Kaa: Policy-based Explorations of a Richer Model for Adjustable Autonomy

Jeffrey M. Bradshaw, Hyuckchul Jung, Shri Kulkarni, Matthew Johnson, Paul Feltovich, James Allen, Larry Bunch, Nathanael Chambers, Lucian Galescu, Renia Jeffers, Niranjan Suri, William Taysom, Andrzej Uszok

Florida Institute for Human and Machine Cognition (IHMC)

40 South Alcaniz Street, Pensacola, FL 32502, USA

{jbradshaw, hjung, skulkarni, mjohnson, pfeltovich, jallen, lbunch, nchambers, lgalescu, rjeffers, nsuri, wtaysom, auszok}@ihmc.us

ABSTRACT

Though adjustable autonomy is hardly a new topic in agent systems, there has been a general lack of consensus on terminology and basic concepts. In this paper, we describe the multi-dimensional nature of adjustable autonomy and give examples of how various dimensions might be adjusted in order to enhance performance of human-agent teams. We then introduce Kaa (KAoS adjustable autonomy), which extends our previous work on KAoS policy and domain services to provide a policy-based capability for adjustable autonomy based on this richer notion of adjustable autonomy. The current implementation of Kaa uses a combination of ontologies represented in OWL and influence-diagram-based decision-theoretic algorithms to determine what if any changes should be made in agent autonomy in a given context. We have demonstrated Kaa as part of ONR-sponsored research to improve naval de-mining operations through more effective human-robot interaction. A brief comparison among alternate approaches to adjustable autonomy is provided.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence – *intelligent agents, multiagent systems.*

General Terms

Performance, Reliability, Human Factors

Keywords

Adjustable Autonomy, Policy, Trust, Human-agent Teamwork, KAoS, Kaa, OWL

1. Introduction

As computational systems with increasing autonomy interact with humans in more complex ways—and with the welfare of the humans sometimes dependent on the conduct of the agents—there is a natural concern that the agents act in predictable ways so that they will be acceptable to people [3]. In addition to traditional concerns for safety and robustness in such systems, there are important social aspects relating to predictability, feedback, order, and naturalness of the interaction that must be attended to [14].

Policies are a means to dynamically regulate the behavior of a system without changing code or requiring the cooperation of the components being governed. They can be used to address the three aspects of trust:

- Through policy, people can precisely express bounds on autonomous behavior in a way that is consistent with their appraisal of an agent's *competence* in a given context.
- Because policy enforcement is handled externally to the agent, *malicious and buggy agents* can no more exempt themselves from the constraints of policy than benevolent and well-written ones can.
- The ability to change policies dynamically means that poorly performing agents can be immediately brought into *compliance* with corrective measures.

Other benefits of policy-based approaches include reusability, efficiency, extensibility, context-sensitivity, verifiability, support for both simple and sophisticated components, and reasoning about component behavior [3].

Researchers have developed platform-independent policy services such as KAoS that enable people to define policies ensuring adequate predictability and controllability of both agents and traditional distributed systems [6; 22]. KAoS has been used in several military and space applications requiring security, robustness, and fault tolerance. A policy-based approach has also been applied to a generic human-agent teamwork model to assure natural and effective interaction in mixed teams of people and robots [2]. In each of these applications, humans have been able to dynamically adjust policies in order to adapt the system to changing situations. However, KAoS lacked a capability for automatically adjustable autonomy, i.e., a means to enable policies to be adjusted without requiring a human in the loop.

Though adjustable autonomy is hardly a new topic in agent systems, here has been a general lack of consensus on terminology and basic concepts. Moreover, current approaches have been based on simplistic assumptions about the nature of human-automation interaction that are generally not informed by

AAMAS'05, July 25-29, 2005, Utrecht, Netherlands.
ACM 1-59593-094-9/05/0007 ...$5.00.

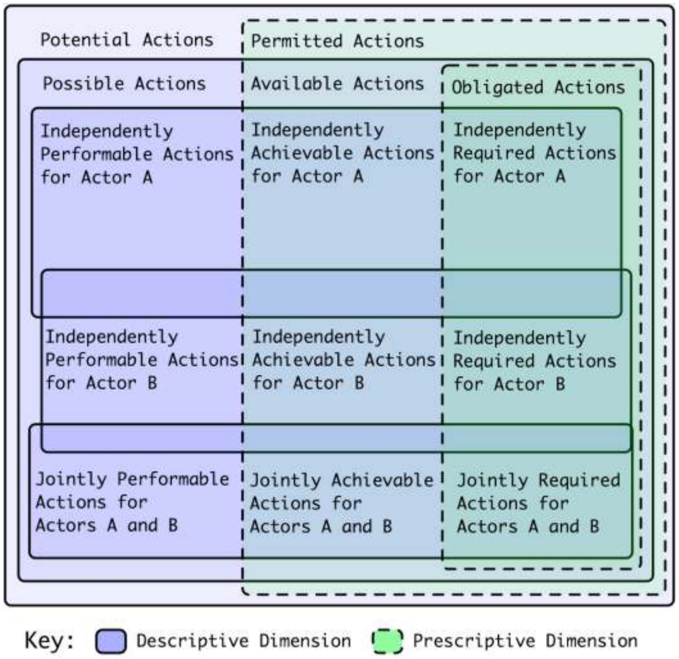

Potential Actions
Permitted Actions
Possible Actions
Available Actions
Obligated Actions
Independently Performable Actions for Actor A
Independently Achievable Actions for Actor A
Independently Required Actions for Actor A
Independently Performable Actions for Actor B
Independently Achievable Actions for Actor B
Independently Required Actions for Actor B
Jointly Performable Actions for Actors A and B
Jointly Achievable Actions for Actors A and B
Jointly Required Actions for Actors A and B
Key:
Descriptive Dimension
Prescriptive Dimension

Fig. 1. Some dimensions of autonomy

the lessons learned from decades of research in human factors and the behavioral and social sciences [14].

In subsequent sections, we describe the multi-dimensional nature of adjustable autonomy as we construe it and give examples of how various dimensions might be adjusted in order to enhance performance of human-agent teams. We then introduce Kaa, the KAoS adjustable autonomy component. Finally, we provide a brief comparison of alternate extant approaches to adjustable autonomy, and offer some concluding remarks.

2. Dimensions of Autonomy

The word "autonomy," which is straightforwardly derived from a combination of Greek terms signifying self-government (*auto-* (self) + *nomos* (law)) has two basic senses in everyday usage. In the first sense, we use the term to denote *self-sufficiency,* the capability of an entity to take care of itself. This sense is present in the French term *autonome* when, for example, it is applied to someone who is successfully living away from home for the first time. The second sense refers to the quality of *self-directedness,* or freedom from outside control, as we might say of a portion of a country that has been identified as an "autonomous region."

Some important dimensions relating to autonomy can be straightforwardly characterized by reference to figure 1. Note that the figure does not show every possible configuration of the dimensions, but rather exemplifies a particular set of relations holding for the actions of a particular set of agents in a given situation. There are two basic dimensions:

- a *descriptive* dimension corresponding to the first sense of autonomy (self-sufficiency) that stretches horizontally to describe the actions an agent in a given context is *capable* of performing; and
- a *prescriptive* dimension corresponding to the second sense of autonomy (self-directedness) running vertically to describe the actions an agent in a given context is allowed to perform or which it must perform by virtue of *policy* constraints in force.

The outermost rectangle, labeled *potential actions,* represents the set of all actions across all situations defined in the ontologies currently in play. Note that there is no requirement that every action in the unknowable and potentially chaotic *universe of actions* that a set of agents may take be represented in the ontology; only those that are of consequence for adjustable autonomy need be included.

The rectangle labeled *possible actions* represents the set of potential actions whose performance by one or more agents is deemed plausible in a given situation [1; 10]. Note that the definition of possibilities is strongly related to the concept of affordances [12; 17], in that it relates the features of the situation to classes of agents capable of exploiting these features in the performance of actions.

Of these possible actions, only certain ones will be deemed *performable* for a given agent in a given situation. *Capability,* i.e., the power that makes an action performable, is a function of the *abilities* (e.g., knowledge, capacities, skills) and *conditions* (e.g., ready-to-hand resources) necessary for an agent to successfully undertake some action in a given context. Certain actions may be *independently performable* by either Agent A or B; other actions can be independently performed by either one or the other uniquely.[1] Yet other actions are *jointly performable* by a set of agents.

Along the prescriptive dimension, declarative policies may specify various *permissions* and *obligations* [9].[2] An agent is *free* to the extent that its actions are not limited by permissions or obligations. *Authorities* may impose or remove involuntary policy constraints on the actions of agents. Alternatively, agents may voluntarily enter into *agreements* that mutually bind them to some set of policies for the duration of the agreement. The *effectivity* of an individual policy specifies when it is in or out of force.

The set of *permitted actions* is determined by *authorization policies* that specify which actions an agent or set of agents is allowed (*positive authorizations* or *A+* policies) or not allowed (*negative authorizations* or *A-* policies) to perform in a given context. The intersection of what is possible and what is permitted delimits the set of *available actions.*

Of those actions that are available to a given agent or set of agents, some subset may be judged to be *independently achievable* in the current context. Some actions, on the other hand, would be judged to be only *jointly achievable.*

Finally, the set of *obligated actions* is determined by *obligation policies* that specify actions that an agent or set of agents is required to perform (*positive obligations* or *O+* policies) or for which such a requirement is waived (*negative obligations* or *O-* policies).[3] *Jointly obligated actions* are those that two or more agents are explicitly required to perform.

[1] Although we show A and B sharing the same set of possible actions, this need not always be the case. Also, note that the range of jointly achievable actions has overlap only with Actor B and not Actor A in the exemplar diagram.

[2] See also the extensive literature on deontic logic.

[3] A negative obligation corresponds to the idea of "you are not obliged to" rather than "you are obliged not to".

3. Adjustable Autonomy

From the perspective of what is written above, adjustable autonomy consists of the ability to dynamically impose and modify constraints that affect the range of actions that the human-agent team can successfully perform in, consistently allowing the highest degrees of useful autonomy while maintaining an acceptable level of trust. We note that is not the case that "more" autonomy is always better[4] since an unsophisticated agent insufficiently monitored and recklessly endowed with unbounded freedom may pose a danger both to others and to itself.

A primary purpose of adjustable autonomy is thus to maintain the system being governed at a sweet spot between convenience (i.e., being able to delegate every bit of an agent's work to the system) and comfort (i.e., the desire to not delegate to the system what it can't be trusted to perform adequately). The coupling of autonomy with policy mechanisms gives the agent maximum freedom for local adaptation to unforeseen problems and opportunities while assuring humans that agent behavior will be kept within desired bounds. If successful, adjustable autonomy mechanisms give the added bonus of assuring that the definition of these bounds can be appropriately responsive to unexpected circumstances.

All this, of course, only complicates the agent designer's task, a fact that has lent urgency and impetus to efforts to develop broad theories and general-purpose frameworks for adjustable autonomy that can be reused across as many agents, domains, and applications as possible. Below we present general description of how autonomy can be adjusted through policies on each dimension.

Adjusting Permissions. A first case to consider is that of adjusting permissions. Reducing permissions may be useful when it is concluded, for example, that an agent is habitually attempting actions that it is not capable of successfully performing—as when a robot continues to rely on a sensor that has been determined to be faulty. It may also be desirable to reduce permissions when agent deliberation about (or execution of) certain actions might incur unacceptable costs or delays.

If, on the other hand, an agent is known to be capable of successfully performing actions that go beyond what it is currently permitted to do, its permissions could be increased accordingly. For example, a flying robot whose duties had previously been confined to patrolling the space station corridors for atmospheric anomalies could be given additional permissions allowing it to employ its previously idle active barcode sensing facilities to take equipment inventories while it is roaming [5].

Adjusting Obligations. On the one hand, "underobligated" agents can have their obligations increased—up to the limit of what is achievable—through additional task assignments. For example, in performing joint action with people, they may be obliged to report their status frequently or to receive explicit permission from a human before proceeding to take some action. On the other hand, an agent should not be required to perform any action that outstrips its permissions, capabilities, or possibilities. An "overcommitted" agent can sometimes have its autonomy adjusted to manageable levels through reducing its current set of obligations. This can be done through delegation, facilitation, or renegotiation of obligation deadlines. In some circumstances, the agent may need to renege on its obligations in order to accomplish higher priority tasks.

Adjusting Possibilities. A highly capable agent may sometimes be performing below its capabilities because of constraints inherent in the current situation. For example, a physical limitation on network bandwidth available through the nearest wireless access point may restrict an agent from communicating at the rate it is permitted and capable of doing.

In some circumstances, it may be possible to adjust autonomy by increasing the set of possibilities available to an agent. For example, a mobile agent may be able to make what were previously impossible faster communication rates possible by moving to a new host in a different location.

Sometimes reducing the set of possible actions provides a powerful means of enforcing restrictions on an agent's actions. For example, an agent that "misbehaved" on the network could be sanctioned and constrained from some possibilities for action by moving it to a host with restricted network access. In both cases, autonomy is adjusted not by directly manipulating the resources themselves, but rather by placing the agent in a situation affording increased possibilities.

Adjusting Capabilities. The capabilities of an agent affect the range of its performable actions. In this sense, the autonomy of an agent can be augmented either by increasing its own independent capabilities or by extending its joint capabilities through access to other agents to which tasks may be delegated or shared. An agent's capabilities can also be affected by changing current conditions (e.g., externally adding or reducing needed resources, or perhaps reallocating one's internal resources and efforts).

An adjustable autonomy service aimed at increasing an agent's capabilities could assist in discovering agents with which an action that could not be independently achieved could be jointly achieved. Or if the agent was hitting the ceiling on some computational resource (e.g., bandwidth, memory), resource access policies could be adjusted to allow the agent to leverage the additional assets required to perform some action. Finally, the service could assist the agent by facilitating the deferral, delegation, renegotiation, or reneging on obligations in order to free up previously committed resources (as previously mentioned in the context of adjusting obligations).

Based on the principal dimensions of autonomy and the possible adjustments described above, we now discuss the implementation of these concepts in Kaa.

4. Kaa: KAoS Adjustable Autonomy

We have developed formalisms and mechanisms for adjustable autonomy and policies that will facilitate effective coordination and mixed-initiative interaction among humans and agents engaged in joint activities. We are doing this in conjunction with a testbed that integrates the various capabilities of heterogeneous systems such as TRIPS, Brahms, and KAoS [2].

KAoS is a collection of componentized policy and domain services.[5] KAoS policy services enable the specification,

[4] In fact, the multidimensional nature of autonomy argues against even the effort of mapping the concept of "more" and "less" to a single continuum. See [4] for an overview of a broad theory of adjustable autonomy and its multi-dimensional nature.

[5] KAoS is compatible with several popular agent frameworks, including Nomads, the DARPA CoABS Grid, the DARPA ALP/UltraLog Cougaar framework, CORBA, Voyager,

management, conflict resolution, and enforcement of semantically-rich policies defined in OWL [21]. On this foundation, we are building Kaa (KAoS adjustable autonomy) a component that permits KAoS to perform automatic adjustments of autonomy consistent with policy.[6]

Assistance from Kaa in making autonomy adjustments might typically be required when it is anticipated that the current configuration of human-agent team members has led to or is likely to lead to failure, and when there is no set of competent and authorized humans available to make the adjustments themselves. Ultimately, the value of performing an adjustment in a given context is a matter of expected utility: the utility of making the change vs. the utility of the status quo.

The current implementation of Kaa uses influence-diagram-based decision-theoretic algorithms to determine what if any changes should be made in agent autonomy. An influence diagram is a belief network extended with special node types for actions and utilities [23]. When invoked, Kaa first builds an influence diagram based on available adjustment options, capabilities/conditions required for the options, and their cost. The utility of various adjustment options (e.g., increases or decreases in permissions and obligations, acquisition of capabilities, proactive changes to the situation to allow new possibilities) is computed with the cost and the risk that are accompanied with each option. Kaa compares the utility of various options, and then—if a change in the status quo is warranted—takes action to implement the recommended alternative.

When evaluating options for adaptively reallocating tasks among team members, Kaa should consider that dynamic role adjustment comes at a cost. Hence, measures of expected utility would ideally be used in the future to evaluate the tradeoffs involved in potentially interrupting the ongoing activities of agents and humans in such situations to communicate, coordinate, and reallocate responsibilities [13].

4.1. A Simple Example: Robot Signaling

One of the most important contributions of more than a decade of research on agent teamwork is the finding that many aspects of effective team behavior rely on a collection of generic coordination mechanisms rather than on deep knowledge of specific application domains [8; 20]. With previous research in agent teamwork, we share the assumption that, to the extent possible, teamwork knowledge should be modeled explicitly and separately from the problem-solving domain knowledge so it can be easily reused across applications. In such an approach, policies for agent safety and security (as well as contextual and culturally sensitive teamwork behavior) can be represented as KAoS policies that enable many aspects of the nature and timing of the agent's interaction with people to be appropriate, without requiring each agent to individually encode that knowledge [3]. Agent designers can then concentrate on developing unique agent capabilities, while assuming that many of the basic rules of effective human-agent coordination will be built into the environment as part of the policy infrastructure.

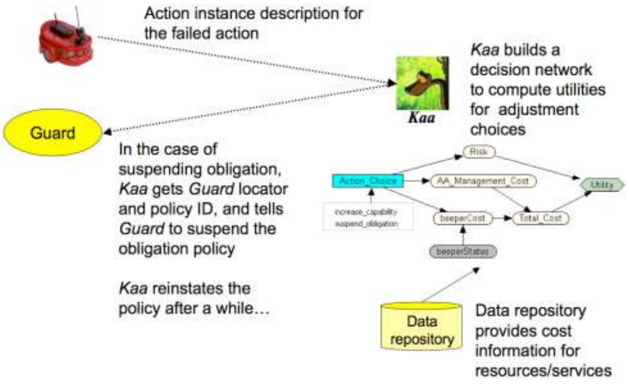

Action instance description for the failed action
Kaa builds a decision network to compute utilities for adjustment choices
Kaa
Guard
In the case of suspending obligation, *Kaa* gets *Guard* locator and policy ID, and tells *Guard* to suspend the obligation policy
Kaa reinstates the policy after a while…
Risk
Action_Choice
AA_Management_Cost
Utility
increase_capability
suspend_obligation
beeperCost
Total_Cost
beeperStatus
Data repository
Data repository provides cost information for resources/services

Fig. 2. Kaa concept of operation for the robot beep failure

As part of this research, we are developing policies to govern various nonverbal forms of expression in software agents and robots [11]. Such nonverbal behaviors are intended to express not only the current state of the agent but also—importantly—to provide rough clues about what it is going to do next. In this way, people can be better enabled to participate with the agent in coordination, support, avoidance, and so forth. In this sense, nonverbal expressions are an important ingredient in enabling human-agent teamwork. A simple example involving a nonverbal expression policy will illustrate a simplified description of how Kaa works.

Assume that a robot's signaling behavior is governed by the following positive obligation policy: *O+: A robot must beep for a few seconds before beginning to move.* The intention of such a policy is to warn others nearby to stay out of the way when a robot is about to move.

Before the robot attempts to move, the robot execution platform, in conjunction with platform-specific KAoS components, requires the robot to ask a KAoS guard responsible for managing local policy enforcement whether the action is authorized.[7] The guard then retrieves and checks the relevant set of policies. In this example, we assume that the guard finds both an authorization policy allowing the robot to move in this context as well as the obligation policy described above.

Under normal circumstances, the obligation policy will first trigger the robot to emit the beep, and then will return the necessary authorization for the robot to move. However, certain states and events, such as a failure of the robot to successfully

Brahms, TRIPS, and SFX.. While initially oriented to the dynamic and complex requirements of software agent applications, KAoS services are also being adapted to general-purpose grid computing and Web Services environments as well [22]. KAoS has been deployed in a wide variety of applications, from coalition warfare and agile sensor feeds, to process monitoring and notification, to robustness and survivability for distributed systems, to semantic web services composition, to human-agent teamwork in space applications, to cognitive prostheses for augmented cognition.

[6] In Rudyard Kipling's Jungle Book, the human boy Mowgli was educated in the ways and secrets of the jungle by Kaa the python. His hypnotic words and stare charmed the malicious monkey tribe that had captured the boy, and Kaa's encircling coils at last "bounded" their actions and put an end to their misbehavior. In a similar way, Kaa attempts to bound the autonomy of agents.

[7] KAoS policy enforcement is described in more detail in. [3].

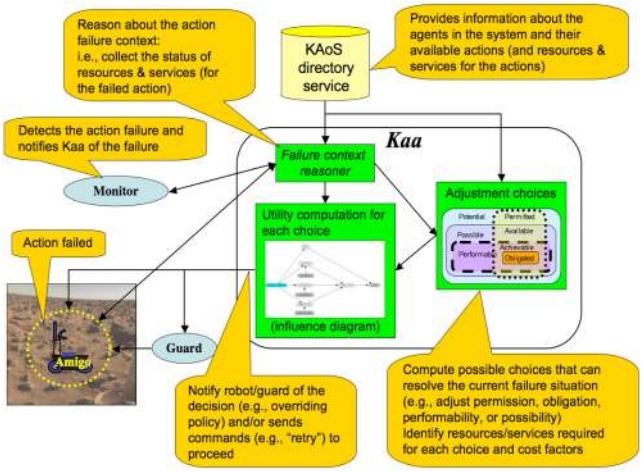

Reason about the action failure context:
i.e., collect the status of resources & services (for the failed action)
KAoS directory service
Provides information about the agents in the system and their available actions (and resources & services for the actions)
Detects the action failure and notifies Kaa of the failure
Kaa
Failure context reasoner
Monitor
Adjustment choices
Potential
Permitted
Possible
Available
Performable
Achievable
Obligated
Utility computation for each choice
(influence diagram)
Action failed
Amigo
Guard
Notify robot/guard of the decision (e.g., overriding policy) and/or sends commands (e.g., "retry") to proceed
Compute possible choices that can resolve the current failure situation (e.g., adjust permission, obligation, performability, or possibility)
Identify resources/services required for each choice and cost factors

Fig. 3. Notional functional architecture for Kaa

sound its obligatory warning, will trigger an attempt by Kaa to intervene in a helpful way.[8]

In such a case, the KAoS-Robot infrastructure[9] creates an action instance description for the failed action and forwards it to Kaa (figure 2). Kaa in turn dynamically constructs an influence diagram based on state-specific information in the action instance description such as the cause of failure and a related policy type combined with information from the KAoS directory service repository (e.g., list of agents with beeping capability for possible delegation).

For instance, if the beeping action is not performable due to a resource (e.g., beeper) failure, the failed action is required by an obligation policy, and there is no other agents with a beeping capability, Kaa excludes the possibility of delegation (i.e., obliging a joint activity with another robot) and determines there are two alternatives: increasing the range of performable actions (by making a non-performable action performable with resource change) vs. decreasing the range of obliged actions. Assume that Kaa determines that temporarily suspending the obligation policy is the best option after considering available alternatives. Then, with this precondition for the move action now removed, the guard can now return its authorization for the move to the robot, and the robot can perform the action. When circumstances permit, Kaa can reinstate the suspended policy.

Figure 3 shows a notional functional architecture for Kaa to handle failures in dynamic and uncertain environment. As shown above, being notified of failure, Kaa reasons about its context and examine possible choices to deal with the situation. Contextual information comes from the failed robot and KAoS directory service, a global data repository. Decision-theoretic reasoning based on influence diagrams computes an optimal choice.

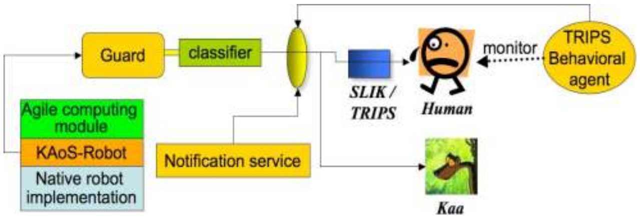


Fig. 4. Relationships of Kaa to external systems

4.2. Application to Naval De-mining Operations

In the Navy's future operations, large numbers of unmanned ground, air, underwater, and surface vehicles will work together, coordinated by ever smaller teams of human operators. A current scenario as a part of Naval Automation and Information Management Technology project is based on a lane clearing operation in shallow water. Using cooperative search algorithms with multiple robots, mine-free lanes are identified in order to allow the landing of amphibious vehicles on the beach. As part of this research we have developed and demonstrated the use of Kaa as described below.

In order to be operationally efficient, effective and useful, the robots must perform complex tasks with considerable autonomy, must work together safely and reliably within policy constraints, must operate flexibly and robustly in the face of intermittent network availability and potentially rapid fluctuation of available infrastructure resources, and must coordinate their actions with each other and with human operators. In addition, the human operator, controlling the actions of many unmanned systems must observe and control them in an intuitive fashion incorporating capabilities for mixed-initiative interaction and adjustable autonomy.

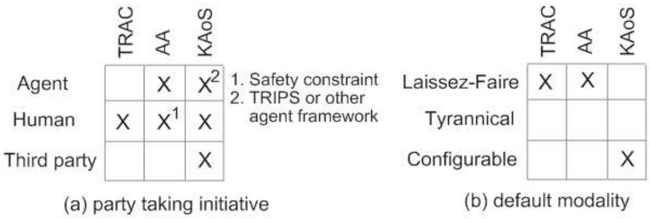
Unlike the simplified example presented in the previous section, in this multi-human/robot mission, either Kaa or a human operator or both can potentially intervene to assist the human-robot team when necessary. For example, given lost network connectivity among robots, agile computing infrastructure[10] frequently tasks an idle robot to move into a position where it can serve as a network relay. However, for one reason or another, a robot may not be authorized by policy to make a given move. Rather than simply turning down the authorization, the guard will forward the request to a classifier. The classifier examines the policy to determine who (if anyone) should be consulted in such circumstances. Normally, the classifier will forward the request to the human, who will decide if the need for restoring network connectivity should override the policy restriction.

Preferences for who should intervene can be expressed on a policy-by-policy basis. Thus, in some situations, the person defining the policy may feel comfortable always letting Kaa handle problems on its own without interrupting the operator. In other situations, the person may only trust the human to intervene. In yet other situations, the person defining the policy may want to give the operator the first opportunity to intervene and only if the operator is too busy to respond will it call on Kaa for help. Finally, a policy may be specified that requires Kaa to

[8] Alternatively, in the future we plan to make Kaa watch for component status in advance and take preemptive action (e.g., imposing negative authorization on actions which require failed resources).

[9] KAoS-Robot provides an ontology-based layer of abstraction for various robot implementations and assists the guard in policy enforcement.

[10] The agile computing module, called FlexFeed, provides the communication and computation framework, including mobile ad-hoc networks, opportunistic resource discovery and exploitation, and flexible, bandwidth-efficient data feeds [19].



	TRAC	AA	KAoS
Agent		X	X[2]
Human	X	X[1]	X
Third party			X

1. Safety constraint
2. TRIPS or other agent framework

(a) party taking initiative

	TRAC	AA	KAoS
Laissez-Faire	X	X	
Tyrannical			
Configurable			X

(b) default modality

Fig 5. Perspective comparison

make the first attempt at resolving any problems, allowing it, however, to call upon a human for help if necessary.

In figure 4, the notification service, in conjunction with the TRIPS[11] behavioral agent, determines the means by which the human should be contacted and the urgency with which it should be presented given an understanding of the state of the human. If after a timeout period the human does not reply, the decision about whether to grant permission can be delegated to Kaa.

In another example, a policy requires the robot to contact the human for help when the robot is not certain about the identification of a mine. With multiple robots moving and many tasks to monitor, it is possible that a robot will find an indeterminate object while the operator is occupied with other tasks. If the operator fails to respond within sufficient time to such a query, a request for help is forwarded to Kaa.

Receiving such a request, Kaa builds an influence diagram where a decision node includes different adjustment alternatives (e.g., removing an obligation, enabling a failed action), and chance nodes represent required resources (e.g., human operator, communication bandwidth), their status, and a measure of utility (e.g., costs, risks, benefits) associated with each alternative.

After ranking alternatives according to their expected utility, Kaa might decide to remove the obligation of contacting the human for help and grant the robot the autonomy it needs to classify the indeterminate object on its own and move on if the operator is not available and the risk of granting autonomy is minimal. Alternatively, Kaa might determine that the best alternative would be to extend the timeout period, up to some maximum allowed it by policy if the operator is only temporarily unavailable (e.g., if the operator is busy in watching a video from a robot which requires finite time). This timeout extension effectively makes a previously unavailable resource (i.e., operator) available. Here, the reasoning about user's situation is enabled by a *subscribe* service in the KAoS Common Services Interface (CSI).[12] Kaa can selectively subscribe and monitor events such as the start or end of video feed as an indirect means of inferring whether the human operator is seated at the workstation.

5. A Comparison of Perspectives

Several groups have grappled with the problem of characterizing and developing practical approaches for implementing adjustable autonomy in deployed systems. Each takes a little different approach and uses similar terminology somewhat differently. It would be helpful to the research community if there were an increased consensus about the concepts and terminology involved.

To characterize a sampling of perspectives and terminology used by various research groups, we will briefly contrast our approach to two other implemented formulations: the SRI TRAC (now SPARK) framework [16] and the Electric Elves agent-based autonomy framework [18]. These two frameworks were compared in [15], making them a convenient choice for further comparison.

Here we will simply consider some of the basic dimensions relating to the adjustment decisions, ignoring for the moment specific features of the frameworks (e.g., analysis tools, user interface, accommodation of heterogeneity) as well as performance and scalability issues: (i) Party taking initiative for adjustment; (ii) Rationale for adjustment; (iii) Type of adjustment; (iv) Default modality; (v) Duration of adjustment; (vi) Party who is final arbiter; (vii)Locus of enforcement.

Party taking initiative for adjustment. In principle, the actual adjustment of an agent's level of autonomy could be initiated either by a human, the agent, or some other software component. Figure 5-(a) illustrates how this is handled in the three frameworks.

TRAC has been characterized as a framework for "user-based adjustable autonomy" in which policies are defined by people. The motivation for these policies is to compensate for limits to agent competence and to allow for personalization.

In contrast, the Electric Elves approach has been characterized as an "agent-based autonomy" (AA) approach where adjustments to autonomy are the result of explicit agent reasoning. A *transfer-of-control* strategy is computed in advance and offline using a Markov decision process (MDP) such that in each possible state the agent knows whether it should make the decision autonomously, ask the user for help, or change its coordination constraints (e.g., inform other agents of a delay).[13]

Since KAoS runs in conjunction with several agent frameworks, the ability for an agent to explicitly reason about autonomy adjustment depends on the particular platform being used. For example, TRIPS allows sophisticated reasoning about these issues, whereas agents built with less capable frameworks may be capable of little or no reasoning of this sort. In addition, KAoS allows humans to define or change policies through a simple GUI. Finally, Kaa, as a selectively trusted third-party, can sometimes make its own adjustments to policy or other dimensions of autonomy.

[11] TRIPS, an interactive planning system, addresses the challenges of providing an effective and natural multimodal interface (including spoken dialog) between the human operator(s) and the robotic platforms [7]. SLIK (Simple Logical Interface to KAoS) is the interface between TRIPS and KAoS.

[12] KAoS CSI provides registration, transport, query, command, subscribe, and policy disclosure services. These services are available to any entities desiring to interact with the robots including internal components (e.g., a deliberative layer) and external entities (e.g., client GUI, other robots or agents).

[13] The theme of social laws has been investigated by the agent research community under two main headings: *norms* and *policies*. Typically, policy-based approaches impose prescriptive constraints on agents externally while norm-based approaches place the responsibility of conformance to social laws on the agents themselves. See [3] for a more extensive discussion of this topic.

Rationale for considering adjustment. Many different factors can constitute the rationale for considering an adjustment. In TRAC, the rationale for modification to policy resides exclusively with the human, whereas in AA it is part of a precomputed set of agent strategies, with choices determined according to a fixed set of agent states.

In KAoS, authorized people or agents can make changes to policy at any time. In addition, any event or state in the world or the ontology that can be monitored by the system could be set up to trigger a self-adjustment process in Kaa. For example, the impetus for Kaa to consider adjustment could be due to the fact that task performance has fallen outside of (or has returned within) some acceptable range. Alternatively, certain events or changes in the state of the environment (e.g., sudden change in temperature), an agent (e.g., agent is performing erratically), or a human (e.g., human is injured; or conversely is now again available to help out) can provide the rationale for adjustment.

Type of adjustment. As outlined in section 2, adjustments to autonomy can be of several types: capabilities (more or less), possibilities, authorizations (positive or negative, more or less), and obligations (positive or negative, more or less). TRAC allows policies to be defined for three sorts of positive obligations: obligations to ask permission from a human supervisor for certain actions *(permission requirements),* obligations to defer decisions about certain actions to a human supervisor *(consultation requirements),* and obligations to accomplish specified tasks in a certain manner *(strategy preference guidance).*

AA allows for agents to determine strategy for and require itself to act upon three kinds of obligations: asking the user for help, making the decision itself, and performing a coordinating action. Additionally, AA allows the human to represent two kinds of safety constraints. The first kind is a sort of negative authorization that can prevent agents from taking an action, while the second kind is a sort of positive obligation requiring them to take a certain action.

KAoS is designed to allow adjustment along any of the dimensions described in section 2. While it is fair to characterize TRAC as an implementation of an approach to adjustable autonomy, the ability to allow humans to define policies in TRAC is different from the *automatically* adjustable autonomy implemented by AA and Kaa.

Default modality. Within a given policy-governed environment, a default modality for authorization policies must be established. In a permissive environment, it is usually easiest to set a permissive default modality and to define a small number of negative authorization policies for any actions that are restricted. In a restrictive environment, the opposite is usually true.

TRAC and AA implement a fixed *laissez-faire* modality where anything is permitted that is not specifically forbidden by policy as illustrated in figure 5-(b). KAoS implements a per-domain-configurable default modality. In other words, for a given application, agents in one domain (i.e., user-defined group) might be subject to a *laissez-faire* default modality, while agents in another domain might be simultaneously subject to a *tyrannical* one (i.e., where everything is forbidden that is not specifically permitted). Modality dominance constraints are used to determine which modality takes priority in the case of agents belonging to more than one domain.

Duration of adjustment. When constraints in any of the three frameworks are put into force or removed, the adjustment to the agent's level of autonomy is changed indefinitely. However, KAoS additionally allows an authorized human or trusted software component such as Kaa to override current policy on a per event basis (e.g., exceptionally allow some action just this once) or for a certain fixed length of time.

Party who is final arbiter. For some actions, there is the question of who is the final authority in case of disagreement between some person and the machine. For example, a policy may allow a human to take manual control of an Unmanned Aerial Vehicle (UAV) if there is a risk of it crashing. On the other hand, the UAV may have a policy preventing a human from deliberately steering it to a forbidden area.

In TRAC, this issue does not arise because it is not possible to explicitly represent authorization policies. In AA and KAoS, authorization policies can definitively limit the kinds of actions that the agent can perform. Additionally, in KAoS, authorization policies can be defined to limit human actions as well.

Locus of enforcement. In both TRAC and AA, the interpretation of policies is integrated with the agent's planning and decision-making process and the agent itself is entrusted with the enforcement of policy.

While KAoS does not prohibit agents from optimizing their behavior through reasoning about policies (to the extent that policy disclosure is itself permitted by policy), the responsibility for enforcement is given to independent control elements of the trusted infrastructure. In this way, enforcement of policy remains effective even when agents themselves are buggy, malicious, poorly designed, or unsophisticated. This is essential if policies are to be regarded as something binding on agents, rather than just good advice. However, there is no reason why KAoS enforcement mechanisms could not be used in a complementary way with the agent-based enforcement in TRAC and AA.

6. Discussion and Conclusions

We believe that policy-based approaches hold great promise in compensating for limitations of competence, benevolence, and compliance of agent systems. Ontology-based approach in KAoS enables flexibility, extensibility, and power for policy specification, modification, reasoning, and enforcement. As the work in this paper demonstrates, the application of policy is now being extended beyond narrow technical concerns, such as security, to social aspects of trust and human-agent teamwork. As research results bring greater experience and understanding of how to implement self-regulatory mechanisms for agent systems, we expect a convergence and a concomitant increase of synergy among researchers with differing perspectives on adjustable autonomy and mixed-initiative interaction.

One of the biggest differences between KAoS and the two other approaches compared in section 5 (TRAC and AA) is in where the locus of initiative for adjustment and enforcement lies. Though allowing policy to be disclosed and reasoned about by agents when required, KAoS policy services aim to assure that policy can be relied on whether or not the agents themselves can be trusted to do the right thing. In contrast, TRAC and AA depend exclusively on the agents to monitor and enforce their own actions. When humans are both in the loop and presumed to be more capable and trustworthy than the agents themselves, the KAoS approach would seem to have merit. However, this solution is insufficient for those situations where the human is unavailable or is judged to be less competent or trustworthy than the machine for dealing with an adjustable autonomy issue.

Our objective in developing Kaa is to address this issue: to enable reasoning about relevant tradeoffs and the taking of appropriate measures in situations where the best action may not be the blind following of a policy but rather the automatic adjustment of one or more dimensions of autonomy. While we have reservations about approaches that *cannot* enforce human-defined policies independently of a potentially untrustworthy or incompetent agent's code, we also have qualms about approaches lacking the means to adjust policies and policy-related autonomy dimensions that have been clearly demonstrated to be *ineffective* in a given context of application. Adding the capabilities of Kaa to KAoS services is intended to achieve the best of both worlds: trustworthy adjustable autonomy regardless of the trustworthiness of agent code.

7. Acknowledgements

The authors gratefully acknowledge the support of the DARPA EPCA (CALO), DAML, and Ultra*Log Programs, the NASA X-Enterprise and Intelligent Systems Programs, the Army Research Lab, and the Office of Naval Research while preparing this paper.

8. References

[1] Barwise, J., & Perry, J. (1983). *Situations and Attitudes*. Cambridge, MA: MIT Press.

[2] Bradshaw, J. M., Acquisti, A., Allen, J., Breedy, M. R., Bunch, L., Chambers, N., Feltovich, P., Galescu, L., Goodrich, M. A., Jeffers, R., Johnson, M., Jung, H., Lott, J., Olsen Jr., D. R., Sierhuis, M., Suri, N., Taysom, W., Tonti, G., & Uszok, A. (2004). Teamwork-centered autonomy for extended human-agent interaction in space applications. *AAAI 2004 Spring Symposium*. Stanford University, CA, AAAI Press,

[3] Bradshaw, J. M., Beautement, P., Breedy, M. R., Bunch, L., Drakunov, S. V., Feltovich, P. J., Hoffman, R. R., Jeffers, R., Johnson, M., Kulkarni, S., Lott, J., Raj, A., Suri, N., & Uszok, A. (2004). Making agents acceptable to people. In N. Zhong & J. Liu (Ed.), *Intelligent Technologies for Information Analysis: Advances in Agents, Data Mining, and Statistical Learning.* (pp. 361-400). Berlin: Springer Verlag.

[4] Bradshaw, J. M., Feltovich, P., Jung, H., Kulkarni, S., Taysom, W., & Uszok, A. (2004). Dimensions of adjustable autonomy and mixed-initiative interaction. In M. Nickles, M. Rovatsos, & G. Weiss (Ed.), *Agents and Computational Autonomy: Potential, Risks, and Solutions. Lecture Notes in Computer Science, Vol. 2969.* (pp. 17-39). Berlin, Germany: Springer-Verlag.

[5] Bradshaw, J. M., Sierhuis, M., Acquisti, A., Feltovich, P., Hoffman, R., Jeffers, R., Prescott, D., Suri, N., Uszok, A., & Van Hoof, R. (2003). Adjustable autonomy and human-agent teamwork in practice: An interim report on space applications. In H. Hexmoor, R. Falcone, & C. Castelfranchi (Ed.), *Agent Autonomy.* (pp. 243-280). Kluwer.

[6] Bradshaw, J. M., Uszok, A., Jeffers, R., Suri, N., Hayes, P., Burstein, M. H., Acquisti, A., Benyo, B., Breedy, M. R., Carvalho, M., Diller, D., Johnson, M., Kulkarni, S., Lott, J., Sierhuis, M., & Van Hoof, R. (2003). Representation and reasoning for DAML-based policy and domain services in KAoS and Nomads. *Proceedings of the Autonomous Agents and Multi-Agent Systems Conference (AAMAS 2003).* Melbourne, Australia, New York, NY: ACM Press,

[7] Chambers, N., Allen, J., & Galescu, L. (2005). A dialogue-based approach to multi-robot team control. *Proceedings of Third International Naval Research Labs Multi-Robot Systems Workshop.* Washington, D.C.,

[8] Cohen, P. R., & Levesque, H. J. (1991). *Teamwork.* Technote 504. Menlo Park, CA: SRI International, March.

[9] Damianou, N., Dulay, N., Lupu, E. C., & Sloman, M. S. (2000). *Ponder: A Language for Specifying Security and Management Policies for Distributed Systems, Version 2.3.* Imperial College of Science, Technology and Medicine, Department of Computing, 20 October 2000.

[10] Devlin, K. (1991). *Logic and Information.* Cambridge, England: Cambridge University Press.

[11] Feltovich, P., Bradshaw, J. M., Jeffers, R., Suri, N., & Uszok, A. (2004). Social order and adaptability in animal and human cultures as an analogue for agent communities: Toward a policy-based approach. In *Engineering Societies in the Agents World IV. LNAI 3071.* (pp. 21-48). Berlin, Germany: Springer-Verlag.

[12] Gibson, J. J. (1979). *The Ecological Approach to Visual Perception.* Boston, MA: Houghton Mifflin.

[13] Horvitz, E. J. (1989). Reasoning about beliefs and actions under computational resource constraints. In L. Kanal, T. S. Levitt, & J. F. Lemmer (Ed.), *Uncertainty in Artificial Intelligence, Volume 3.* Amsterdam: North-Holland.

[14] Klein, G., Woods, D. D., Bradshaw, J. M., Hoffman, R., & Feltovich, P. (2004). Ten challenges for making automation a "team player" in joint human-agent activity. *IEEE Intelligent Systems*, 19(6), 91-95.

[15] Maheswaran, R. T., Tambe, M., Varakantham, P., & Myers, K. (2004). Adjustable autonomy challenges in personal assistant agents: A position paper. In M. Klusch, G. Weiss, & M. Rovatsos (Ed.), *Computational Autonomy.* (pp. in press). Berlin, Germany: Springer.

[16] Myers, K., & Morley, D. (2003). Directing agents. In H. Hexmoor, C. Castelfranchi, & R. Falcone (Ed.), *Agent Autonomy.* (pp. 143-162). Dordrecht, The Netherlands: Kluwer.

[17] Norman, D. A. (1988). *The Psychology of Everyday Things.* New York: Basic Books.

[18] Scerri, P., Pynadath, D., & Tambe, M. (2002). Adjustable autonomy for the real world. In H. Hexmoor, C. Castelfranchi, & R. Falcone (Ed.), *Agent Autonomy.* (pp. 163-190). Dordrecht, The Netherlands: Kluwer.

[19] Suri, N., Carvalho, M., & Bradshaw, J. M. (2004). Proactive resource management for agile computing. C. Bryce & G. Czaijkowski (Ed.), *Proceedings of the Tenth Annual ECOOP Workshop on Mobile Object Systems and Resource-Aware Computing.* Oslo, Norway,

[20] Tambe, M., Shen, W., Mataric, M., Pynadath, D. V., Goldberg, D., Modi, P. J., Qiu, Z., & Salemi, B. (1999). Teamwork in cyberspace: Using TEAMCORE to make agents team-ready. *Proceedings of the AAAI Spring Symposium on Agents in Cyberspace.* Menlo Park, CA, Menlo Park, CA: The AAAI Press,

[21] Tonti, G., Bradshaw, J. M., Jeffers, R., Montanari, R., Suri, N., & Uszok, A. (2003). Semantic Web languages for policy representation and reasoning: A comparison of KAoS, Rei, and Ponder. In D. Fensel, K. Sycara, & J. Mylopoulos (Ed.), *The Semantic Web—ISWC 2003. Proceedings of the Second International Semantic Web Conference, Sanibel Island, Florida, USA, October 2003, LNCS 2870.* (pp. 419-437). Berlin: Springer.

[22] Uszok, A., Bradshaw, J. M., Johnson, M., Jeffers, R., Tate, A., Dalton, J., & Aitken, S. (2004). KAoS policy management for semantic web services. *IEEE Intelligent Systems*, 19(4), 32-41.

[23]. Howard, R.A. and J.E. Matheson, *Influence diagrams*, in *Readings on the Principles and Applications of Decision Analysis*, R.A. Howard and J.E. Matheson, Editors. 1984, Strategic Decisions Group: Menlo Park, California. p. 719-762.