Extending MAPE-K to support Human-Machine Teaming

Jane Cleland-Huang  
Ankit Agrawal 
janehuang@nd.edu 
aagrawa2@nd.edu 
Computer Science and Engineering 
University of Notre Dame 
Notre Dame, IN, USA

Michael Vierhauser 
Michael Murphy  
michael.vierhauser@jku.at  
murphyhm18@gmail.com  
LIT Secure and Correct Systems Lab  
Johannes Kepler University Linz 
Linz, Austria

Mike Prieto  
mprieto2@nd.edu 
Computer Science and Engineering 
University of Notre Dame 
Notre Dame, IN, USA

ABSTRACT
The MAPE-K feedback loop has been established as the primary reference model for self-adaptive and autonomous systems in domains such as autonomous driving, robotics, and Cyber-Physical Systems. At the same time, the Human Machine Teaming (HMT) paradigm is designed to promote partnerships between humans and autonomous machines. It goes far beyond the degree of collaboration expected in human-on-the-loop and human-in-the-loop systems and emphasizes interactions, partnership, and teamwork between humans and machines. However, while MAPE-K enables fully autonomous behavior, it does not explicitly address the interactions between humans and machines as intended by HMT. In this paper, we present the MAPE-K\textsubscript{HMT} framework which augments the traditional MAPE-K loop with support for HMT. We identify critical human-machine teaming factors and describe the infrastructure needed across the various phases of the MAPE-K loop in order to effectively support HMT. This includes runtime models that are constructed and populated dynamically across monitoring, analysis, planning, and execution phases to support human-machine partnerships. We illustrate MAPE-K\textsubscript{HMT} using examples from an autonomous multi-UAV emergency response system, and present guidelines for integrating HMT into MAPE-K.

CCS CONCEPTS
• Human-centered computing → Collaborative interaction; 
  HCI theory, concepts and models.

KEYWORDS
Self-Adaptive Systems, Human-Machine Teaming, Autonomous Systems, MAPE-K

1 INTRODUCTION
The MAPE-K feedback loop [7, 43], is a well-adopted reference model for managing and controlling autonomous and self-adaptive systems, and its use has enabled significant advances in autonomous systems over the past decades, for example, in areas such as autonomous driving and traffic management [34], Unmanned Aerial Vehicles [55, 59], Smart Home and IoT applications [6, 36], and assistive robots [39]. Furthermore, rapid advancements in Artificial Intelligence (AI), supported by frameworks such as MAPE-K, have shifted the focus from traditional human-directed robots to fully autonomous ones that do not require explicit human control. These systems, which are commonly developed as “Human-on-the-Loop” (HotL) [30] systems, differ from “Human-in-the-Loop” (HitL) systems in several important ways. In HitL systems, humans make decisions at key points of the system’s execution; while HotL systems take full advantage of machine autonomy to perform tasks independently, efficiently, and quickly.

However, given technological advances in autonomic computing, a more advanced form of collaboration, referred to as Human Machine Teaming (HMT) has emerged [47]. HMT emphasizes interactions, partnership, and teamwork between humans and machines. It capitalizes upon the respective strengths of both the human and the machine, whilst compensating for each of their potential limitations [47, 54]. According to McDermott et al., effective HMT requires transparency of the machine’s progress and plans, as well as augmented cognition to empower the machine to adapt as needed, keep the human partner aware of critical problems, and allow both human and machine to explore the shared solution space. Coordination between humans and machines establishes shared knowledge and trust between both human and machine partners and empowers a human partner to direct the machine’s behavior when desired [56, 57]. Finally, the machine’s self-adaptation capabilities are extended to allow it to dynamically configure and reconfigure its interactions with human partners throughout the mission.

HMT systems incorporate aspects of both Cyber-Physical Systems (CPS) [74], and Socio-Technical Systems (STS) [25, 81]. In the context of HMT, systems are still expected to operate fully autonomously; with all the capabilities that MAPE-K is designed to support. In fact, not only are the machines capable of performing their tasks autonomously, but they are perceived as true partners and not just “tools” in achieving mission goals. To make this transition from the HotL paradigm to HMT, humans and machines must interact more closely – not in a way that reduces or curtails the autonomous behavior of the machine, but in one that leverages that behavior to create meaningful partnerships.
The primary goal of any feedback control system is to remove humans from the loop, and therefore MAPE-K focuses upon autonomous decision-making and self-adaptation without emphasizing the human aspects of a CPS. This was reflected in the results of a recent systematic literature review [10] which reported that runtime models associated with self-adaptation primarily target the architecture, structure of the system, and/or its goals, but hardly incorporate any human-related factors or activities, such as user interaction or situational awareness [26]. This creates a gap between the existing MAPE-K framework and the capabilities needed by an autonomous system to fully interact with human partners in an HTM environment. Kephart proposed bridging this gap through creating highly interactive "dialogs" between humans and machines; however, his examples are all drawn from information systems and not real-time robotic environments [42].

We propose a solution for bridging this gap in more diverse system environments through MAPE-K<sub>HMT</sub>, which enhances the fundamental MAPE-K loop with runtime support for HMT, and aligns teaming factors, identified from the HMT literature [57] with the different phases of the MAPE-K loop. We then present a set of runtime models for use in MAPE-K<sub>HMT</sub> and describe how they enable support for bidirectional human-machine interactivity and decision-making. Our approach is illustrated using a set of worked examples taken from a multi-agent system of autonomous Unmanned Aerial Vehicles (UAV) for supporting emergency response missions [21, 24]. Based on these examples, we have derived a lightweight process and recommendations for integrating HMT into MAPE-K.

The remainder of this paper is structured as follows. Section 2 introduces MAPE-K<sub>HMT</sub> and its extensions to the MAPE-K loop. Section 3 introduces our case study system, while Section 4 presents six examples of HMT-related runtime models and their integration into MAPE-K<sub>HMT</sub>. Section 5 then describes a process for utilizing MAPE-K<sub>HMT</sub> in a MAPE-K system. Finally, Sections 6 to 8 present threats to validity, related work, and conclusions.

2 THE MAPE-K<sub>HMT</sub> LOOP

With MAPE-K<sub>HMT</sub> we aim to leverage the benefits of active human engagement while preserving the autonomous behavior of the self-adaptive system. MAPE-K<sub>HMT</sub> follows the same general structure of MAPE-K, however, as illustrated in Fig. 1, each of the phases is augmented with additional capabilities targeted to support HMT. The original MAPE-K loop [43] consists of four pivotal phases: Monitoring where information is collected from the environment, Analysis where data is analyzed to determine if adaptations need to be performed, Planning where corresponding actions and adaptations are planned, and finally Execution in which the proposed plans are enacted. Additionally, the “K” stands for an underlying knowledge base, accessible to all other parts of the MAPE loop, and often supported by runtime models [35].

In the context of human-machine teaming, a number of capabilities are required to ensure both successful human-machine interaction and cooperation [57]. In Table 1, we summarize some of the primary dependencies. Transparency is supported by “Observability” (TF1) of the autonomous partner’s task progress and “Predictability” (TF2) of its future plans. Cognition is augmented through “Directing Attention” (TF3) to critical problems, for example by raising meaningful alerts to increase situational awareness. “Solution Exploration” (TF4) and “Adaptability” (TF5) imbue both
Table 1: Key impacts of HMT factors upon phases of the MAPE-K loop. The factors (TF1-TF8) are proposed by Mc Dermott et al. [56] and augmented to reflect the bidirectional partnerships proposed in this paper.

| Teaming Factor | Definition | Key example of HMT realization in MAPE-K_HMT |
|---------------|------------|---------------------------------------------|
| TF1 Observability | Visibility of the task progress of the automated partner and human actions. | [M⁺⁺] Data related to status & task progress is collected from sensors, UI inputs, and software probes. |
| TF2 Predictability | Transparency of future intentions, states, and activities. | [A⁺] Progress is dynamically visualize for humans. |
| TF3 Directing Attention | Keeping human partners aware of critical problems through warnings, recommendations, and indicators. | [A⁺] Relevant data is collected & analyzed. |
| TF4 Solution Exploration | Ability for both partners to leverage multiple views, knowledge, & candidate solutions. | [P⁺] Human engages in adaptation planning alongside machine. |
| TF5 Adaptability | Ability to address potentially unexpected evolving, dynamic situations through adaptation. | [M⁺⁺] Humans draw on their observations & analysis to intervene in the machine’s plans. [E⁺] The machine requests support from the human when its confidence is low. |
| TF6 Directability | Humans ability to direct/redirect an automated partner’s resources, activities, and priorities. | [A⁺⁺] The machine requests appropriate trust-related indicators at runtime [E⁺]. |
| TF7 Calibrated Trust | Trustworthiness indicators of machine’s ability to make correct decisions in current context. | [M⁺⁺] Based on collected data [M⁺⁺] the machine [P⁺⁺] computes reliability of its own autonomous decisions and actions, and displays appropriate trust-related indicators at runtime [E⁺]. |
| TF8 Common Ground | Shared beliefs, assumptions, and intents across human and automated partners. | [M⁺⁺] The Human understands data collection, analysis & use, [A⁺⁺], & the machine’s capabilities [P⁺⁺/E⁺⁺]. |
|               |              | [E⁺⁺] The machine responds to human directions as expected. |

the machine and humans with the knowledge and capabilities they need to make and enact decisions. Finally, Coordination is supported through “Directability” (TF6), “Calibrated Trust” (TF7), and establishing “Common Ground” (TF8) to enable informed, trustworthy, and trusted partnerships. Achieving each of these capabilities requires consideration for how humans and machines can work in teams to accomplish their goals, then establishing runtime models to collect, aggregate, and visualize information that supports HMT. As a result, human and machine collaborators can engage in meaningful interactions. These eight capabilities represent Teaming Factors that are applied across the MAPE-K loop to augment each phase as M⁺, A⁺, P⁺, E⁺, and K⁺ respectively. In the following sections, we explain how each phase is augmented to support HMT.

### 2.1 The Monitoring Phase (M)

In the MAPE-K loop, monitoring is primarily concerned with collecting data from the self-adaptive system and the environment in which it operates. Hardware sensors provide raw data such as temperature, distance to potential obstacles, video streams, or GPS locations, while software probes provide data from the running system such as its resource usage, response times, and currently executing tasks [37]. This data is collected, persisted, and used in runtime models to guide subsequent analysis and self-adaptation decisions [40, 79].

M⁺⁺: To forge effective human-machine partnerships, HMT environments must provide bidirectional situational awareness; therefore, the machine not only collects data about its own state but also collects human-initiated data reflecting human goals, directives, workload, and response times [26]. This data is collected via Graphical UIs (GUIs) and via hardware interfaces such as radio controllers (RC), audio devices, pointing devices, and even eye-trackers [17, 32, 62, 63] or brain interfaces [33, 61]. Furthermore, given the dissonance between the way humans and machines perceive the world [85], supplementary data is needed to support human awareness and interactions and must be collected from all three sources (i.e., machine, environment, and human inputs). The data collection process is therefore expanded accordingly and used across subsequent analysis, planning, and execution phases to support the HMT goals of transparency, cognition, and coordination, with an emphasis on the teaming factors of observability, directing attention, calibrated trust, and common ground.

### 2.2 The Analysis Phase (A)

The MAPE-K analysis phase is concerned with determining whether adaptation actions are required, based on the current and predicted state of the system, the environment it is operating in, and its defined goals, safety constraints, and quality of service specifications. Automated analysis enables timely and fast reactions to changes in the environment and emergent situations.

A⁺⁺: HMT recognizes the value of augmenting machine analysis capabilities with human perspectives and inputs [85]. This requires information to be exchanged between machines and humans through...
human-facing interfaces that provide insights into the autonomous machine’s intent, performance, plans, and reasoning processes’ [38, 71]. Human analysis often relies upon a combination of real world observations as well as software supported interactions. In direct real-world analysis, for example, the human uses their physical senses to observe and analyze the machine’s behavior, and to intervene quickly and directly through either a hardware or software interface, potentially causing a temporary interrupt to the MAPE-K loop. In contrast, in software-supported analysis, the human analyzes a representation of the machine’s behavior and/or state through a GUI. As humans need time to understand and analyze the information and formulate decisions, static snapshots become stale within seconds – even milliseconds. This leads to the conclusion that GUI support for HMT must include dynamic runtime views that reflect current system states and historical information about past actions, rather than just static snapshots of the system. They are therefore often built on top of existing runtime models with continually updated views. Historical views that enable the human to explore why a machine made a past decision (e.g., [67, 70]), or what it plans to do next, are also needed. The MAPE-K_{HMT} analysis has a broad impact across almost all of the teaming factors as analysis is a precursor to human engagement.

2.3 The Planning Phase (P)

In the planning phase the machine plans self-adaptation actions such as switching states to perform different tasks, reconfiguring existing features, activating or deactivating sensors, or modifying polling frequencies to preserve power or to collect additional information about the system or its environment.

P**: HMT introduces two additional considerations to the planning phase. First, the human leverages their observations of the machine, its operating environment, interactions with other team members, and profound experience as a “human knowledge base” to engage directly in the planning process. Humans might reconfigure the mission’s goals or plans, or temporarily intervene in the operation of the machine, for example by assuming manual control of a task that the machine is not able to perform autonomously or when the machine malfunctions. However, this introduces a potential tug-of-war that can occur when humans and machines create competing plans [22]. This was catastrophically illustrated in the crash of Lion Air Flight 610 and Ethiopian Airlines Flight 302 in which the MCAS (Maneuvering Characteristics Augmentation System) incorrectly perceived the angle of attack to exceed predefined limits and therefore pushed the nose of the plane down, whilst pilots struggled to push it back up [31]. The system was not designed to detect and mitigate this type of tug-of-war, and ultimately the machine “won”, causing the planes to crash. Achieving effective coordination between humans and machines is a challenging problem which we discuss further in Section 4.3.

HMT has a second major impact on the planning phase, as the machine may make additional self-adaptation plans targeted at enhancing the human’s interactive experience. For example, the machine might self-adapt its internal alert system to adjust the type and frequency of human alerts if the system perceives that human response time is starting to lag [3].

2.4 The Execution Phase (E)

During the execution phase, the previously generated plan or adaptation strategy is executed on the physical machine or device. E**: In an HMT setting, both the machine and the human partners enact plans – sometimes closely coordinating their work whilst at other times working more independently on tasks that each partner is best suited to perform.

2.5 The Knowledge Base (K) + Runtime Models

MAPE-K employs diverse runtime models that represent the structure, behavior, and/or goals of a system at runtime. These models are pivotal for guiding autonomous adaptation decisions [8]. In a recent systematic literature review Bencomo et al., [10] reported that Models@run.time have been used in numerous ways – for example to depict the current state of the system [28] and its behavioral dynamics specifying exactly what the system is able to do from its current state [29], model system goals [18, 69, 77] and functional and non-functional requirements [20, 41], and to depict product variability [82]. They provide a bidirectional reflection layer, such that changes in the runtime model trigger changes in the goals, structure, and/or behavior of the underlying system, whilst changes in the system are reflected in the model. As such, runtime models support system autonomy, imbuing the system with the ability to sense, analyze, predict, and to make independent decisions.

K**: In MAPE-K_{HMT} runtime models must also support the HMT goals of transparency, augmented cognition, and coordination between human and machine partners. Depending on the type of system, application domain, or the type of missions that are executed, this requires different types of runtime models that are explicitly designed to provide information to the user or relay critical information from the user to the system so that informed adaptation decisions can be made.

3 CASE PROJECT: A MULTI-UAV SYSTEM

Throughout this paper, we illustrate MAPE-K_{HMT} with examples drawn from our Drone Response system – a flexible and configurable framework for multi-UAV missions [21, 24].

3.1 The Drone Response Ecosystem

Drone Response is fully deployable with both physical and simulated UAVs [46, 65]. The Drone Response architecture includes diverse user interfaces, a Ground Control Station (GCS), and autonomy capabilities located onboard each UAV. The MAPE-K infrastructure is distributed across the Drone Response ecosystem as follows.

- **UAV Onboard Pilot**: The Onboard Pilot module acts as an application layer for the autopilot stack which in turn includes flight control software and hardware for executing plans. Its internal State Machine receives mission specifications and instantiates itself accordingly. It then uses its onboard state machine to progress through a series of tasks, with transitions triggered by events it detects by analyzing its own sensor data and/or messages received from the GCS or other UAVs. MAPE-K’s monitoring, analysis, planning, and execution occur onboard each UAV with an emphasis on managing the tasks and adaptations of each individual UAV.
3.2 A Motivating Scenario

Drone Response’s UAVs are designed to leverage computer vision and work alongside humans in emergency response missions such as search-and-detect, surveillance, and rescue scenarios. As depicted in Fig. 3, the UAVs first use their onboard computer vision to search for, and potentially detect the victim. They then notify the human responders about the victim’s location. Drone Response then has the option of delivering a flotation device, as shown in Fig. 4; however, this requires very close human-machine teaming. For example, if a physical rescue from a boat is imminent, then simultaneously dropping a flotation device in the location of the victim could hinder the rescue operation and introduce a potential safety concern. This example provides a clear differentiation between HiTL, HoTL, and HMT paradigms. In a HiTL environment, the UAV would wait to be dispatched by a human and subsequently request input regarding specific tasks to be performed. In a HoTL environment, the UAV might decide to initiate the delivery of the flotation device independently; however, the human could decide to intervene and cancel the operation. In contrast, while human intervention is definitely part of HMT, the HMT scenario calls for joint decision-making, whereby the human and machine leverage a shared set of beliefs and understanding of the mission to evaluate their capabilities and opportunities, weigh the options with respect...
to global mission goals, and ultimately make a joint decision for the good of the mission.

All eight HMT teaming factors (cf. Table 1) play a vital role in fostering collaborations between emergency responders and the UAVs. Because humans need to develop trust in the autonomy capabilities of the UAV during the mission, they require observability of their tasks, and expect predictability of their decisions and actions. The system must raise appropriate alerts in order to help humans to maintain situational awareness. In the spirit of the HMT partnership, the human can directly contribute to the search – for example, providing information for use by the computer vision algorithm or hints to the route planner, that the victim is wearing a red sweater and that sightings have been reported in a certain part of the search area. To increase the likelihood of a successful search, the UAVs and humans must establish a shared conceptual model of the mission and humans must be able to explore the solution space through user-facing perspectives of the current mission, including the location, status, and progress of each UAV.

4 APPLYING MAPE-K\textsubscript{HMT}

In this section, we describe six different runtime models from our Drone Response project that support HTM within the MAPE-K loop. Each of the runtime models contributes towards one or more HMT goals of transparency, augmented cognition, or human-machine coordination. Furthermore, each model not only reflects the runtime behavior or structure of the machine, but also provides direct or indirect support for a human-facing UI component in order to support human-machine collaboration. In the following sections we describe models that are particularly pertinent to transparency, cognition, and coordination respectively.

4.1 Runtime Models for Transparency

HMT’s transparency goal focuses on observability and predictability and therefore aligns closely with Endsley’s Situational Awareness goals of observing and understanding [26, 27]. Observability means that humans are aware of what their autonomous partner is doing, including its goals, current tasks, future intentions, status, progress, ability to adapt to changing contexts, rationales for adaptations, and challenges or constraints that impact its ability to solve the current problem. Additionally, predictability helps to remove potential surprises introduced by the machine’s decisions, provides insights into uncertainties [9] and ways in which reliability of the autonomous partner changes over time, under what circumstances it changes, and how its decisions are made. Given the importance of the transparency goal, and the many facets of observability and predictability, systems will likely have several associated runtime models. These models tend to use runtime data about the machine’s progress and its environment collected using runtime monitors. Based on this, visualization and interactivity support is provided for the analysis phase. Drone Response primarily supports transparency through the use of several map-based views that depict the current location and status of each UAV, as well as task-based views such as the one presented in the following example.

- **Task-Centric Model:** In a multi-agent system, the human needs to understand exactly what each individual agent is currently doing and its progress towards the overall mission plan. The Drone Response task-centric view generates a global state transition model combining the states and transitions from all active UAVs’ onboard state machines, and then tracks and displays the current task performed by each UAV in the merged state transition model. Progress is then visualized in a UI showing the current task of each UAV as a colored token assigned to a specific node. This is illustrated in Fig. 5, which shows that the Green (G) delivery UAV is on standby, Red (R) and Orange (O) UAVs are searching, the Purple (P) UAV is performing surveillance, and the Blue (B) UAV has detected a victim.

4.2 Runtime Models that Augment Cognition

The cognition goal extends far beyond basic transparency and is intended to help humans understand emergent problems and their
Figure 6: The ‘Alert Triage’ model stores user-created and/or machine-learned prioritization rules, and triages the display of alerts for each view (e.g., Map view on the right).

causes, provide insights into the decisions and actions taken by the autonomous partner, and allow the user to explore different perspectives and solutions [50]. MAPE-K_HMT runtime models support cognition goals through enabling the right information to be presented or available to the human at the right time, without overloading their cognitive abilities [3, 45, 83]. The two examples we present here focus on triaging runtime alerts and explaining autonomous actions of the machine by generating human-readable explanations of autonomous behavior; however, our previously presented task-centric model (cf. Fig. 5) provides an additional example, as its interactive GUI also provides interactive and more detailed views of individual UAV’s tasks and progress.

- Alert Prioritization Model: The Drone Response alert prioritization model is designed to avoid the situational awareness “design demon” of information overload [26], and is built upon a formal meta-model for human-UAV collaborations [4]. It focuses particularly on the HMT goal of augmenting cognition with an emphasis on directing human attention to important messages. In Fig. 6 alert rules and priorities are shown on the left. They are initially provided as default values by human stakeholders but can be dynamically adapted at runtime, by both the human and the machine. They specify essential alerts which must always be displayed (✓), and prioritized alerts (1-5) which will only be displayed if they don’t cause the maximum threshold to be exceeded. The triage part of the model (e.g., Map View), is dynamically maintained by the system at runtime and is responsible for managing alerts in each active GUI view. It is notified whenever an alert is generated by a runtime model hosted on the UAV or on a GCS microservice. It is also notified by the GUI server whenever a new GUI is activated or deactivated. The alert prioritization model thus builds upon services already available in the basic MAPE-K loop, by collecting, aggregating, and processing the data they produce, in order to support the HMT-focused capability of triaging alerts.

(a) Explanation templates for internally and externally triggered adaptation events initiated by either the human (H) or machine (M).

(b) An example explanation generated by the runtime model.

Figure 7: The Autonomy Explanation Model generates an explanation for all major adaptation decisions.

- Adaptation Explanation Model: Finally, the explanation model generates explanations for UAV autonomous decisions so that humans can gain insights into the machine’s reasoning and assess the appropriateness of individual adaptations [3, 48]. These insights potentially strengthen the human’s trust and confidence in its machine partner. The predefined explanation templates shown in Table 7a are used to dynamically generate human-readable textual explanations for all adaptations performed by a UAV. Whenever a UAV self-adapts, it collects three types of information. These are explanation snippets describing relevant external events (e.g., “misty weather conditions” or “victim detected”), the UAV’s response to the events (e.g., “reduced altitude by 8 m” or “switched to tracking mode”), and finally, the rationale behind those actions (e.g., “limited visibility”, or “high confidence in victim sighting”). An example of a weather-related adaptation explanation is shown in Figure 7b. Upon receipt of the adaptation message, the runtime model selects the appropriate template and generates the explanation by filling in the missing parts with the data provided by the UAV [3]. This model, therefore, utilizes the outputs of existing runtime adaptation models to provide explanations that are critical for HMT.

4.3 Runtime Models for Coordination

Coordinating the actions of both humans and machines represents a challenging problem [51] that is exacerbated by the differing cadences of human and machine response times. Successful coordination depends on many factors, including a shared conceptual
model of the operating environment, respect for each partner’s capabilities, and well-calibrated bidirectional trust. Humans trust the machine to operate autonomously when capable and to request help when needed. Conversely, machines accept interventions from humans and expect feedback when requested, if the human is available. In this context, we present three different runtime models covering the cases of machine-initiated and human-initiated coordination, as well as the particularly challenging case that occurs when humans and machines generate conflicting plans of action.

- **Machine Initiated Coordination**: In Drone Response much of the autonomous behavior of the UAV is supported by its onboard CV capabilities (cf. Fig. 8) [1]. For example, when searching for a drowning victim, the UAV uses CV to continuously analyze the video stream and detect objects classified as “person”. For each detected object, the CV module generates two scores. The confidence score represents the probability that the object is correctly classified, while the reliability score accounts for any uncertainty arising from image noise or mismatch between the current context and the training data [64]. Based on learned threshold values, the UAV uses these scores to decide whether it can autonomously decide on its actions (e.g., track the object vs. continue to search) or should request help from the human. Both the CV model and the related human-machine coordination mechanism are supported by runtime models which take a coordination specification as input (cf. Algorithm 1), instantiate a simple state machine (managed by a microservice on the GCS), and use the existing messaging system to choreograph human and machine tasks. In closely related work Li et al., also proposed a form of choreography that provides users with advanced knowledge of tasks they would be requested to participate in [51].

- **Human Initiated Coordination Models**: In order for the human to engage in meaningful interactions with the machine, they must have high degrees of situational awareness and a clear understanding of what they can and cannot do at any point during a mission. Furthermore, the different cadences of human and machine response times within MAPE-KHMT create a significant coordination challenge resulting in problems such as the human making plans based on stale data or intervening after the machine has already completed an action. An example of a relevant runtime model determines currently available human interaction options and dynamically adapts each active GUI to activate and deactivate widgets (e.g., icons, buttons, menu options) according to the ways that humans can feasibly interact with the machine given the current state of the mission. For example, if the UAV is currently in active search mode, it is reasonable for the human to request a view of the UAV’s annotated video stream; however, this option should not be available if the UAV is in RTL mode with cameras turned off to preserve limited power. A suitable affordance (e.g., a button) should be activated and deactivated accordingly. Tracking these currently available human actions is handled by a dedicated runtime model.

- **Mitigating Tug-of-War Scenarios**: Finally, a significant challenge in HMT is reconciling potentially conflicting actions taken by the human and machine, a problem that is again exacerbated by the different operating speeds of the machine and human. As the machine has the advantage of faster cadence, it will often win any disagreements with potentially devastating results, as illustrated by the case of the Lion Air Crash (cf., Section 2.3). Similar scenarios play out across other domains. For example, a UAV may place itself close to the river to collect better imagery of the riverbank. On a sunny day, reflections from the light on the water could impact the UAV’s sensors causing sudden altitude fluctuations and triggering a land-in-place failsafe mechanism to activate. An alerted human might quickly intervene to prevent the UAV from landing in the river directing it to ascend to a safe altitude; however, once the UAV’s autonomy kicks back into gear, the whole cycle could repeat itself – causing a tug-of-war with respect to the ideal altitude placement. HMT systems must allow humans and/or machines to detect and break interwoven cycles of human and machine actions that are indicative of a tug-of-war. As a simple solution, detected cycles could be broken by curtailing the UAV’s autonomy until reestablished by the human.

### 4.4 Integrating the RunTime Data Probes

While we have presented these six example models as independent entities, many of them share common data inputs (e.g., data read from sensors and software probes), or rely on data/outputs from

---

**Algorithm 1**: Human-Machine coordinated decision-making addressing Computer Vision reliability problems

```plaintext
if Object detected at low reliability then
  UAV raises alert and requests help from human;
  if Human is available and responsive then
    Human evaluates video stream, makes decision, and selects CONFIRM or REJECT option;
    if Human confirms victim sighting then
      UI Server sends CONFIRMATION message to UAV;
    else
      Human refutes sighting;
      UI Server sends REFUTATION message to UAV;
  end
  if no response from human within waiting_period then
    NO RESPONSE message sent to UAV;
    'human failure to respond' event is logged;
    Responsibility reverts to UAV;
end
```
another runtime model. For example, in order to generate meaningful explanations where multiple UAVs are involved, information is collected from the individual UAV onboard runtime models. The Autonomy Explanation Model relies upon “adaptation notification events” provided by each individual UAV, which in turn employs their own state machine. This information is then aggregated and post-processed, so that the Alert Prioritization Model (cf. Fig. 6) can select and display critical alerts from individual UAVs or global alerts from the system. While both the knowledge base, and (parts) of the runtime models are distributed across the respective agents they share data as needed. This means that the UAV retains full autonomy when a person is detected, and adaptation decisions are made (onboard), but at the same time raises alert events which support decision-making of the human partner. This dependence on accurate and appropriate information between the different models calls for thorough design and planning to ensure that the right information is available at the right time in a potentially resource-constrained operating environment.

5 ANALYSIS AND TAKE-AWAYS

In this paper, we have proposed augmenting the MAPE-K loop with support for HMT. MAPE-K\textsubscript{HMT} assumes that both humans and machines are capable of autonomous behavior and decision-making, and that mission goals are achieved jointly through an interactive partnership. Based on our own experiences in applying MAPE-K\textsubscript{HMT} to the Drone Response system, we lay out an initial process for designing and deploying MAPE-K\textsubscript{HMT}.

(1) **Stakeholder Identification:** HMT systems inherently involve humans, and therefore, it is important to identify CRACK (Collaborative, Representative, Authorized, Committed, Knowledgeable) [12] stakeholders serving as direct users and domain experts. Engaging with stakeholders, who will become the human partners, helps to uncover interactions and expectations users have on the system, for example, by carefully exploring the human-machine interactions related to mission-related scenarios [76].

(2) **Elicitation of HMT requirements:** Once stakeholders are identified, requirements must be elicited. As a starting point, McDermott et al. [57] has described a detailed elicitation process, supported by a list of key questions associated with each of the HMT factors [56]. For example, to understand “predictability” requirements, analysts must discover (a) automation goals, abilities, and limitations, (b) how the human partner’s goals and priorities are tracked, (c) reliability of different automated tasks within different contexts, and (d) the types of changes that are expected to occur and trigger subsequent adaptations.

(3) **Requirements Analysis and Specification:** The elicited requirements are subsequently analyzed to negotiate and reconcile trade-offs, and to identify and specify requirements that support human-machine interactions [68]. While our process is agnostic to specific techniques, the domains in which MAPE-K operates generally dictate that the requirements process includes a rigorous safety analysis (e.g., [23, 49, 80]), and determines that requirements should be specified sufficiently formally to capture timing and other performance constraints and/or to establish formal goal models. In many cases, the requirements specification (e.g., Goal Models or state transition diagrams) provides the foundation for the respective runtime models [5, 10, 15].

(4) **Design and Integration of HMT runtime models:** MAPE-K\textsubscript{HMT} emphasizes the importance of HMT-related runtime models. We, therefore, start the design process by taking an inventory of existing runtime models, identifying gaps where HMT requirements are not adequately supported by existing models, designing new runtime models as needed, and then finally composing models into workflows to service each HMT requirement. This involves assessing who (machine or human-role), when, and where each model will be used and updated, and what data sources are required as inputs (e.g., probes, message subscriptions, or human-initiated data and events). Furthermore, as actions are performed at different speeds, the required refresh frequencies must be determined for each constituent element of each runtime model, and a system-wide plan established so that each collected data attribute satisfies refresh frequencies of all relevant models.

(5) **User Interface Design:** User interfaces need to be designed to provide interactive support for humans. Depending on the data that is displayed, or the input that is expected from the human, this may include GUIs, or hardware interfaces such as Joysticks or radio controllers. Mission- and safety-critical information needs to be provided to the human in a timely manner without creating cognitive overload, and must not be “hidden” in submenus or views that require multiple steps to access [26].

(6) **Implementation, Testing, and Deployment:** The final steps involve implementing the system, verifying that required runtime models, UIs, and supporting features are implemented as intended, and finally validating that the deployed system satisfies its stated requirements and supports the desired HMT. Further discussion of these steps is outside the scope of this paper.

6 THREATS TO VALIDITY

Our work is subject to three primary threats to validity. First, all of the runtime models described in this paper are designed to support HMT in our own Drone Response system, representing a multi-agent, multi-human system operating in a mid-level safety-critical domain [53, 78]. As types of interactions are influenced by the operating domain, the human-machine partnerships in Drone Response may differ significantly from other application domains – for example, those with a single operator or a single machine, or operating in a highly safety-critical domain. On the other hand, instead of proposing a specific set of HMT-related runtime models, MAPE-K\textsubscript{HMT} provides a simple process for identifying, developing, and integrating context-specific models in support of transparency, augmented cognition, and coordination goals, which are known to be applicable across diverse operating environments [47, 56, 57]. Further, our process could be easily extended – for example, by integrating a more formal safety analysis for more critical domains. Our future work will apply this process to more varied systems.

Second, while our runtime models have been prototyped in various forms, only some have been integrated into the full version of Drone Response. As such, this paper represents a vision, supported by concrete examples. Future work will focus on concrete
implementation details. For example, while we have discussed the creation of shared data probes that service the refresh frequencies of all runtime models, and have developed monitoring capabilities to achieve this, this aspect of the work needs further investigation.

Finally, while Drone Response has been deployed in the physical world, formal evaluations of HMT user interfaces have relied upon user studies conducted in Drone Response’s simulator (e.g., [2, 3, 79]). However, the Drone Response GUI is identical for both physical and simulated UAVs, and our past experiences have shown that findings from these user studies have been effective when deployed in physical field tests. Nevertheless, further experiments targeted specifically at HMT need to be conducted in real-world settings where humans collaborate with UAVs under far more noisy, volatile, and potentially stressful conditions.

7 RELATED WORK
A significant body of work, primarily in the HCI community has focused on designing UIs to support situational awareness of Cyber-Physical Systems (e.g., [2, 4, 11, 66, 72, 75]). The focus has primarily been on enabling users to perceive, understand, and make effective decisions [26, 27]. While this related work supports HMT goals, it puts little emphasis on integration with underlying runtime models which are needed in order to deliver accurate, timely, and often aggregated data for use in the UIs. Yekpol designed an increased interactivity between humans and users in adaptive decision-making [42]; however, their experiments were all taken from domains in which humans had plenty of time to consider their decisions. Integrating HMT into the MAPE-K loop for real-time robotics systems introduces additional, and very challenging, timing constraints.

In the HMT domain, researchers have explored many facets of human-machine teaming. For example, Klein et al. [44] identified challenges associated with achieving shared goals, preventing breakdowns in team coordination, and fostering communication and collaboration. Furthermore, Schmid et al. [73] studied ways to adjust system automation in complex, safety-critical environments in order to better support human operators. While their work has significant relevance to MAPE-KHMT, it perceives humans as “operators” rather than true partners.

Calhoun et al. [16] proposed a flexible architecture that allows the degree of automation to vary according to the human’s current engagement and workload. These goals are reflected in our discussion on adaptation in MAPE-KHMT, which embraces the notion of adapting for improved human collaboration and performance. To increase context awareness in order to better engage humans in the decision-making process, Li et al. [51] proposed a formal framework based on probabilistic reasoning to determine when advanced notifications are useful for humans interacting with self-adaptive systems. Their work specifically addresses the cadence problem in which humans may be required to respond quickly but require “thinking time”. However, their evaluation was conducted in a robotics goods delivery domain, which has a lower decision cadence than a multi-UAV system.

Finally, several papers have explored self-adaptation and/or human interactions in the UAV domain. Vierhauser et al. [80] identified human-UAV interaction hazards and potential mitigations, whilst Miller et al. [58] explored different techniques by which users could interact with multiple UAVs. However, these papers focus on Human-on-the-loop rather than HMT. Other researchers have proposed UAV-related self-adaptation frameworks. For example, Braberman et al. [13, 14] presented a MAPE-K reference architecture for unmanned aerial vehicles, while Yu et al. [84] proposed a self-adaptive framework for UAV forensics. Neither of these explored the HMT aspects of adaptation. Finally, in more closely related work Lim et al. [52] explored ways to adapt human-robot interactions in a multi-UAV system, with a focus on modulating automation support according to the cognitive states of the human operator. This creates a subtle, but important difference from our MAPE-KHMT goal, as it centers primarily around the needs of the operator rather than optimizing teamwork goals.

8 CONCLUSION
This paper has described the techniques and process we have used to augment MAPE-K in order to address the three HMT goals of transparency, augmented cognition, and coordination. We have described how MAPE-K provides a meaningful self-adaptation framework in which humans and machines collaborate together, and have mapped HMT factors to the MAPE-K phases and then used a series of examples to illustrate how carefully designed runtime models enable meaningful support for human-machine partnerships. Rather than diminishing the autonomy of machines, HMT draws upon the autonomous abilities of both humans and machines to deliver an even stronger solution realized through developing meaningful teamwork.

In conducting this work we have identified two key challenges. The first stems from the very different cadences at which humans and machines operate. This creates the potential for humans to make decisions based upon stale data, and for directives interjected into the machine’s plans to inadequately reflect the current state of the system. Existing HMT solutions, such as turn-taking [19], are only effective in scenarios which can tolerate slower response times. In Drone Response we partially address this challenge by dynamically adapting the UIs to only reflect currently available human interaction options, and by implementing a reliable messaging system which ensures that messages are only sent to the UAV when it is in a state to handle them; however, the problem is a complex one which warrants further exploration. The second challenge relates to designing and supporting the integrated MAPE-KHMT environment. As different runtime models are required for effective human-machine teaming, and these need to be identified, designed, integrated, deployed, and effectively maintained at runtime.

Even beyond these challenges, much additional work is needed in order to realize the vision of trusted, coordinated, and accountable teams of humans and machines. Our ongoing work will therefore investigate solutions for addressing these challenges, create a fully integrated MAPE-KHMT environment for evaluation purposes, conduct field-tests with physical UAVs, and explore the deployment of MAPE-KHMT across a much broader set of different domains.

ACKNOWLEDGMENT
The work in this paper has been funded by the US National Science Foundation (NSF) under Grant# CNS-1931962 and also by the Linz Institute of Technology (LIT-2019-7-INC-316).
Margarita Konaev and Husanjot Chahal. 2021. Building trust in human-machine teams. 

Patricia McDermott, Cindy Dominguez, Nicholas Kasdaglis, Matthew Ryan, and Nianyu Li, Javier Cámara, David Garlan, Bradley Schmerl, and Zhi Jin. 2021. Nathan Koenig and Andrew Howard. 2004. Design and Use Paradigms for UAVs in civil airspace: Safety re-

Colin Paterson, Radu Calinescu, Suresh Manandhar, and Di Wang. 2019. Us-

Robert Loh, Y. Bian, and Tim Roe. 2009. UAVs in civil airspace: Safety re-

Jeamin Koo, Jungsuk Kwac, Wendy Ju, Martin Steinert, Larry Leifer, and Clifford Nass. 2015. Why did my car just do that? Explaining semi-autonomous driving actions to improve driver understanding, trust, and performance. International Journal on Interactive Design and Manufacturing (IJIDeM) 9, 4 (2015), 269–275.

Nancy Levenson. 2011. The Use of Safety Cases in Certification and Regulation. Technical Report. MIT.

Nianyu Li, Sridhar Adepu, Eunsuk Kang, and David Garlan. 2020. Explanations for human-on-the-loop: A probabilistic model checking approach. In Proc. of IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems. ACM, New York, 181–187.

Nianyu Li, Javier Cádiz, David Garlan, Bradley Schmerl, and Zhi Jin. 2021. How Preparing Humans to Do Tasks in Self-Adaptive Systems. In Proc. of 2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems. IEEE, 48–58.

Torkel Klingberg. 2009.

Owen Reynolds, Antonio Garcia-Dominguez, and Nelly Bencomo. 2020. Auto-

Trevor Kistan, Nestor Azizer, and Daniel J. Bursch. 2021. Adaptive Human-Robot Interactions for Multiple Unmanned Aerial Vehicles. Robotics 10, 1 (2021). https://doi.org/10.3390/robotics10010102

Robert Loh, Y. Bian, and Tim Roe. 2009. UAVs in civil airspace: Safety re-

Arad M. Madni and Carla C. Madni. 2018. Architectural Framework for Exploring Adaptive Human-Machine-Team Options in Simulated Dynamic Environ-
ments. Systems 6, 4 (2018). https://doi.org/10.3390/systems06040044

Paulo Henrique Maia, Lucas Vieira, Mathias Chagas, Yijun Yu, Andrea Zisman, and Basha Nuseibeh. 2019. Dragonfly: a tool for simulating self-adaptive drone behaviours. In Proc. of 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems. IEEE, 107–113.

Patricia McDermott, Cindy Dominguez, Nicholas Kasdaglis, Matthew Ryan, and Isabel Trahan. 2018. Human-Machine-Teaming Systems Engineering Guide, Technical Report # MP180941. (2018).

Patricia L. McDermott, Katherine E. Walker, Cynthia O Dominguez, Alex Nelson, and Nicholas Kasdaglis. 2018. Quenching the Thirst for Human-Machine Teaming Guidance: Helping Military Systems Acquisi-
tion Leverage Cognitive Engineering Research, The MITRE Corporation. https://www.nist.gov/publications/technical-papers/quenching-the-thirst-for-human-machine-teaming-guidance-helping.

Christoph A. Miller, Joseph Mueller, Christopher Geb, David LaVerge, Philip Walker, and Joshua Hamel. 2019. User Interaction Approaches For Managing Unmanned Vehic-

Lucas Sakizloglou, Sona Ghahremani, Matthias Barkowsky, and Holger Gies.

Tim Claudius Stratmann and Susanne Boll. 2016. Demon Hunt - The Role of End-

Aurélien Vialon, Kenji Tei, and Samir Aknine. 2017. Soft-Goal Approximation Context Awareness of Goal-Driven Self-Adaptive Systems. In Proc. of 2017 IEEE International Conference on Autonomic Computing. IEEE, 233–238.

Michael Vierhaus, Sean Bayley, Jane Wngaard, Wandi Xiong, Jinghui Cheng, Joshua Huseman, Robyn R. Lutz, and Jane Cleland-Huang. 2019. Interlocking Safety Cases for Unmanned Autonomous Systems in Shared Airspaces. IEEE Transactions on Software Engineering 45, 7 (2019), 899–918. https://doi.org/10.1109/ TSE.2019.2907595

Michael Vierhaus, Jane Cleland-Huang, Sean Bayley, Thomas Krismayer, Rick Rabiser, and Pau Brunibacher. 2018. Monitoring CPS at runtime-A case study in the US NAV domain. In Proc. of 2018 Euromicro Conference on Software Engineering and Advanced Applications. IEEE, 73–80.

Michael Vierhaus, Md Nafez Al Islam, Ankit Agrawal, Jane Cleland-Huang, and James Mason. 2021. Hazard analysis for human-on-the-loop interactions in uas systems. In Proc. of 29th ACM/SIGAPP Symposium on Applied Computing Conference and Symposium on the Foundations of Software Engineering. ACM, New York, 8–19.

Guy H. Walker, Neville A. Stanton, Paul M. Salmon, and Daniel P. Jenk.

Mathias Barkowsky, Philipp Chrszon, Thomas Springer, and Alexander Schill. 2017. Decontextualized Execution of Adaptations in Distributed Self-Adaptive Software Systems. In Proc. of 11th International Conference on Self-Adaptive and Self-Organizing Systems. IEEE, 111–120.

Robert H Wortham, Andrea Thedorou, and Joanna J Breyson. 2017. Robot transparency: Improving understanding of intelligent behaviour for designers and users. In Annual Conference Towards Autonomous Robot Systems. Springer, 144–149.

Yijun Yu, Danny Barthaud, Blaine A. Price, Arosha K. Bandara, Andrea Zisman, and Basha Nuseibeh. 2019. LiveBox: A Self-Adaptive Forensic-Ready Service for Drones. IEEE Access 7 (2019), 148401–148412. https://doi.org/10.1109/ACCESS. 2019.2942833

Zijian Zhang, Jaspreeet Singh, Uywal Gadiraju, and Avishek Anand. 2019. Disso-

Cleland-Huang, Agrawal, Vierhaus, Murphy, Prieto