

# Integrating Detailed Real-Time Agent Control with Control for Large-Scale, Open, Multi-Agent Systems

Tracking # 461

Thomas A. Wagner  
Honeywell Laboratories

<http://mas.cs.umass.edu/~wagner>  
wagner\_tom@htc.honeywell.com

Victor R. Lesser  
University of Massachusetts

<http://mas.cs.umass.edu/~lesser>  
lesser@cs.umass.edu

## Abstract

Software agents operating in large-scale, open and dynamic, multi-agent systems must be able to reason about how their candidate tasks relate to those being performed by other agents and how tasks relate to satisfying organizational goals and objectives. Reasoning at this level, however, is inherently different than the class of reasoning required to evaluate and coordinate the detailed domain actions of each agent. In this paper we define an approach for integrating detailed agent control, in the form of TÆMS, Design-to-Criteria agent scheduling, and GPGP coordination, with the *MQ* framework for organizational level control.

## 1. INTRODUCTION

Our research community is showing significant progress in making the agent computing paradigm a reality. However, there are research questions that must be addressed in order for investment in agent computing research to pay full dividends. Consider the requirements and characteristics of the problem space. One long term objective is to create agent-based *open, large-scale*, information and computational systems that exist in dynamic environments and are flexible, adaptable, robust, persistent, and autonomous. Now, consider the implications of this. Operating in dynamic open environments is a hard problem because it means that the tasks on which an agent is operating may change frequently due to new goals, new requests, or new collaborations with other agents. Interactions between tasks may change frequently as well for the same reasons. The agents themselves may come and go from the network and be periodically unavailable. These conditions often act to thwart agent technologies that rely on detailed predictability or static properties of the problem space. In our work, we take the view that openness leads to a requirement for soft real-time agent problem solving so that agents can respond to change and unexpected outcomes online.

Moving the scale of multi-agent systems from small groups to large groups, e.g., tens of thousands, throws two other problems into the mix: *increased interaction overhead* and *situational complexity*. The term interaction overhead denotes the increase in communication between agents required to detect interactions in their problem solving and to coordinate their activities, i.e., it denotes the sheer volume of message traffic and problem solving required to evaluate the messages. This is being dealt with by imposing organizational structure on the agents so that they do not all communicate and by creating coordination and communication technologies that are adjustable. The other issue is increased situational complexity. At the conceptual level, when agents are situated in a large open environment, and organizational structure is imposed upon them, they have different organizational objectives and they must reason about how their problem solving relates to satisfying their multiple, and possibly conflicting, organizational objectives.

We have developed the *motivational quantities (MQ)* framework to address the situational complexity of agents deployed in large-scale MAS. The *MQ* framework has been documented in [25, 27] and is being used to organize agents and to support the study of ranges of cooperative behavior in organized agents and the implications of this to agent negotiation [29, 28]. The *MQ* framework addresses situational complexity from two directions: 1) modeling or *representing* the agent's candidate activities and the necessary attributes of the organizational situation in which activities are located (e.g., how a given task relates to an organizational objective),

and, 2) *reasoning* about the tasks to determine a course of action for the agent. This reasoning process determines which tasks the agent should perform, and when, by valuing candidate tasks in light of the agent's situational complexity or organizational context and by considering resource limitations and deadlines on the tasks.

It is important to emphasize that this research pertains to complex problem solving agents, e.g., [14, 13] and the BIG Information Gathering Agent [17], where the agents are situated in an environment, able to sense and effect, and have explicit representations of candidate tasks and explicit representations of different ways to go about performing the tasks. Additionally, tasks are quantified or have different performance characteristics and, following in the thread of complex problem solving [8] there are relationships between the tasks. The implications are that tasks cannot be reasoned about independently and that the value or utility of particular tasks differs depending on the context. We call the process of reasoning about which tasks to perform, when, with what resources, how or in what fashion, and with whom to coordinate, the *local agent control* problem. The term "local" is used in this expression because agency, as we use it, denotes an autonomous distributed problem solving entity. In our work, there is no global picture of all activities being carried out by all agents nor are the agents situated in specialized, tightly coupled environments like Tambe's teams [22] or robotic soccer.

Though the *MQ* framework is proving appropriate for reasoning about an agent's candidate activities from a high-level, it lacks a detailed view of the domain problem solving processes being performed by the local agent. For example, the *MQ* framework does not model and quantify interactions between tasks – other than to note that one task may produce an artifact necessary for the performance of another task. The implication of this high-level view is that the *MQ* framework can select a course of action for the agent that may not be desirable or even feasible once all of the details of the domain problem solving process are examined and detailed coordination between the agents takes place. There are two reasons for this: 1) the *MQ* framework focuses on the situational complexity that arises in large organizations of agents, it does not focus on the details of domain problem solving, 2) until the details are worked out between all the involved agents, the local agent does not actually know the exact temporal and resource constraints of candidate tasks and thus it cannot fully compute their value or determine feasibility at an organizational level. To address these issues so that agents can reason about both detailed problem solving and the situational complexity present in large MAS, an integration path between the *MQ* level and more detailed reasoning/coordination processes is required.

In our work, detailed agent problem solving is performed via the TÆMS task modeling framework [7, 6, 16], GPGP coordination [6], and Design-to-Criteria (DTC) real-time agent scheduling [19, 24, 26, 23]. TÆMS, GPGP, and DTC are mature research artifacts and have been successfully reused in many applications. However, TÆMS (Section 3) is not suited to addressing the situational complexity that arises when agents are deployed in larger groups or in open environments. One of the fundamental limitations of TÆMS is that it is a static representation of an agent's problem solving process at a given instant in time. It is, in essence, a snapshot of the options available to the agent and a snapshot of their characteris-

tics. In our applications, generally, when the situation changes and the characteristics of tasks (used to determine utility) change, the domain problem solver emits a new TÆMS task structure that reflects the current (changed) situation. Another limitation is that in TÆMS, action performance produces *quality* which then propagates throughout the entire graph-like structure in ways that is intended to model distributed problem solving as in a distributed interpretation problem. The formal details of TÆMS are in [7]. While this view is appropriate for reasoning about interrelated domain problem solving activities at a detailed level, it is not readily used to model concepts like tasks that contribute to one organizational objective while being detrimental to another. TÆMS also does not adequately support concepts like the value of forming a commitment with another agent or the penalty for decommitting from an activity once a commitment is formed.

The *MQ* framework (Section 2) addresses these limitations by representing tasks and actions at a different level of abstraction and with a different emphasis. The *MQ* framework also uses state to achieve “automatic” changes in task valuation or utility (unlike the static view taken in TÆMS). The *MQ* framework describes tasks in many different attribute dimensions so that we can model tasks contributing to, or detracting from, different objectives to different degrees. While control at the TÆMS level pertains to detailed evaluation of domain problem solving activities of an agent, control at the *MQ* level pertains to high-level valuation of candidate tasks based on an understanding of the relationship between tasks and organizational objectives. In other words, in the *MQ* framework, task value is determined not only by the intrinsic properties of tasks, but by the benefits and costs of the intrinsic properties as determined by the agent’s current organizational situation.

In this paper we define the process by which these detailed agent technologies (TÆMS, GPGP, DTC) are integrated with *MQ* level reasoning. We also identify the necessity of a two way iterative flow between *MQ* level control and the detailed agent control level. Due to the inherent complexities of the issues involved, in the sections that follow we summarize the *MQ* framework and detailed control technologies. The integration techniques are then presented and key issues identified.

## 2. SUMMARY OF THE MQ FRAMEWORK FOR LARGE-SCALE MAS

In the *MQ* framework, all tasks have value or a *motivation* for performing the task where the value is determined both by the value of the task and by the importance of the organizational objective with which the task is associated (and the current state of goal achievement). This enables the agent to compare and value tasks that are associated with different organizational goals, or tasks that are detrimental to one organizational goal while having positive benefit to a different organizational goal, or tasks associated with different organizations entirely, or tasks motivated by self-interested reasons to cooperative reasons. The *MQ* framework quantifies these different underlying motivational factors and provides the means to compare them via a multi-attributed utility function. Definitions:

**Agents** are autonomous, heterogeneous, persistent, computing entities that have the ability to choose which tasks to perform and when to perform them. Agents are also rationally bounded, resource bounded, and have limited knowledge of other agents. Additionally:

- Each agent has a set of *MQs* or *motivational quantities* that it tracks and accumulates. *MQs* represent progress toward organizational goals. *MQs* are produced and consumed by task performance where the consumption or production properties are dependent on the context. For example, two agents interacting to achieve a shared organizational goal may both see an increase in the same local *MQ* levels as progress is made (this is not a zero sum game), whereas agents interacting to satisfy different goals may each obtain different types and quantities of *MQs* from the same interaction.
- Not all agents have the same *MQ* set. However, for two agents to form a commitment to a specific course of action, they must have at

least one *MQ* in common (or have the means for forming an *MQ* dynamically). If they do not have an *MQ* in common, they lack any common goals or objectives and lack any common medium of exchange.

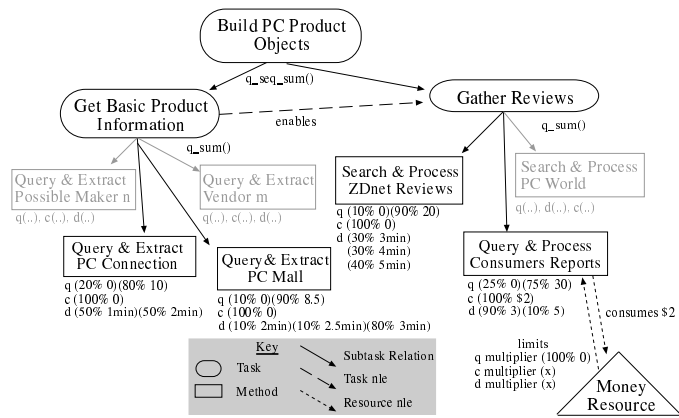
- For each  $MQ_i$  belonging to an agent, it has a preference function or utility curve,  $U_{f_i}$ , that describes its preference for a particular quantity of the  $MQ$ , i.e.,  $\forall MQ_i, \exists U_{f_i}()$  such that  $U_{f_i}(MQ_i) \mapsto U_i$  where  $U_i$  is the utility associated with  $MQ_i$  and is not directly interchangeable with  $U_j$  unless  $i = j$ . Different agents may have different preferences for the same  $MQ_i$ . Preferences in the framework are defined by the relation between task performance and organizational goals or directives.
- An agent’s overall utility at any given moment in time is a function of its different utilities:  $U_{agent} = \gamma(U_i, U_j, U_k, \dots)$ . We make no assumptions about the properties of  $\gamma()$ , only that it enables agents to determine preference or dominance between two different agent states with respect to  $MQs$ .
- For simplicity of presentation, let us assume that  $\gamma()$  is not a multivariate utility function and instead that for each  $U_i$  there is an associated function  $\omega_i()$  that translates  $MQ$  specific utility into the agent’s general utility type, i.e.,  $\forall U_i, \exists \omega_i()$  such that  $\omega_i(U_i) \mapsto U_{agent}$ . Thus  $U_{agent}$  may take the form of  $U_{agent} = \sum_{i=0}^n \omega_i(U_i)$ . (Note:  $\omega_i()$  could be combined with  $U_{f_i}()$ . These are partitioned for mapping different organizational influences.)
- Change in agent utility, denoted  $\Delta U_{agent}$ , is computed through changes to the individual utilities,  $U_i, U_j$ , etc. Let  $U_i$  denote the utility associated with  $MQ_i$  before the quantity of the  $MQ$  changes (e.g., as the result of task performance). Let  $U'_i$  denote the utility associated with the changed  $MQ$  quantity. The change in overall utility to the agent, in this simplified model, is expressed as  $\Delta U_{agent} = |\sum_{i=0}^n \omega_i(U'_i) - \omega_i(U_i)|$

**MQ Tasks** are abstractions of the primitive actions that the agent may carry out. *MQ* tasks:

- May have deadlines,  $deadline_i$ , for task performance beyond which performance of said task yields no useful results. This temporal constraint, as with the one following, is particularly important for multi-agent coordination and temporal sequencing of activities over interactions.
- May have earliest start times,  $start_i$ , for task performance before which performance of said task yields no useful results.
- Each *MQ* task consists of one or more *MQ* alternatives, where one alternative corresponds to a different performance profile of the task, i.e., one different way to go about performing the task. Each alternative:
  - Requires some time or duration to execute, denoted  $d_i$ .
  - Produces some quantity of one or more *MQs*, called an *MQ production set* (*MQPS*), which is denoted by:  $MQPS_{i,j,k} = \{q_i, q_j, q_k, \dots\}$ , where  $\forall i, q_i \geq 0$ . These quantities are *positive* and reflect the benefit derived from performing the task, e.g., progress toward a goal or the production of an artifact that can be exchanged with other agents.
  - Akin to the *MQPS*, tasks may also consume quantities of *MQs*. The specification of the *MQs* consumed by a task is called an *MQ consumption set* and denoted  $MQCS_{i,j,k} = \{q_i, q_j, q_k, \dots\}$ , where  $\forall i, q_i \leq 0$ . Consumption sets model tasks consuming resources, or being detrimental to an organizational objective, or agents contracting work out to other agents, e.g., paying another agent to produce some desired result or another agent accumulating favors or good will as the result of task performance. Consumption sets are the *negative* side of task performance.
  - All quantities, e.g.,  $d_i, MQPS, MQCS$ , are currently viewed from an expected value standpoint.

Space limitations preclude a full presentation of the model. This brief summary lacks definitions and properties necessary to actually build the framework and to use it in agents. This summary also lacks some of the motivations behind these design decisions. For more information, interested readers are advised to consult [25, 27].

The model relates to other recent work in the multi-agent community, such as agents interacting via obligations [2], or notions



**Figure 1: Simplified Subset of an Info. Gathering Task Structure**

of social commitment [5], but it differs in its quantification of different concerns and its dynamic, contextual, relative, evaluation of these. The model resembles MarCon [18] as the different degrees-of-satisfaction afforded by the *MQ* model is akin to MarCon’s constraint optimization approach, and MarCon too deals with utilities / motivations that cannot always be commingled. MarCon, however, views constraints as agents, assigning particular roles to particular agents, and the issue of which tasks to perform do not enter into the problem space.

### 3. SUMMARY OF DETAILED AGENT CONTROL COMPONENTS

In our research detailed agent control is carried out by a variety of different components. The three primary technologies are: 1) the TÆMS task modeling language that is used to represent an agent’s domain problem solving options, 2) the Design-to-Criteria (DTC) agent scheduler that analyzes TÆMS task structures and decides on a course of action for the agent, and 3) the GPGP coordination module which enables multiple agents to coordinate their activities and joint problem solve.

The components are domain independent. In our work, a domain problem solver, planner, or legacy system describes its problem solving options via TÆMS and the remainder of the components, e.g., DTC and GPGP, operate on the TÆMS task structures. This simplifies the process of agent construction as the legacy system or domain expert can effectively be wrapped through TÆMS to create an agent capable of meeting real-time deadlines, meeting resource limitations, performing trade-off analysis, and coordinating its activities with other agents. This approach was used in the BIG [17] information gathering agent to integrate a blackboard problem solver.

With respect to other approaches to agent control, e.g., BDI-based [20, 4] problem solvers or the *MQ* framework, the TÆMS-based tools operate at a different level of detail. The general idea is that DTC and GPGP perform detailed *feasibility analysis* and *implementation* of high-level goals and tasks selected by other components.

**TÆMS** – TÆMS (Task Analysis, Environment Modeling, and Simulation) [7] is a domain independent task modeling framework used to describe and reason about complex problem solving processes. TÆMS models are hierarchical abstractions of problem solving processes that describe alternative ways of accomplishing a desired goal; they represent major tasks and major decision points, interactions between tasks, and resource constraints but they do not describe the intimate details of each primitive action. All primitive actions in TÆMS, called *methods*, are statistically characterized via discrete probability distributions in three dimensions: quality, cost and duration. Uncertainty in each of these dimensions is implicit in the performance characterization – thus agents can reason about the certainty of particular actions as well as their quality, cost, and duration trade-offs. The uncertainty representation is also applied to task interactions like enablement, facilitation and hindering effects, e.g., “10% of the time facilitation will increase the quality by 5% and 90% of the time it will increase the quality by 8%.”

To illustrate, consider Figure 1, which is a conceptual, simplified sub-graph of a task structure emitted by the BIG [17] information gathering agent; it describes a portion of the information gathering process. The top-level task is to construct product models of retail PC systems. It has two subtasks, *Get-Basic* and *Gather-Reviews*, both of which are decomposed into methods, that are described in terms of their expected quality, cost, and duration. The *enables* arc between *Get-Basic* and *Gather* is a non-local-effect (nle) or task interaction; it models the fact that the review gathering methods need the names of products in order to gather reviews for them. Other task interactions modeled in TÆMS include: *facilitation*, *hindering*, *bounded facilitation*, *sigmoid*, and *disablement*. Task interactions must be modeled to support coordination as they identify instances in which tasks assigned to different agents are interdependent.

Returning to the example, *Get-Basic* has two methods, joined under the *sum()* quality-accumulation-function (*qaf*), which defines how performing the subtasks relate to performing the parent task. In this case, either method or both may be employed to achieve *Get-Basic*. The same is true for *Gather-Reviews*. The *qaf* for *Build-PC-Product-Objects* is a *seq.sum()* which indicates that the two subtasks must be performed, in order, and that their resultant qualities are summed to determine the quality of the parent task; thus there are nine alternative ways to achieve the top-level goal in this particular sub-structure. In general, a TÆMS task structure represents a family of plans, rather than a single plan, where the different paths through the network exhibit different statistical characteristics or trade-offs.

**The Design-to-Criteria Scheduler** – Given a family of plans, the problem then is to decide what to do. The Design-to-Criteria (DTC) scheduler is the agent’s local expert on making control decisions. The scheduler’s role is to consider the possible domain actions enumerated by the domain problem solver (via TÆMS) and choose a course of action that best addresses: 1) the local agent’s goal criteria (its preferences for certain types of solutions), 2) the local agent’s resource constraints and environmental circumstances, and 3) the non-local considerations expressed by the GPGP coordination module. The general idea is to evaluate the options in light of constraints and preferences from many different sources and to find a way to achieve the selected tasks that best addresses all of these. DTC’s problem space is thus a resource-bounded hybrid planning/scheduling problem – DTC must determine which tasks are feasible or appropriate given the agent’s objectives, and for each of these, decide how to perform each task, determine appropriate orderings, and continuously evaluate selections to make sure that deadlines are met, resource constraints are not broken, and so forth. Additional details on DTC are located in [19, 24, 26, 23].

For the purposes of this paper, it is necessary to describe DTC’s goal directed problem solving behavior as it plays a role in the integration process. In general, there are an exponential number of possible schedules for a given TÆMS task structure and in practice this is a real issue. To help control the combinatorics, one of the techniques used in DTC is goal, or in this case, “criteria,” directed problem solving. In addition to a set of TÆMS task structures, DTC takes as input a specification of the agent’s objective function. This objective function defines a set of goal criteria and DTC problem solves to produce solutions that adhere to the desired characteristics. For instance, if the agent needs an even trade-off between quality, cost, and duration, the criteria would specify this. If, in a different situation, the agent needed to improve quality certainty while staying under a particular time deadline, the criteria would also specify this. In each case, DTC will produce a different set of solutions. This is how we obtain custom, targetable, control for the agent and how the agent can respond adaptively to different circumstances.

The criteria is defined using a *slider* metaphor. Conceptually, the agent, or a human user, defines the relative importance of quality, cost, and duration in each of these dimensions using a slider associated with each. The sliders are shown graphically in Figure 2. The specification mechanism breaks down relative preferences into different groups, which are represented as slider banks in the figure. The raw goodness group (leftmost slider group) specifies relative preference between quality, cost, and duration. Moving to the right, the next slider group is where the agent or user can specify a desired quality threshold or set limits on duration and cost. The group that is 3rd from the left specifies the relative importance of quality-certainty, duration-certainty, and cost-certainty. The group that is 4th from the left is where a user/agent can set desired certainty thresholds in each dimension and the rightmost group, the meta group, defines the relative importance of the previous four banks of sliders. The entire slider mechanism is simply a way to define a utility or preference function [23] for DTC. DTC then optimizes its process to meet the specified objectives.

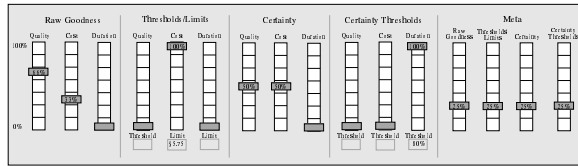


Figure 2: A Slider Set Describing Particular Criteria

**The GPGP Coordination Module** – DTC and TÆMS alone can be used to create an individual agent that can reason about trade-offs, meet deadlines, and adjust problem solving to address different resource allocations. In order for said agent to coordinate its activities with other agents, the GPGP module must be added to the mix. GPGP (Generalized Partial Global Planning) is the agent’s tool for interacting with other agents and coordinating joint activity. GPGP is a modularized, domain independent, approach to scheduling-centric coordination. In GPGP, coordination *modulates* local control by posting constraints on an agent’s local DTC scheduler. The GPGP coordination module is responsible for communicating with other agents and forming (and sometimes breaking) task related *commitments* with other agents. The coordination module is comprised of several modular coordination mechanisms, subsets of which may be applied during coordination depending on the degree of coordination desired. For the formal details, see [7] and for additional details consult [6, 7].

#### 4. INTEGRATING MQ REASONING WITH DETAILED AGENT CONTROL

While the research with DTC and TÆMS provides some of the intellectual foundation for the *MQ* framework, each has unique strengths and each addresses a different class of issues. At the *MQ* level we have abstract representation of tasks and a unified utility-based view of tasks and possibly task/commitments pairs. At the DTC/TÆMS level, we have representation and reasoning about the details of domain actions, including task interactions and complex structurings of tasks to achieve a goal. The *MQ* level lacks detailed representations such as task interactions or grouping relations on the tasks whereas the DTC/TÆMS level lacks an organizational level view and lacks a unified reasoning perspective. The proper integration of these technologies can give agents the benefits of both – at one level reasoning about organizational concerns and at another level handling detailed *feasibility analysis* and *implementation* of the objectives.

In this section we present an integration path in which the TÆMS task structures of the lower level are abstracted and mapped into *MQ* task attributes. The integration of these technologies in this fashion raises an interesting issue. Currently, the *MQ* level does not represent or reason about detailed interactions between tasks. This type of control problem solving occurs at the detailed level via DTC and GPGP. Integration of the *MQ* level with the lower-level enables agents reasoning with the *MQ* framework to dynamically coordinate activities with other agents – to reason about interactions between detailed problem solving activities and to form commitments to sequence detail level activities over the interactions. This differs from the organizational temporal constraints that may be imposed directly at the *MQ* level or the request-with-service-time model used in the examples of [25, 27]. The problem with this integration is that when interactions span agents at the TÆMS level, it decreases the ability of control at the *MQ* level to produce predictable results (by definition, activities at other agents are difficult for the local agent to guarantee). In such cases, a two way interface in which the *MQ* level selects a high-level course of action which is refined by coordination at the lower-level may be appropriate. Note that building a coordination process at the *MQ* level does not resolve the issue because this class of interactions occurs at the more detailed level.

The integration path presented here assumes detailed coordination occurs at the lower level and that the *MQ* level is responsible for high-level task selection. These *MQ* level tasks correspond to TÆMS level task groups (conceptually each group is a graph) which are then scheduled via DTC and coordinated by GPGP in the

multi-agent context. The overall approach to integration is developed incrementally in the following sections and we first explore a restricted scenario in which the TÆMS level contains no interactions that span agents.

#### 4.1 Restricted Scope Integration through Bottom-up Characterization

To integrate the TÆMS and *MQ* technologies, the main issues that must be addressed are *model translation* issues, e.g., what an attribute at one level corresponds to at another level, and *control integration*, e.g., the implications of decisions made at one level to the other level. Some issues fall into both categories and in this section we do not attempt to divide the issues into one set or the other – the categorizations are for pedagogical use only.

When integrating the models, there are several different mappings that must take place. The *MQPS* and *MQCS* sets at the *MQ* level must be mapped into attributes at the TÆMS level (this includes determination of *which MQs* to map in each case as well as the *quantities* involved). Options include the TÆMS quality, cost, and duration distributions (and the uncertainty of these), and the TÆMS level resources. Another issue is the goal criteria used by the scheduler; the scheduler produces schedules to meet the criteria and thus the criteria setting defines the types or classes of schedules produced by DTC, i.e., regardless of the mapping to q,d,c (quality, duration, cost), obtaining desired results may require a particular agent objective function for DTC. Utility functions and preferences at the *MQ* level may also be mapped to TÆMS, though their role at the TÆMS level is unclear. At this time, preferences are used to select which tasks to perform at the *MQ* level but do not serve a direct role in control at the detailed level.

The mechanics of mapping of these attributes depends on the “direction” of the mapping. Note that in all cases, control at the *MQ* level addresses a more abstract class of issues and is designed for higher-level concerns. Accordingly, it is our view that the *MQ* framework should be responsible for the high-level selection of tasks for the agent, where *MQ* tasks correspond to a TÆMS task group that contains the details of task performance. There are two conceptual directions for mapping and integrating the two frameworks:

1. Mapping can flow in a *top down* fashion from the *MQ* level to the DTC level where tasks at the *MQ* level are associated with TÆMS task structures at the lower level. When using this approach, however, it is unclear how to characterize the TÆMS task structures at the *MQ* level and, more importantly, it is unclear what to do about the uncertainty involved. For any given TÆMS task structure, it is difficult to ascertain pre-analysis (by DTC) exactly what types of schedules are possible. The implication of this is that if the *MQ* level identifies a task,  $T_1$ , that should be performed using an associated TÆMS task group,  $TG_{T_1}$ , the *MQ* level cannot arbitrarily set deadlines or other requirements on  $TG_{T_1}$  without there being some probability that  $TG_{T_1}$  cannot be performed to meet the requirements. The solution to this problem is to statistically characterize the TÆMS task structures and to tie *MQ* tasks directly to candidate schedules for the TÆMS tasks, which brings us to the next mapping approach.
2. Mapping can flow in a *bottom up* fashion from the TÆMS level to the *MQ* level. This is the approach that we have selected. In this model, the different classes of schedules that can be produced by a TÆMS task structure are characterized and translated into alternatives of a single *MQ* task at the *MQ* level. This removes the unpredictability issue above because the statistical characterizations used at the *MQ* level are accurate reflections of what is possible at the TÆMS level.

To explore the *bottom up* mapping approach, let us start with a small algorithm and then expand discussion to include the full set of details that must be addressed. In this case, the objective is to create a duration profile for a single TÆMS task structure that can be used as the duration values of an associated *MQ* task.

- Let  $cb_i$  denote a scheduler criteria bundled as described in Section 3 and in [23] consisting of settings for the raw goodness quality slider,  $rgq$ , and the raw goodness sliders for duration and cost,  $rgd$ ,  $rgc$ . Let  $cb_i$  also contain a setting for the meta sliders (that define how to relate the other slider banks, e.g., raw goodness with certainty). In



```

Alternative: 0
  Duration: 600
  MQPS: {}
  MQCS: {}
  Criteria Bundle: rg_q = 100, rg_d = 0

Alternative: 1
  Duration: 520
  MQPS: {}
  MQCS: {}
  Criteria Bundle: rg_q = 80, rg_d = 20

Alternative: 2
  Duration: 480
  MQPS: {}
  MQCS: {}
  Criteria Bundle: rg_q = 60, rg_d = 40

Alternative: 3
  Duration: 440
  MQPS: {}
  MQCS: {}
  Criteria Bundle: rg_q = 40, rg_d = 60

Alternative: 4
  Duration: 420
  MQPS: {}
  MQCS: {}
  Criteria Bundle: rg_q = 20, rg_d = 80

Alternative: 5
  Duration: 324
  MQPS: {}
  MQCS: {}
  Criteria Bundle: rg_q = 0, rg_d = 100

```

**Figure 5: Skeletal  $MQ$  Alternatives for  $TG$**

rectly to  $TG$ . For example, if  $MQ_{TG}$  has a deadline of 20 and an earliest start time of 14,  $TG$ 's deadline may also be set to 20 and its earliest start time to 14. Design-to-Criteria will then construct schedules for execution so that they start no earlier than 14 and finish no later than 20 (in a statistical sense, as discussed in [24]). As the duration associated with the scheduled alternative of  $MQ_{TG}$  originates from  $TG$ , it is also assured that  $TG$  can be scheduled within the start time / deadline window specified by  $MQ_{TG}$ 's constraints.

The mapping approach thus far is incomplete. We must also map quality and possibly cost to different attributes at the  $MQ$  level to enable reasoning at that level to consider the different trade-offs of each alternative generated by the algorithm. This is a complex issue as it entails defining a mapping that determines which  $MQ$ s correspond to quality, which to cost, how these are related, and the quantity of each that should be associated with the quality and cost characterizations of the TÆMS actions. To illustrate the issue, consider a mapping of the DishWasher agent's TÆMS attributes to the  $MQ$  level. Note that each row in Table 1 includes expected quality as well as expected duration. Let the DishWasher agent be embedded in an organized environment and a member of the *cleaning agents* organization. Let the organization have an objective to produce ten units of cleaning in a given cycle and let the DishWasher agent's tasks pertain only to washing dishes. The agent thus produces two  $MQ$ s,  $MQ_{dishes\_washed}$  and  $MQ_{cleaning}$  where  $MQ_{cleaning}$  is the only  $MQ$  of interest to the organization. Let the importance or weight given to the dish washing task be such that each cycle of washing dishes produces between 1 and 2 units of  $MQ_{cleaning}$  where the range is correlated with the quality of  $TG$  as follows:  $0 \leq quality \leq 120$  produces 1 unit of  $MQ_{cleaning}$ ,  $120 < quality \leq 140$  produces 1.5 units of  $MQ_{cleaning}$ ,  $140 < quality \leq 160$  produces 1.75 units of  $MQ_{cleaning}$ , and  $quality > 160$  produces 2 units of  $MQ_{cleaning}$ . Let there be a 1:1 mapping from TÆMS quality to  $MQ_{dishes\_washed}$ . Using this mapping,  $MQ_{TG}$ 's six alternatives are characterized as shown in Figure 6.

With the mapping of quality to the  $MQ$  level  $MQPS$ , the agent can reason about the utility/duration trade-offs of the different alternatives and select accordingly. In some cases, it may not be

```

Alternative: 0
  Duration: 600
  MQPS: {(mq_cleaning 2.0), (mq_dishes_washed 165)}
  MQCS: {}
  Criteria Bundle: rg_q = 100, rg_d = 0

Alternative: 1
  Duration: 520
  MQPS: {(mq_cleaning 2.0), (mq_dishes_washed 165)}
  MQCS: {}
  Criteria Bundle: rg_q = 80, rg_d = 20

Alternative: 2
  Duration: 480
  MQPS: {(mq_cleaning 1.75), (mq_dishes_washed 155)}
  MQCS: {}
  Criteria Bundle: rg_q = 60, rg_d = 40

Alternative: 3
  Duration: 440
  MQPS: {(mq_cleaning 1.5), (mq_dishes_washed 135)}
  MQCS: {}
  Criteria Bundle: rg_q = 40, rg_d = 60

Alternative: 4
  Duration: 420
  MQPS: {(mq_cleaning 1.0), (mq_dishes_washed 120)}
  MQCS: {}
  Criteria Bundle: rg_q = 20, rg_d = 80

Alternative: 5
  Duration: 324
  MQPS: {(mq_cleaning 0.0), (mq_dishes_washed 0)}
  MQCS: {}
  Criteria Bundle: rg_q = 0, rg_d = 100

```

**Figure 6:  $MQ$  Alternatives for  $TG$**

worth the extra time required to produce two rather than one unit of  $MQ_{cleaning}$  and in other cases the longer duration alternatives may be the only way for the agent to meet its organizational cleaning goal. Cost and resource consumption are not addressed in this example. In the DishWasher task group, tasks have both associated costs and they consume resources such as hotwater – all of which can be mapped to  $MQCS$  and reasoned about along with the  $MQPS$  at the  $MQ$  level.

## 4.2 The Need for Approximation

Earlier we mentioned the requirement that tasks in  $TG$  not have interactions with other tasks not belonging to  $TG$ . The requirement is to ensure predictability. If all of the interactions in  $TG$  are local to  $TG$ , then the characterizations used to construct  $MQ_{TG}$  in the previous section are accurate. If there are interactions between  $TG$  and task groups belonging to other agents, the characterizations may not be accurate as there is no way, in the general case, to “pre-coordinate” over the interactions that occur at the TÆMS level. The reason for this is twofold. The first problem with such interactions is that when the classification of the TÆMS task structures is created, the agent does not know when the corresponding  $MQ$  task will be selected and scheduled, if ever. Thus, even if the agents with which the local agent interacts were willing to form commitments during classification, the local agent would be unable to indicate when the interacting activities would be executed. The implication of this is that the other agents would have to issue a blanket guarantee or willingness to cooperate and this guarantee would have no temporal scope. This brings us to the second reason that interactions pose a problem. The only way to obtain predictable classifications of TÆMS task structures involving inter-agent interactions is to require blanket guarantees from the other agents and it is highly unlikely that they can actually give such guarantees even if they wish to. The problem is again that the guarantees are not fixed temporally. The other agents have no way of knowing what other activities in which they will be engaged at that time. Even if they were able to give a firm commitment to cooperate when asked, they could only give the commitment to one agent over one interaction. Because the interaction has no temporal scope, if two agents wanted hard guar-

antees from the same third party, it would be unable to determine if the two guarantees would be “called in” at the same time and thus would be unable to give such assurances.

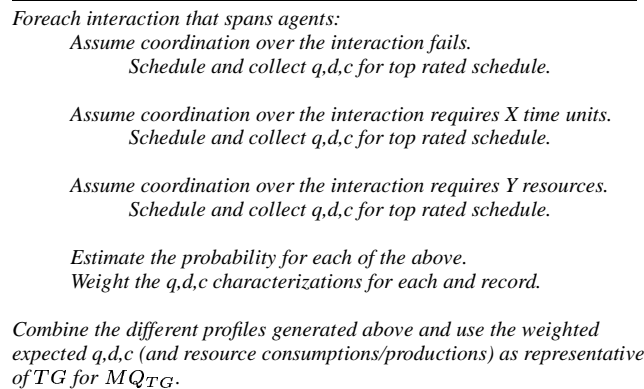
The bottom line is that when interactions span agents, the classification process is, by definition approximate or an estimate of what may be possible at run time. A two-way interface between the  $MQ$  level and the TÆMS level is thus required for effective control. The control is as follows:

1. **Classify** TÆMS task structures are classified in an approximate fashion (see below).
2. **MQ Schedule** Control reasoning at the  $MQ$  level selects a set of candidate tasks for the agent to perform.
3. **Backward Mapping** The  $MQ$  tasks are mapped into TÆMS by selecting the associated task group and coupling it with the criteria bundle created in the classification step. Additional changes to the TÆMS attributes may be necessary to reflect the importance of particular  $MQ$ s.
4. **Lower-Level Analysis** DTC and GPGP attempt to obtain the desired result given the criteria and the task structure.
5. **Feedback** Based on the output of the DTC/GPGP analysis, if certain tasks are unachievable or cannot be performed in an acceptable fashion, the characteristics of the associated  $MQ$  alternative are changed.
6. **MQ Schedule** The  $MQ$  reasoning process is reinvoked with the new performance estimates. If it is worthwhile, possibly the same task group will be selected with its revised estimates. If not, some other task(s) may be selected.
7. **Repeat** Steps 3-6 repeat.

In terms of approximately characterizing  $TGs$ , there are many options. One option is to examine prior interactions between the agents and characterize the task group on that basis. Another option is for all involved agents to participate in the characterization process using a set of assumptions, e.g., the other agents will be lightly loaded, the other agents will be highly constrained, etc. Still another option is to assume that the other agents will be able to handle the interactions in a timely fashion. An algorithm using this latter local-assumption approach could also be used to determine the possible implications of not being able to handle a given interaction; a sketch is shown in Figure 7. There are many different estimation possibilities – the discussion here is intended to identify the issue and outline possible solution paths.

### 4.3 Efficiency Issues

Even when the requirements of independence and floating temporal scope are met, there is an inefficiency hidden in the view presented thus far. Let TÆMS tasks  $T_1$  and  $T_2$  be the only members of  $TG$  and let both be required for all solution paths within  $TG$ . Let all actions in  $TG$  require more than 5 units of time to execute. Let  $T_1$  enable  $T_2$  such that the enablement (Section 3) requires 5 time



**Figure 7: Overview of Algorithm to Estimate Implications of Interactions Spanning Agents by Testing Different Assumptions**

units to propagate. Though there is an interaction, it is contained within  $TG$  and thus the characterizations of  $TG$  are accurate. Let  $MQ_{TG}$  correspond to the  $MQ$  task associated with  $TG$  and let  $MQ_{TG_{other}}$  correspond with the  $MQ$  task associated with some other task group,  $TG_{other}$ . Let  $TG_{other}$  consist entirely of actions that require 5 time units to execute and let  $TG_{other}$  adhere to the independence and floating temporal requirements, i.e., it too has accurate characterizations.

Consider what happens if the agent schedules  $MQ_{TG}$  followed by  $MQ_{TG_{other}}$  at the  $MQ$  level. If the agent then schedules  $TG$  independently from  $TG_{other}$  at the DTC level, the 5 time units spent waiting for results to flow from  $T_1$  to  $T_2$  (in  $TG$ ) are wasted. However, if  $TG$  and  $TG_{other}$  are scheduled together at the DTC level, one of the actions from  $TG_{other}$  can be inserted into the delay period in  $TG$  and the total duration of the two is decreased by the 5 (now utilized) time units.

This issue also comes into play when interactions span agents, though it may take a different form. Consider a set of task groups over which there are  $n$  uniformly distributed interactions between agent X and agent Y. Agent Y may regard contracting for  $n$  small individual tasks differently than a single larger contract that has  $n$  different components. It may be possible to address this issue by simply classifying the different task groups independently and then merging them when being scheduled by DTC for actual execution. This would possibly create a larger scheduling problem for DTC (and/or coordination problem for GPGP), but, it may also improve the efficiency of agent problem solving. The caveat is that if the performance improvements are significant, the agent may have selected a different course of action at the  $MQ$  level (to utilize the available time or resources that were hidden by the independent view). At this time, we hesitate to draw any conclusions and intend primarily to identify the issue.

## 5. RELATED WORK

There is a large body of related work for each of the technologies mentioned in this paper, e.g., TÆMS, DTC, GPGP, or the  $MQ$  framework. Some references are made in this paper though interested parties should consult papers dedicated to these technologies for a more thorough discussion.

With respect to the integration of high and low level control, this particular topic is not new to AI or to computer science as a whole. In the AI realm, the idea of combining reactive control with a more deliberative planning process is a very familiar model to most ([12, 1, 10] to cite a few examples). In this model reactive control has a short temporal scope over which decisions are made and the higher, more deliberative, level operates over a longer temporal scope. For instance, the lower level might respond to tactical issues or immediate threats whereas the upper level might do strategic planning.

This familiar model is related to the research explored in this paper, though, the details and specific issues are different. Both the TÆMS/ $MQ$  pairing and the classic reactive/deliberative pairing separate issues to different levels and generally, in complex systems, there is some interplay between the upper level and the lower level. For instance, in the reactive/deliberative world, the deliberative level might plot a course for an underwater AUV that would run the AUV aground if it were not for the reactive level recognizing an uncharted obstacle and responding accordingly. There are two key differences between the TÆMS/ $MQ$  pairing and the other work in this area. The first is that the TÆMS/ $MQ$  pairing involves the coupling of technologies that do not necessarily examine issues over different temporal scopes or issues of different levels of complexity. In fact, the  $MQ$  and TÆMS levels may actually operate over the same temporal scope and perform decision processes of equal complexity. The distinction between the levels is that the focus of each level is different. At the lower level, we have complex control problem solving over the domain process. At the upper level, we have complex control problem solving over the agent’s organizational situation. Another important difference between the TÆMS/ $MQ$  pairing and the reactive/deliberative model is that the TÆMS/ $MQ$  pairing must

address the complexity of interactions that span agents and interactions that do so only at the TÆMS level. This introduces a new kind of complexity because interactions, unlike avoiding an obstacle along a given path, can make the course of action chosen at a higher level completely infeasible. A harder issue still is the gray zone where interactions make the higher level reasoning more unpredictable, cause degradation in efficiency, or affect the valuation process at the higher level so that what appeared desirable at that level is less so when the details are examined. This may occur during the performance of the selected course of action or *a priori*. Conceptually this is more akin to coupling two interdependent planners than coupling a planner and a rective component.

Another body of work that is particularly relevant to this integration is Durfee's control via behavior spaces [9]. In this research, agents coordinate by building different abstractions of the same underlying space – where the abstractions are constructed for a specific coordination purpose. Durfee's approach to integration is, in essence, to use the same technique and same underlying representation for all control/coordination reasoning. This is particularly clean and elegant. Our technology differs in that the different levels focus on entirely different classes of issues and use different reasoning processes. The intersection point in our problem space is that of the task selection process and the placing of temporal constraints. The analog of Durfee's work in our work would be to abstract directly from TÆMS to build constructs that represent exactly the desired feature of the TÆMS task structure. This is not possible with the *MQ* model because TÆMS and *MQs* have very different focuses (*MQs* are not an abstraction of TÆMS). Because of these differences, the integration through bottom-up abstraction process presented here still leaves a set of unresolved control problems, such as coordination across agents, that must be addressed by iterating over the different technologies or letting the technologies message one another as needed to converge on a feasible and acceptable solution space.

## 6. CONCLUSION

We have presented an approach to integrating detailed control via TÆMS, DTC, and GPGP with organizational level control via the *MQ* framework. The integration of these technologies gives agents the ability to reason about the situational complexities that arise when agents are deployed in large-scale, open, environments plus the ability to perform detailed analysis of domain problem solving activities and to coordinate activities across agents. The paper identifies several areas of future work – one particularly difficult issue is how to automate the mapping of quality/cost at the TÆMS level into the *MQ* production/consumption sets and preference curves for use at the upper level. In the example given, the mapping was specified manually. A similar issue arises when using the *MQ* framework to achieve organizational level control – we currently lack a formal procedure for constructing the models and mapping organizational objectives onto *MQs* and utility functions or preference curves. Full automation of these mappings would require deep understanding of the domains at hand and is probably unrealistic at this time. However, it would be desirable to give the agent system designer higher-level tools for these mapping problems.

The issue of interactions spanning agents, and the implications of this to the integration approach described here, is not fully explored in this paper. It is a deep issue in terms of control because interactions represent an unknown quantity and at some point during problem solving, assumptions must be made and problem solving must move forward on that basis. One issue in particular not explored here is that when interactions span agents, the sequencing of *MQ* tasks may impose arbitrary constraints on coordination at the lower levels that lead to performance degradation. This is a case where a high-level assumption is made and while intractability at the lower levels may refine the upper level, with the two way interface defined here, inefficiencies are possible. In general, we regard a certain amount of inefficiency as being acceptable given the intractability of the problem space.

## 7. REFERENCES

- [1] Ron C. Arkin. *Behavior-Based Robotics*. MIT Press, Cambridge, MA, 1998.
- [2] Mihai Barbuceanu. Agents that work in harmony by knowing and fulfilling their obligations. In *Proceedings of the 15th Nat. Conf. on AI*, pages 89–96, 1998.
- [3] Craig Boutlier, Yoav Shoham, and Michael P. Wellman. Economic Principles of Multi-Agent Systems. *Artificial Intelligence*, 1-2(1-6), 1997.
- [4] M.E. Bratman. *Intention Plans and Practical Reason*. Harvard University Press, Cambridge, MA, 1987.
- [5] Cristiano Castelfranchi. Commitments: From individual intentions to groups and organizations. In *Proceedings of the First Intl. Conf. on Multi-Agent Systems (ICMAS95)*, pages 41–48, 1995.
- [6] Keith Decker and Jinjiang Li. Coordinated hospital patient scheduling. In *Proceedings of the 3rd Intl. Conf. on Multi-Agent Systems (ICMAS98)*, pages 104–111, 1998.
- [7] Keith S. Decker. *Environment Centered Analysis and Design of Coordination Mechanisms*. PhD thesis, University of Massachusetts, 1995.
- [8] Edmund H. Durfee, Victor R. Lesser, and Daniel D. Corkill. Coherent cooperation among communicating problem solvers. *IEEE Transactions on Computers*, 36(11):1275–1291, November 1987.
- [9] Edmund H. Durfee and Thomas A. Montgomery. Coordination as distributed search in a hierarchical behavior space. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6):1363–1378, 1991.
- [10] M.P. Georgeff and A.L. Lansky. Reactive reasoning and planning. In *Proceedings of AAAI-87*, pages 677–682, 1987.
- [11] Barbara J. Grosz and Sarit Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86:269–357, 1996.
- [12] Barbara Hayes-Roth. An Architecture for Adaptive Intelligent Systems. *Artificial Intelligence*, 72(12):329–365, 1995.
- [13] Bryan Horling, Regis Vincent, Roger Mailler, Jiaying Shen, Raphen Becker, Kyle Rawlins, and Victor Lesser. Distributed sensor network for real-time tracking. In *Proceedings of Autonomous Agent 2001*, 2001.
- [14] N.R. Jennings, J.M. Corera, L. Laresgoiti, et al Using ARCHON to develop real-world dai applications for electricity transportation management and particle accelerator control. *IEEE Expert*, 1995.
- [15] V. Lesser, M. Atighetchi, B. Horling, B. Benyo, A. Raja, R. Vincent, T. Wagner, P. Xuan, and S. Zhang. A Multi-Agent System for Intelligent Environment Control. In *Proceedings of the 3rd Intl. Conf. on Autonomous Agents (Agents99)*, 1999.
- [16] Victor Lesser, Bryan Horling, and et al. The TÆMS whitepaper / evolving specification. <http://mas.cs.umass.edu/research/taems/white>.
- [17] V. Lesser, B. Horling, F. Klassner, A. Raja, T. Wagner, and S. Zhang. BIG: An agent for resource-bounded information gathering and decision making. *Artificial Intelligence*, 118(1-2):197–244, May 2000. Elsevier.
- [18] H. Van Dyke Parunak, Allen Ward, and John Sauter. A Systematic Market Approach to Distributed Constraint Problems. In *Proceedings of the 3rd Intl. Conf. on Multi-Agent Systems (ICMAS98)*, 1998.
- [19] Anita Raja, Victor Lesser, and Thomas Wagner. Toward Robust Agent Control in Open Environments. In *Proceedings of the Fourth Intl. Conf. on Autonomous Agents (Agents2000)*, 2000.
- [20] A.S. Rao and M.P. Georgeff. Modelling rational agents within a BDI-architecture. In J. Allen, R. Fikes, and E. Sandewall, editors, *Proceedings of the 3rd Intl. Conf. on Principles of Knowledge Representation and Reasoning*, pages 473–484. Morgan Kaufmann, 1991.
- [21] Sandip Sen and Anish Biswas. Effects of misconception on reciprocal agents. In *Proceedings of the Second Intl. Conf. on Autonomous Agents (Agents98)*, pages 430–435, 1998.
- [22] Milind Tambe. Agent Architectures for Flexible, Practical Teamwork. In *Proceedings of the 14th Nat. Conf. on AI*, pages 22–28, July 1997.
- [23] Thomas Wagner, Alan Garvey, and Victor Lesser. Complex Goal Criteria and Its Application in Design-to-Criteria Scheduling. In *Proceedings of the 14th Nat. Conf. on AI*, pages 294–301, July 1997.
- [24] Thomas Wagner, Alan Garvey, and Victor Lesser. Criteria-Directed Heuristic Task Scheduling. *Intl. Journal of Approximate Reasoning, Special Issue on Scheduling*, 19(1-2):91–118, 1998. Also available as UMASS CS TR-97-59.
- [25] Thomas Wagner and Victor Lesser. Relating quantified motivations for organizationally situated agents. In N.R. Jennings and Y. Lespérance, editors, *Intelligent Agents VI (Proceedings of ATAL-99)*, Lecture Notes in AI. Springer-Verlag, Berlin, 2000.
- [26] Thomas Wagner and Victor Lesser. Design-to-Criteria Scheduling: Real-Time Agent Control. In *Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent Systems*. LNCS. Springer-Verlag, 2001.
- [27] Thomas Wagner and Victor Lesser. Evolving real-time local agent control for large-scale mas. In J.J. Meyer and M. Tambe, editors, *Intelligent Agents VIII (Proceedings of ATAL-01)*, Lecture Notes in AI. Springer-Verlag, Berlin, 2002.
- [28] Shelley XQ Zhang, Victor Lesser, and Thomas Wagner. How Cooperative Should I Be? Under review, 2001.
- [29] Shelley XQ Zhang, Victor Lesser, and Thomas Wagner. A proposed approach to sophisticated negotiation. In *Proceedings of AAAI Fall Symposium on Negotiation Methods for Autonomous Cooperative Systems*, 2001.