team.

To provide this insight, the simulation engine logs when each action is performed, the transfer of information resources (as a measure for required communication), and exchange of physical resources (as a measure for required physical interaction), as actions executed by different agents share information and physical resources. From these basics, one can compute a wide variety of secondary metrics, which will be part of the postsimulation analysis discussed in the next section.

Example: On-Orbit Maintenance

Continuing the discussion of the on-orbit maintenance scenario, the work model shown in the ADS earlier in Figure 1 was implemented as a collection of action and resource models within the WMC framework. In this case, five agents in the team are also represented by agent models in WMC. The three work allocations in Figures 3–5 are each inputs to different simulation runs, defining how the actions and agents need to be linked through authority and responsibility, and, additionally, what teamwork modes are to be assessed.

The timeline of each agent’s actions in this scenario with Work Allocation 3 are shown in Figure 7. The grayscale actions are taskwork actions, as originally identified in the ADS. The actions in color (lower right of the legend) were automatically engendered and recorded by the simulation engine based on the work allocation and desired teamwork modes. These results reveal that there are many instances in which agents need to wait on each other to complete
their work. There are a couple of interesting work patterns that are worth highlighting. For example, some taskwork actions for the EV astronaut need to be delayed because the EV astronaut also needs to teleoperate and monitor the RMS in handling the panels. In addition, there are several instances in which physical resources need to be fetched by the free-flying robot, and agents need to wait for these resources to arrive before they can continue their work. Finally, dependencies through teamwork for commanding and confirming robotic actions result in many interdependencies between the MCC and the free-flying robot, where these two agents need to continuously coordinate the timing of their work. These are examples of how the constraints and dependencies, in both the taskwork and teamwork, intertwine in emergent ways that would not be foreseeable from a static analysis.

**POSTSIMULATION: ANALYSIS**

The results from modeling and simulation create the timelines of when the different actions can be performed given the constraints and dependencies in the work, the work environment, and the agents. Following the simulation, these raw data can be further analyzed to create more specific metrics, expanding on initial discussions of metrics of function allocation provided by Pritchett, Kim, and Feigh (2014a). Analysis can compare across allocations of work and between agents. Each metric can be analyzed in the aggregate (over the entire scenario) or in detail, considering specific timing of actions to analyze how emergent effects impact the work distributions of individual agents. This section discusses, first, the comparison of different work allocations as the focus of this methodology. More detailed analysis of individual allocations of work, and even of individual agents, can then focus on better understanding why a particular allocation of work shows certain emergent behavior.

In comparing allocations of work, the post-simulation analysis considers the team’s efficiency through calculation of performance and taskload metrics, but then also explicitly accounts for metrics of coordination, communication, and interaction. The combined effort provides a detailed analysis of factors that are important to defining effective human-robot teams.

Performance metrics include mission duration, idle time, busy time, failure time, and taskwork time. These performance metrics provide insight into the efficiency with which the human-robot team operates and can be considered both in the aggregate as well as per agent and for specific parts of the mission. They can be calculated directly from the output of the simulations.

Taskload is measured as the number of actions that need to be executed per work allocation, per subteam, or per agent, and might provide a coarse
assessment of the workload requirements for agents (although it should be noted that workload is dependent on many more factors). Analysis of taskload metrics includes consideration of total taskload, the ratios of different task types such as regular task actions, teamwork actions, failure actions, traversal actions, and fetching actions, as well as taskload through time for estimating taskload spikes and drops (potential for over- or underloading agents).

Comparison of the number of teamwork actions between work allocations provides insight into the amount of coordination that each requires, particularly for those types of teamwork actions that need to be synchronized with specific taskwork (e.g., monitoring that needs to occur at the same time as its corresponding taskwork action). Likewise, fundamental requirements for when and how much information must be communicated can be compared by measures of information exchange needed between agents (i.e., when one agent’s actions set a resource that another agent’s actions get). These results can then be used to assess the feasibility or desirability of particular work allocations, especially in work domains where there might be concerns with limited bandwidths or communication delays.

Similarly, the amount of physical interaction that is required can be measured through the number of physical resource handovers that are inherent to the work model and allocation. Each resource exchange implies a need for physical interaction between two team members as they hand over the resource. In addition, one can consider the number of joint actions that are required as a metric for physical interaction, especially when the need for joint activity arises from physical attributes of the work or the agents (e.g., this panel cannot be lifted by one agent alone, and therefore the lifting requires two agents to work together jointly).

The simulation results also reveal how dependencies in the work and interactions between agents manifest themselves over time, and provide an elaborate picture that identifies how teamwork and work dependencies add to the collective network of interdependencies and constraints within the team’s collective work. This understanding can contribute not just to work allocation decisions, but also to other design decisions benefiting from improved predictions of the work that will actually be required. Exemplar visualizations of these effects are given in the next subsection’s discussion of the on-orbit maintenance case study.

**Example: On-Orbit Maintenance**

Figure 8 compares the three work allocations in terms of busy time (Figure 8a) and idle time (Figure 8b), with each agent’s contribution shown separately. Mission durations ranged from Work Allocation 1a, which was created for coherency in the generalized functions, has the longest mission duration of 1 hr and 26 min. Considering idle versus busy time, the long mission duration seems to be attributable to the agents idling relatively long in Work Allocation 1a, which would be expected for its lack of opportunities for parallel work. A detailed inspection of the timeline additionally revealed that because the EV astronaut is required to control and monitor the Humanoid robot while it is replacing a panel, the subsequent inspections are delayed, which results in long idle times for the other agents and a longer mission duration. With the different interaction mode wherein the EV astronaut commands and confirms the Humanoid’s operation (Work Allocation 1b), the EV astronaut is free to continue inspection, resulting in a mission duration of only 1 hr and 3 min.

Work Allocation 2, striving for coherency in information and physical resources, has a mission duration of 1 hr and 21 min, with both busy and idle times slightly shorter compared with Work Allocation 1a. Work Allocation 3, created to enable parallel work, has—as one would expect—the best performance in terms of temporal metrics, with a mission duration of 1 hr and 11 min, and the shortest idle time.

Figure 9 shows the total taskload, split into taskwork and different forms of teamwork actions. All work allocations look fairly similar, with Work Allocation 3 showing the highest number of required traversal actions. This high traversal load can be attributed to incoherency in geographical location, with the Humanoid needing to go back and forth between different inspection and repair sites, as well to incoherency in physical resources, which often need to be shared between agents and
therefore fetched from different locations. Work Allocation 2 shows the lowest taskload, due to lower monitoring and control demands. Work Allocations 1a and 1b are distinctly different from each other in that 1a requires more monitoring and control, and fewer command and confirmation actions. Thus, Work Allocation 1a would require more support for real-time datalinks and synchronous operations as opposed to Work Allocation 1b, which relies more on interaction modes that involve asynchronous operations.

Figure 10a shows the required exchange of information resources between agents as a measure for the required communication. Information requirements at the same time and between the same two agents are represented as one information exchange in this figure. Figure 10b shows the required transfer of physical resources, as a measure for the required physical interaction between agents. Work Allocation 3 shows the highest number of required information exchanges, which is due to its incoherency in information resources. Likewise, it shows the highest number of physical resource handovers. Work Allocation 2 shows a low number of information exchange, which is to be expected given that it was specifically created with coherency in resources in mind. Work Allocation 1a has the lowest number of information exchanges. These three work allocations have relatively high control and monitoring load with associated information requirements. Indeed, looking at Work Allocation 1b, which uses command and confirmation instead of control and monitoring for the Humanoid robot, there is a 13% reduction in information requirements.

Finally, Figure 11 demonstrates a network visualization of the information requirements between agent pairs for Work Allocation 3.
The nodes represent the agents, and the edges represent the number of information requirements between the agents (the thicker the line, the more information requirements). Here, one can identify subteams within the larger team: For example, the IV astronaut and the Humanoid seem to be relatively interconnected. The RMS, on the contrary, has connections with all other agents, but the thin lines indicate that information exchange is infrequent and mostly one-way (toward the RMS). Likewise, MCC is only pushing information to other agents, never pulling or receiving any information.

Other network graph representations can be used to also include shared information and physical resources, which can then be used to identify where coherency in resources can be further increased or decreased.

**DISCUSSION**

The methodology presented in this paper is targeted at early design stages, when the major design regarding the allocation of work and interaction modes can be made with the least cost and to the greatest effect, and when computational simulations of activity can quickly examine a range of options before committing to specific design decisions. For human-robot teams, particularly in novel operations not captured reliably by current-day subject matter experts, there is a need for objective insight into and identification of the emergent behavior of design decisions.

With this application, the methodology can potentially be used at different scales of analysis ranging from testing a few candidate team designs to doing large-scale exploration of design trade-spaces. Although this paper’s case study was limited to three work allocations (each chosen to represent particular team design constructions) and two interaction modes, the design methodology is particularly powerful for the latter type of analysis, and the computational simulation can easily be automated and streamlined to be able to test several thousand team designs within a couple of seconds. Indeed, although the example given in this paper applied the method to two aspects of team design, namely, work allocation and interaction modes between...
humans and robots, this analysis can extend to also examining team composition in terms of the number of agents and the actions they each are capable of (through training and technology design), and other team design factors such as the information provided to each agent and the mechanisms for communication between agents.

As progress is made in the design process of human-robot teams, the models can be iteratively refined to account for more detailed design considerations. In this approach, one could start with a fairly granular work model, which can be broken down further in later design phases, depending on where the analysis warrants more detailed consideration. For example, one can start with a single high-level inspection action, but as the analysis progresses and it is realized that there are different ways of inspecting an object, perhaps distributed over multiple team members, this action model can be converted into several more detailed action models for another iteration within the static modeling, simulation, and analysis. In many cases, such as the spaceflight operations case study in this paper, detailed dynamic models of particular technologies and operations will have been created for other aspects of the design process, and can also be incorporated into the computational simulations.

Likewise, different types of agent models can be used to account for performance aspects of agents. Here, it is relevant to note that there are benefits to initially starting with simple performance models for early analyses, as any emergent effects that are identified in the simulation are the direct result of the work dependencies and constraints, unconfounded by assumptions made about agent performance. However, in more detailed design phases, there can be more interest in the emergent effects of performance variability in agents, and in these cases, more detailed performance models can be implemented into the simulation framework.

Ultimately, final test and evaluation of a team design will be tested in human-in-the-loop prototyping and evaluation. The methodology in this paper is targeted at the earlier stages in design, serving to identify issues inherent to the work dynamics. In many cases, for example, our experience has been that the computational simulation identifies concerns inherent in the domain, even when the team’s activity is executed perfectly: Any subsequent variability in agent performance cannot address such concerns without work-arounds or further sacrifices in outcome. Thus, the methodology proposed here is meant to streamline out big issues in the team design so that the later design stages can be confidently targeted to those issues manifesting with humans in the loop.

CONCLUSION

This paper presented a methodology for the allocation of work and selection of interaction modes for human-robot teams. Design decisions related to these aspects of human-robot teaming are now often made implicitly in the development of robotic technology, when the robot is fielded with certain capabilities (and assumptions for human interaction) which the human team members must work around. Instead, this paper demonstrates that these decisions can be considered explicitly in the conceptual design phases of multiagent human-robot teams.

Three main attributes of the methodology were highlighted. First, the methodology is based on the analysis of dependencies and constraints in the work and the work environment. These factors drive the collective behavior of human-robot teams and therefore provide a constant basis for design. Second, dependencies in work translate into requirements for coordination between team members, which is explicitly taken into account in the methodology as teamwork. Third, computational work models provide insight beyond traditional static analysis of human-robot teams, by allowing for the evaluation of emergent behavior originating from the complex interplay of work, work environment, and agents.

ACKNOWLEDGMENTS

The work presented in this paper is sponsored by the National Aeronautics and Space Administration (NASA) Human Research Program with Jessica Marquez serving as Technical Monitor under Grant No. NNJ15ZSA001N. The authors would also like to thank the other Work Models that Compute (WMC) developers.
REFERENCES

Ashoori, M., & Burns, C. (2010, September). Reinventing the wheel: Control task analysis for collaboration. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 54, 274–278.

Ashoori, M., & Burns, C. (2013). Team cognitive work analysis: Structure and control tasks. Journal of Cognitive Engineering and Decision Making, 7, 123–140.

Bass, E. J., & Pritchett, A. R. (2008). Human-automated judgment learning: A methodology to investigate human interaction with automated judges. IEEE Transactions on Systems, Man and Cybernetics. Part A: Systems and Humans, 38, 759–776.

Bisantz, A., & Roth, E. (2007). Analysis of cognitive work. Reviews of Human Factors and Ergonomics, 3, 1–43.

Bualat, M., Barlow, J., Fong, T., Provencher, C., & Smith, T. (2015, August–September). Astrobot: Developing a free flying robot for the international space station. Paper presented at AIAA SPACE 2015 Conference and Exposition, Pasadena, CA.

Burns, C., Torenvliet, G., Chalmers, B., & Scott, S. (2009). Work domain analysis for establishing collaborative work requirements. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 53, 314–318.

Christoffersen, K., & Woods, D. D. (2002). How to make automated systems team players. In E. Salas (Ed.), Advances in human performance and cognitive engineering research (Vol. 2, pp. 1–12). New York, NY: Elsevier. doi:10.1016/S1479-3601(02)2002-7

Dekker, S., & Woods, D. D. (2002). MABA-MABA or abracadabra? Progress on human-automation co-ordination. Cognition, Technology & Work, 4, 240–244.

Diffler, M. A., Mehlung, J. S., Abdallah, M. E., Radford, N. A., Bridgewater, L. B., Sanders, A. M., Ambrose, R. O. (2011, May). Robonaut 2—The first humanoid robot in space. Paper presented at IEEE International Conference on Robotics and Automation, Shanghai, China.

Entin, E. E., & Entin, E. B. (2001, June). Measures for evaluation of team process and performance in experiments and exercises. Paper presented at 6th International Command and Control Research and Technology Symposium, Annapolis, MD.

Feigh, K. M., & Pritchett, A. R. (2014). Requirements for effective function allocation: A critical review. Journal of Cognitive Engineering and Decision Making, 8, 23–32.

Fischer, U., Mosier, K., & Orasanu, J. (2013). The impact of transmission delays on mission control-space crew communications. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 57, 1372–1376.

Fitts, P. M., Viteles, M. S., Barr, N. L., Brimhall, D. R., Finch, G., Gardner, E., . . . Stevens, S. S. (1951). Human engineering for an effective air-navigation and traffic-control system. Oxford, UK: National Research Council.

Fong, T., Zumbado, J. R., Currie, N., Mishkin, A., & Akin, D. L. (2013). Space telerobotics: Unique challenges to human-robot collaboration in space. Reviews of Human Factors and Ergonomics, 9, 6–56.

Hiltz, M., Rice, C., Boyle, K., & Allison, R. (2001, June). Canadarm: 20 years of mission success through adaptation. Paper presented at International Symposium on Artificial Intelligence, Montreal, Québec, Canada.

Hollenbeck, J. R., Ilgen, D. R., Sego, D. J., Hedlund, J., Major, D. A., & Phillips, J. (1995). Multilevel theory of team decision making: Decision performance in teams incorporating distributed expertise. Journal of Applied Psychology, 80, 292–316.

Hutchins, E. (1995). Cognition in the wild. Cambridge, MA: The MIT Press.

Iltsma, M., Ma, L. M., Feigh, K. M., & Pritchett, A. R. (2019). Analysis of work dynamics for objective function allocation in manned spaceflight operations. In M. A. Vidulich & P. S. Tsang (Eds.), Advances in aviation psychology (Vol. 3, pp. 99–118). Boca Raton, FL: CRC Press.

Iltsma, M., Pritchett, A. R., Ma, L. M., & Feigh, K. M. (2017, July). Modeling human-robot interaction to inform function allocation in manned spaceflight operations. Paper presented at Robotics: Science and Systems, Workshop: Bridging the Gap in Space Robotics, Boston, MA.

Jenkins, D. P., Stanton, N. A., Salmon, P. M., Walker, G. H., & Young, M. S. (2008). Using cognitive work analysis to explore activity allocation within military domains. Ergonomics, 51, 798–815.

Johnson, M. (2014). Coactive design—Designing support for interdependence in human-robot teamwork. Delft, The Netherlands: Delft University of Technology.

Johnson, M., Bradshaw, J. M., & Feltonvich, P. J. (2018). Tomorrow’s human–machine design tools: From levels of automation to interdependencies. Journal of Cognitive Engineering and Decision Making, 12, 77–82.

Johnson, M., Bradshaw, J. M., Feltonvich, P. J., Jonker, C. M., Van Riemsdijk, M. B., & Sierhuis, M. (2014). Coactive design: Designing support for interdependence in joint activity. Journal of Human-Robot Interaction, 3, 43–69.

Klein, G. (2001). Features of team coordination. In M. McNeese, M. R. Endsley, & E. Salas (Eds.), New trends in cooperative activities (pp. 68–95). Santa Monica, CA: HFES.

Klein, G., Feltonvich, P. J., Bradshaw, J. M., & Woods, D. D. (2004). Common ground and coordination in joint activity. In W. R. Rouse & K. B. Boff (Eds.), Organizational simulation (pp. 139–184). New York, NY: Wiley.

Klein, G., Woods, D. D., Bradshaw, J. M., Hoffman, R. R., & Feltonvich, P. J. (2004). Ten challenges for making automation a “team player” in joint-human-agent activity. IEEE Intelligent Systems, 19, 91–95.

Lee, J. D., & See, K. A. (2004). Trust in automation: Designing for appropriate reliance. Human Factors, 46, 50–58.

Miller, C. A. (2018). The risks of discretization: What is lost in (even good) levels-of-automation schemes. Journal of Cognitive Engineering and Decision Making, 12, 74–76.

Miller, C. A., & Parasaruman, R. (2007). Designing for flexible interaction between humans and automation: Delegation interfaces for supervisory control. Human Factors, 49, 57–75.

Muir, B. (1994). Trust in automation: Part I. Theoretical issues in the study of trust and human intervention in automated systems. Ergonomics, 37, 1905–1922.

Muir, B., & Moray, N. (1996). Trust in automation. Part II. Experimental studies of trust and human intervention in a process control simulation. Ergonomics, 39, 429–460.

Naikar, N., Pearce, B., Drum, D., & Sanderson, P. M. (2003). Designing teams for first-of-a-kind, complex systems using the initial phases of cognitive work analysis: Case study. Human Factors, 45, 202–217.

Pitt, J. (2004). Digital blush: Towards shame and embarrassment in multi-agent information trading applications. Cognition, Technology & Work, 6, 23–36.

Pritchett, A. R., Bhattacharyya, R. P., & Iltsma, M. (2016). Computational assessment of authority and responsibility in air traffic concepts of operation. Journal of Air Transportation, 24, 93–102.

Pritchett, A. R., Feigh, K. M., Kim, S. Y., & Kannan, S. K. (2014). Work models that compute to describe multiagent concepts of
Martijn IJtsma received the BSc and MSc degrees in aerospace engineering from the Delft University of Technology, the Netherlands, and the PhD degree in aerospace engineering from the Georgia Institute of Technology, Atlanta, USA. His research interests include human-robot/automation teaming and the design of automation support for decision-making in complex work environments. His current research focuses on the use of computational simulation to study work strategies in human-robot teams for future manned space missions.

Lanssie M. Ma received a BA in computer science from the University of California, Berkeley in 2014, and the PhD degree in Computational Science and Engineering from the Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology. Her interests span from human computing interaction to wearable computing and exploratory outer space robotics. She currently explores work allocation in human-robot teaming through modeling and simulating space exploration scenarios to evaluate effective teamwork.

Amy R. Pritchett is the head of the aerospace engineering department and professor at Penn State. She received an SB, SM, and ScD in aeronautics and astronautics from MIT in 1992, 1994, and 1997, respectively. Previously, she served on the faculty of Georgia Tech, and as NASA’s (National Aeronautics and Space Administration) director of the Aviation Safety research program. She has been awarded the AIAA (American Institute of Aeronautics and Astronautics) Sperry Award, the RTCA William H. Jackson Award and, as part of CAST, the Collier Trophy, and the AIAA has named a scholarship for her. Her research interests include the application of cognitive engineering to the design of complex work environments, particularly in aerospace.

Karen M. Feigh received the BS in aerospace engineering from the Georgia Institute of Technology, Atlanta, GA, USA, the M. Phil in engineering from Cranfield University, UK, and the PhD degree in industrial and systems engineering from the Georgia Institute of Technology. She is currently an associate professor with the School of Aerospace Engineering, Georgia Institute of Technology. Her research interests include cognitive engineering, design of decision support systems, human-automation interaction, and behavioral modeling.