At the Boundary of Workflow and AI

Karen L. Myers and **Pauline M. Berry**
Artificial Intelligence Center
SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025
myers@ai.sri.com, berry@ai.sri.com

Abstract

Many domains of interest to the workflow community are characterized by ever-changing requirements and unpredictable environments. Workflow systems must increase in sophistication to provide the reactivity and flexibility necessary for process management under such dynamic conditions. This paper describes how techniques from the AI community, specifically reactive control, planning, and scheduling, could be leveraged to develop powerful, next-generation adaptive workflow engines that provide many of the these advanced process management capabilities. Although motivated by somewhat different concerns and grounded in different perspectives, there is much overlap between the objectives and requirements of these two communities of workflow management and Artificial Intelligence. Two systems under development by the authors which embrace this synergy are described.

Introduction

Large enterprises employ a broad range of processes to conduct their day-to-day operations. Many of these processes, while well-defined, are either implicit in the operation of the organization or only informally recorded. This lack of explicit representation makes it difficult to track the progress of active processes, to modify ongoing processes in response to situational changes, or to adopt newly defined processes. Assurance that processes are executed successfully and efficiently is difficult or impossible to obtain.

The widespread use of computers and increasing availability of online information has sparked interest in employing automation to improve process management. The field of workflow management (WFM) has emerged as an outgrowth of this interest in recent years. The workflow community advocates the use of explicit models and representations of processes, along with automated tools to support the activation and ongoing management of workflow processes.

Structured management of processes can provide benefits in several ways. Articulation of explicit processes can help to standardize operation, thus reducing the likelihood of introduced errors. Explicitly represented processes can be tuned to improve efficiency and effectiveness. Automated tools for process management hold promise for deeper insight into current and planned operations, through the provision of timely updates on progress and status. Such enriched understanding of operating processes will lead to better-informed and more principled decision-making by users. Automation can also provide adaptive capabilities that enable agility of operation and more effective use of resources. Such characteristics are critical for operation in dynamic environments where requirements and conditions can change rapidly and unpredictably. The explosive growth in interest in WFM, within both the commercial and military sectors, reflects the degree to which organizations believe that structured process management through explicit models and automated tools will improve operations considerably.

Workflow constitutes a relatively new technical field, having been established only within the past 5 years. However, the Artificial Intelligence (AI) community has been involved with related research on process management for several decades. While workflow has concentrated on business and manufacturing processes, the AI community has been motivated primarily by domains that involve active control of computational entities and physical devices (e.g., robots, antennas, satellites, computer networks, software agents). Despite their differing emphases, there is much overlap between the objectives, requirements, and approaches for process management within these two communities.

This paper describes how techniques from the AI community could be leveraged to produce technologies that provide many of the advanced process management capabilities envisioned by the workflow community. The paper focuses on the role that AI technologies can play in the construction of workflow engines that manage complex processes, while remaining robust, reactive, and adaptive in the face of environmental changes.

Three subfields of AI are directly relevant to these objectives for WFM, and as such are the focus of the discussion within this paper. *Reactive control* has developed techniques for intelligent process management, with particular emphasis on requirements of adaptivity for dynamic and unpredictable environments. *Scheduling* provides methods for resource and task allocation, spanning both initial generation and incremental modifications to support dynamic operating conditions. *Planning* provides capabilities for synthesizing new processes and repairing previously defined processes that are no longer suitable for a given situation. This document provides descriptions of the current state of the art for each of these subfields, along with discussions of their potential contributions to workflow technology. Shortfalls relative to workflow requirements are also discussed.

Other disciplines within AI similarly have much to contribute to the overall vision of the workflow community. The field of *knowledge acquisition* has developed methodologies and tools to support user definition of process models, along with preliminary techniques for critiquing defined processes to identify both syntactic and semantic inconsistencies. Related work has explored methods for explaining both template content and problem-solving behavior, but is currently limited in scope. *Distributed AI* has produced numerous protocols for the coordination of multiple players to achieve distributed tasks. Work on *agent architectures* provides frameworks that can support distributed workflow systems. This paper does not discuss these topics at length, although they are clearly relevant to the development of sophisticated workflow frameworks.

Overview of Workflow Management

Workflow is a fast-evolving area that has been influenced by several fields, including computer-supported cooperative work (CSCW), business management, modeling, simulation, and to a lesser extent distributed artificial intelligence (DAI). It has evolved primarily from the desire to understand, organize, and (often) automate the processes upon which a business is based. These roots are reflected in the following definition of workflow management from the Workflow Management Coalition (WfMC), an international organization focused on the advancement of workflow management technology and its use in industry.

> The automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules (Hollingsworth 1994).

These business roots notwithstanding, the requirements and capabilities inherent to workflow apply broadly to a wide variety of process coordination and control problems. Within this paper, we use the term workflow to refer to this more general notion of process management.

The role of a *workflow management system* is to provide the services required for automating workflow processes. Figure 1 presents an overview of a generic WFM system, adapted from the WfMC's Reference Model (Hollingsworth 1994).

At the heart of a WFM system lies a library of templates that encode explicit *process models* for the relevant problem domain. Templates can be represented as explicit sets of tasks or activities to be undertaken, or more abstractly as collections of constraints on allowed activity. Activities can encompass a broad array of operations, including transmission of data, tasking of software agents, or communication with human operators. As reflected in Figure 1, process modeling and representation are typically limited to build time within current WFM technology. However, process creation and adaptation will inevitably migrate from build time to runtime as applications demand flexible adaptation to environment dynamics.

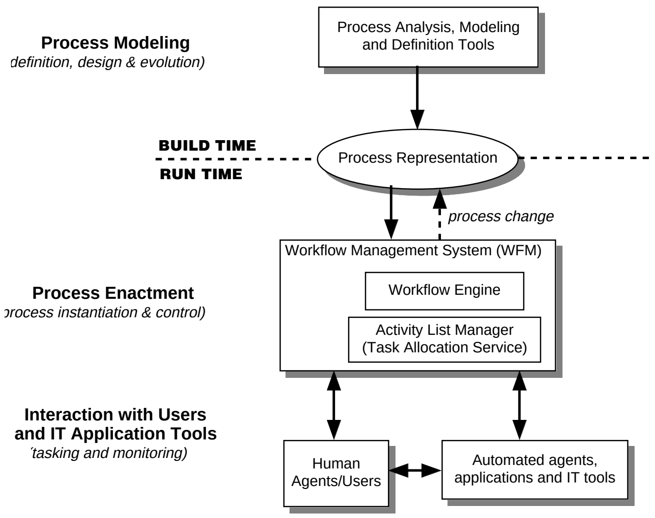
Process Modeling
(definition, design & evolution)
Process Analysis, Modeling and Definition Tools
BUILD TIME
RUN TIME
Process Representation
process change
Workflow Management System (WFM)
Workflow Engine
Activity List Manager (Task Allocation Service)
Process Enactment
(process instantiation & control)
Interaction with Users and IT Application Tools
(tasking and monitoring)
Human Agents/Users
Automated agents, applications and IT tools

Figure 1: Architecture for Current WFM Systems

Templates are selected for *activation* based on current tasking and environmental conditions. Selected templates can be tailored to a range of situations through appropriate instantiation of template variables. Typically, activation results in the addition of the constituent activities of the instantiated template to an *activity list*, which contains information about all current and pending tasks. As part of activation, resource allocation is performed for new activities based on current and projected resource availability. These activities are then scheduled in accordance with temporal sequencing constraints specified within the process template

The term *enactment* is generally used to refer to the execution and ongoing management of activated processes. For many domains, process execution will occur within highly unpredictable and dynamic operating environments. Thus, enacted processes must evolve and adapt over time in response to a number of factors, including changes in the environment, the addition of new tasks, and partial execution results. Monitoring plays a critical role during enactment, to detect key events that may necessitate adaptations to current processes, or enactment of new or different processes. Similarly, process templates should evolve over time, to reflect improved models of the domain obtained by analyzing information gathered from previous enactment episodes.

Process Modeling

Process modeling as a discipline has been around for some time. To date, three main approaches have been defined (Cichocki *et al.* 1998; Lawrance 1997):

- *Communication-based* modeling (based on (Winograd & Flores 1987)) defines processes in terms of communication acts between customers and performers.
- *Artifact-based* modeling focuses on the objects (e.g., data or information) that are created, modified, and used within a process, along with their paths through a series of activities.

- *Activity-based* modeling focuses on the activities that are to be performed within a process, along with dependencies and constraints among them.

These three approaches are not exclusionary: process models can integrate information flow requirements, activity decomposition, and communication constraints. For example, an organizational process might be represented using an activity model closely linked to an information flow model, which combines artifact- and communication-related knowledge, and a capabilities model relating activities to processing entities. The WfMC focuses on the activity-based paradigm for process modeling.

The development and maintenance of process definitions is an integral part of workflow management. As techniques for process creation and adaptation migrate into runtime, it will be possible to automate the maintenance of processes across time, organizational, and business changes.

Requirements for Workflow Engines

A workflow engine provides the runtime environment for activating, managing, and executing workflow processes. Different models for workflow engines can be employed. The WfMC focuses on a paradigm in which the engine instantiates a workflow specification, decomposes it into taskable activities, and then allocates activities to processing entities for execution. This approach distinguishes between the *process definition*, which describes the processes which may be executed, and the *process instance*, which is the actual enactment (execution) of the process. This paradigm is referred to as the *scheduler-based paradigm* (Cichocki *et al.* 1998).

Implementation of the scheduler-based approach to workflow can be described in terms of a state transition machine. Individual process or activity instances change state in response to workflow engine decisions or external events, such as the completion of an activity. Thus, to give a few example states, a process instance may be *initiated* once selected for enactment, *active* after at least one of its activities has been started, *suspended* (perhaps waiting for some event), or *completed*. Similarly, an activity may be *inactive*, *active*, *suspended*, or *completed*. It is the role of the workflow engine to manage this state transition machine, selecting processes to be instantiated, initiating activities by scheduling them to processing components, and controlling and monitoring the resulting state transitions. The workflow engine must also implement the rules that govern the transitions between tasks, updating the processes as tasks complete or fail, and taking appropriate actions in response.

Process Selection One key responsibility of the workflow engine is to manage the selection and instantiation of process templates. The engine will respond to some stimulus (i.e., a *triggering event*) by selecting a suitable process from the library of templates. Possible triggering events include arrival of a new user request, generation of a product by an already active process, or even the passage of time. The workflow engine manages the instantiation of the relevant process. There may be alternative applicable processes that must be compared to the triggering events and selected appropriately. In many existing WFM systems this task is trivial, as there is no or little choice among processes given the predefined stimulus for enactment. But there are domains where there may be many, or even no, directly applicable processes for a given stimulus, thus requiring process selection, adaptation, or even dynamic process creation.

Task Allocation Once a process is instantiated, the workflow manager forwards activities to an *activity list manager* to control the tasking of processing entities for the activities. An activity is assigned to a processing entity according to the capability and availability of the entity and temporal and sequencing constraints of the activity. This allocation task can be viewed as a scheduling problem. Thus, the workflow engine takes a centralized role in coordinating the operation of processing entities. The *activity list manager* can be implemented as a simple in-tray of work items or a complex set of agenda-based load balancing and interrelated work assignments with complex supervisory roles.

Scheduling techniques within workflow management systems to date employ straightforward enumerative or heuristic-based algorithms. As the complexity of WFM domains increases, more sophisticated approaches will be critical that provide robust reactive scheduling to accommodate processing entities that require some autonomy, can be tasked by other workflow managers, may break down, or have variable capabilities. For example, there has been some work on exploiting the dependencies between activities within the workflow task allocation problem (Attie *et al.* 1996).

Enactment Control, Execution Monitoring, and Failure Recovery The workflow engine must maintain all the knowledge and internal control data to identify the states of each of the individually instantiated activities, transition conditions, connections among processes (for example, parent/child relationships), and performance metrics.

There has been a significant amount of work on control within workflow enactment systems, based primarily on state transition mechanisms and transaction processing. However, current-generation WFM systems have limited abilities to recover from failures, with many simply assuming that operations will complete successfully. Workflow failures may result from a processing entity not completing a task correctly or on time, or by an unexpected event in the environment. WFM systems should provide a variety of recovery mechanisms, from the clean termination of a process to the adaptation of the process to the failure.

Work in this area to date has focused on the use of transactional methods to ensure data integrity, especially when concurrent activities must share data. For example, most WFM systems offer only basic locking mechanisms to protect data, although there has been some work on longer-lived transactions (Garcia-Molina *et al.* 1990; Reuter 1989; Mohan *et al.* 1995). These approaches focus on returning the state of execution to a previously known safe state. Two popular techniques are *checkpointing* and *roll-back.*

- *Checkpointing* relies on log files that store safe states; when problems occur, the system can be reset to the most recent safe state and processing restarted.

- *Roll-back* (Eder & Liebhart 1995) employs invertible actions: when a problem occurs, the effects of actions are undone by employing their inverses in reverse order.

These approaches, while suitable for certain forms of data-driven workflow, are inadequate for dynamic domains in which the world changes continuously and unpredictably, and in which actions may not be undoable. More flexible methods for failure recovery are required, including the ability to adapt processes at runtime in response to failures or changes in situation, and to accommodate open-ended activities.

Open Issues for Workflow Management

While workflow technology has seen an explosion of interest and advances in recent years, numerous technical challenges must be addressed in order to provide the kind of flexible workflow management systems required by complex and dynamic application domains. Here, we briefly discuss challenges for process representation and process management to which the fields of Reactive Control, Scheduling, and Planning can contribute. In ensuing sections, the potential contributions of these fields are discussed in more detail.

Process Representation Challenges Effective workflow management requires representations that makes the process logic explicit, thus allowing processes to be readily understood and adapted. Hierarchical models are generally desirable for modeling sophisticated processes due to their capacity to simplify complex tasks.

Basic constructs that should be recorded for a process include the purpose (e.g., what it accomplishes), expected effects, applicability conditions, resource requirements, scheduling constraints, participants, and subprocesses that may be invoked. Process representations should support the definition of metrics relating to the time, cost, or quality of performing a process, thus allowing comparisons and improvements to be made. Concepts such as authority and accountability are also essential.

The representations should support a rich set of control metaphors, including iteration, sequencing, concurrency, monitoring, testing, and suspension/resumption. Representations of uncertainty are critical, given the dynamics and unpredictability of the intended operating environments. In addition, constructs to support coordination of distributed multi-entity activity will be required. At a low level this includes support for transactional operations, synchronizing primitives and information exchange constructs; above that, there should be distributed control protocols, such as negotiation or self-organization strategies.

Process Management Challenges Workflow management will require enactment and management of processes within highly dynamic and uncertain operating environments. Thus, WFM systems should provide support for monitoring for critical events and responding appropriately when such events are detected. Processes should evolve over time in response to retasking, failures, and policy changes. Automatic load balancing is essential to ensure effectiveness and timeliness. Full tracking of process execution and auditing capabilities are required for understandability and to enable adaptation of processes based on experience.

Future WFM systems should support human interaction. Not all aspects of a process may be automated, or automatable, thus requiring that WFM systems work in partnership with humans. This includes flexible integration with humans at the control level as well as the ability to manage the tasking of human processing entities. Key issues include user visibility into enacted processes, controllability of WFM systems, flexible automatic load balancing for processing components, and support for conflict resolution.

AI Technologies for Building Workflow Engines

The disciplines of *reactive control*, *scheduling*, and *planning* from the AI community have developed technologies and models that go a long way toward meeting the requirements for effective workflow engines outlined in the previous section. Here, we briefly summarize the roles that these capabilities can fill within a workflow engine.

Reactive Control: Adaptive Process Management

Reactive control systems provide a strong foundation upon which to ground the process activation and management capabilities of a workflow engine. A reactive control system is a form of knowledge-based software controller that operates as an embedded system within dynamic environments. Reactive controllers perform actions to accomplish explicitly assigned tasks, while providing real-time response to unexpected events that result from factors that lie beyond the system's control. Such smooth integration of event- and goal-driven activities is an essential component of adaptive workflow management, which must combine taskability with responsiveness to environmental changes. Most relevant to the design of workflow engines are the *procedural* reactive controllers, which employ libraries of explicitly encoded processes to guide the activities of the system (e.g., (Firby 1994; Georgeff & Ingrand 1989; Myers 1996a; Musliner, Durfee, & Shin 1993; Howe & Cohen 1991)).

The development of reactive control technology has been motivated primarily by domains that involve control of computational processes and physical devices (e.g., robots, antennas, satellites, computer networks, software agents). However, many of the techniques and methods can be transferred readily to managing the sorts of processes that have been the focus of the workflow management community.

To operate effectively in highly dynamic environments, reactive control systems require the ability to respond appropriately when problems arise. Reactive control frameworks operate as embedded systems in dynamic environments, performing activities that can change the world in irrevocable ways. For this reason, recovery techniques grounded in *checkpoint* schemes, where coherent states are saved periodically to enable rollback to a consistent state in case of unrecoverable failures, are not viable. Instead, *forward* recovery methods are required that can identify actions that will transition from a failed state to some known, safe state.

Within the procedural reactive control community, such recovery mechanisms are currently *ad hoc* or domain-specific. For the most part, it is the process modeler's responsibility to ensure that the system either avoids problematic states or has procedures to recognize and recover from them. Tools for ensuring key properties (e.g., safety conditions, liveness, guaranteed response time) generally lack sophistication and are not commonly used. More principled recovery mechanisms have been explored for the repair of automatically generated plans, as discussed further below.

More progress has been made in *failure prevention* within reactive control systems. Forward search methods have been defined that determine actions to take from various states that will enable attainment of a specified goal while avoiding failure states (Drummond 1989; Godefroid & Kabanza 1991) (so-called *safety* rules). Early work ignored possible exogenous effects, while more recent work attempts to preempt possible failures that exogenous events could engender (Musliner, Durfee, & Shin 1995). This work presents a promising approach to avoiding failure through runtime lookahead analysis.

AI Scheduling: Adaptive Resource Allocation

AI scheduling provides a rich set of techniques for allocating tasks and resources within workflow management systems, with particular relevance to scheduler-based approaches to workflow enactment. AI scheduling technology can provide both initial task and resource allocation, as well as the ability to adapt and adjust allocations in response to changing requirements and resource availability.

Robust schedule construction is critical to dynamic workflow management because a valid schedule must be maintained while new activities are added and activity duration may be dependent on external influences. One successful AI approach to integrates the knowledge of domain and scheduling uncertainties into the construction of a schedule. For example, the work on fuzzy scheduling associates degrees of uncertainty with setup and duration times for processing (Kerr & Walker 1989; Turksen 1997). Other techniques include constructive methods that use predictive information to build schedules that are resistant to failure in the face of unexpected events (Berry 1992; Sadeh 1994).

The ability to reschedule during execution is an ongoing research problem but provides some solutions to the tasking of entities which do not always behave in the expected fashion and where new processes or unexpected event disrupt the workflow. Techniques include the following:

Constraint-directed algorithms (Smith 1994) Constraint analysis is used to prioritize outstanding problems in the current schedule, identify important modification goals, and estimate the possibilities for efficient and nondisruptive schedule modification. Possible modification actions include reordering contiguous sequences of operations, re-sequencing groups of operations on one resource, substituting resources, temporal right or left shift operations, and pairwise exchange of resource or temporal assignments. Intelligent schedule repair techniques of this type have been explored by Sycara and Miyashita (Sycara & Miyashita 1992).

Constrained iterative repair Recently, there has been a move toward *anytime* algorithms (Zweben *et al.* 1994; Drummond *et al.* 1993) in which a legal but possibly low-quality solution is generated quickly, then continuously updated to higher-quality solutions. Stochastic techniques (e.g., simulated annealing, genetic algorithms) can readily be used to iteratively improve a schedule by making small changes, but must be constrained to repair only the disrupted part of a schedule while maintaining stability.

An alternative approach to tasking is offered by combining AI scheduling techniques and advances in agent-based technologies. Most current WFM systems view scheduling as a centralized function. Forexample, scheduler-based WFM systems assume that the workflow engine has the responsibility for scheduling tasks to processing entities. Alternatively, scheduling can be performed in a distributed fashion with tasking decisions being undertaken by the processing entities themselves. The distributed AI community (DAI) has investigated a variety of strategies and protocols for tasking and negotiation within a distributed agent paradigm. An agent-based approach might view these entities as agents competing for tasks or cooperating to fulfill a set of tasks. Thus, as tasks are placed on the activity worklists, agents may competitively bid for work (Lesser & Corkill 1983) or may negotiate in a cooperative manner to distribute work (Zlotkin & Rosenschein 1989). There has been quite a lot of work on distributed or agent-based agenda ordering and task allocation (Lui & Sycara 1998; Gasser & Huhns 1989).

AI scheduling combines a rich representation of constraints and powerful constraint reasoning mechanisms with intelligent search techniques. In workflow domains, the tasking requirements will be volatile, and entities may have characteristics varying from well-defined automated processes to less predictable humans. AI scheduling provides techniques to address these forms of uncertainty. Work on reactivity, the use of predictive analysis, and algorithms for robust scheduling, are directly relevant. The ability to reason about constraints and manage their propagation and relaxation effectively will contribute to the human-computer interactions required in many workflow domains. In summary AI scheduling techniques can contribute to workflow in terms of offering increased reactivity, advanced constraint management techniques, increased ability to perform resource balancing and rich representation schema for constraints and domain heuristics (or policy).

AI Planning: Process Synthesis and Repair

Workflow engines can select from among defined processes to respond to new tasks and events based on current situational and problem-solving information. For certain tasks and events, however, it is necessary to consider the long-term ramifications of action choices (for example, to ensure that sufficient resources will be available to complete a task given other projected commitments). For such situations, automated planning techniques can be used to synthesize new processes from previously defined process templates that are guaranteed correct relative to current commitments and knowledge.

Methods from the AI planning community enable composition, adaptation, and synthesis of processes, thus providing the means to expand predefined process libraries to accommodate new situations and requirements. In addition, work on plan repair from this community provides techniques for modifying activated processes in response to execution-time failures and unexpected events.

Two automated plan generation methods lend themselves most naturally to the synthesis of new processes from libraries of previously defined processes.

- *Hierarchical task network (HTN) planning* (Erol, Hendler, & Nau 1994) synthesizes plans using libraries of processes (referred to as *task networks*) defined over multiple levels of abstraction. Planning consists of incrementally refining tasks at high levels of abstraction by applying more refined task networks, eventually bottoming out in a set of directly executable tasks. HTN planning is well-suited to workflow management, given the similarity between processes and task networks. HTN planning can be used to compose long-range processes from smaller units, using the power of lookahead analysis to ensure the viability of the constructed process.
- *Case-based planning* (Hammond 1989; Veloso 1992) generates new plans for a given situation and task by retrieving solutions for similar problems from a previously defined *case library*, and then adapting them to meet the requirements of the current situation. As such, case-based planning methods provide a way to build on experience with previously defined processes, providing adaptation to suit new conditions and requirements.

Plan repair methods provide the basis for principled recovery from problems during process execution. Most plan repair methods begin with an analysis of the *dependency structures* that underlie a plan to identify problems relative to the current state and execution results (Wilkins 1985; Kambhampati & Hendler 1992). As with plan synthesis, repair methods grounded in different methodologies suit different repair tasks.

Within generative planning systems, plan repair focuses on *replanning*, which involves generating new subplans to replace portions of the original plan for which dependency analysis has identified problems (Wilkins 1985; Kambhampati & Hendler 1992; Veloso, Pollack, & Cox 1998). Generative replanning methods emphasize properties of *correctness-preserving* (i.e., the modified plan is proven correct relative to the current state of knowledge) and *minimal-perturbation* (i.e., only portions that must be changed to ensure correctness are modified). Minimization of change is important to ensure the continuity of the plan, and because of the potentially high costs of redirecting execution entities. Case-based methods (Hammond 1990) apply libraries of predefined domain-independent adaptation methods to replace problematic portions of the plan. These approaches are generally correctness-preserving but generally don't guarantee minimal perturbation.

Ideally, a process management system should provide a spectrum of plan repair mechanisms ranging from the correct but costly minimal-perturbation, correctness-preserving methods to transformational approaches that employ domain-specific transformation rules (in the spirit of (Ambite & Knoblock 1997; Howe 1995)) that may trade correctness for efficiency.

Complementing the work on plan repair are efforts to develop plans that are less likely to fail in the first place. This work is in the early stages but will become increasingly important as plans are generated that are to be executed in dynamic, real-world environments. For example, methods for *contingency planning* (Pryor & Collins 1996) provide a start toward robustness by conditionalizing plans relative to enumerable sets of possible alternative situations (e.g., providing different subplans for good and bad weather conditions). As such, they accommodate certain levels of variability in uncertain aspects of the world state. Current methods are suitable when uncertainty is limited and well-circumscribed. For domains where uncertainty is pervasive and small sets of alternative situations are not enumerable, more sophisticated methods are needed to determine how many options to generate and for which combinations of unknown factors.

Adaptive Systems: From AI to Workflow

We are involved with two ongoing projects focused on adaptive process management. The first, the *Continuous Planning and Execution Framework* (CPEF) (Myers 1998), is developing a framework that supports the generation, adaptation, and execution of complex plans to attain assigned goals in highly dynamic and unpredictable environments. The second, *Smart Workflow for ISR Management* (SWIM), has a more conventional workflow flavor. It is our intent to leverage technologies being developed in CPEF to construct the adaptive workflow engine for SWIM.

CPEF

CPEF is a multiagent framework for performing and managing complex tasks in dynamic and uncertain environments. It provides *taskability* (i.e., the ability to formulate and execute plans to achieve assigned high-level tasks) and *reactivity* (i.e., the ability to adapt behavior based on changes in the operating environment). Tasks often involve long-term commitments that require look-ahead analysis; for this reason, generative planning technology is employed to compose new plans from libraries of operator templates.

In contrast to many integrated planning and execution systems, CPEF embraces the philosophy that plans are dynamic, open-ended artifacts that must evolve in response to an ever-changing environment. In particular, plans are updated in response to new information and requirements in a timely fashion to ensure that they remain viable and relevant, and replaced by alternatives when they are not. Users are an integral part of the overall process, providing input that influences the types of plans that are generated, the number of options to consider, failure assessments, plan repair strategies, and overall control of system behavior.

CPEF leverages several sophisticated AI technologies as components. SIPE-2 (Wilkins 1988) provides HTN planning and minimal-perturbation plan repair. The Advisable Planner (Myers 1996b) supports user provision of advice to guide the process of plan generation. The Procedural Reasoning System (PRS) (Georgeff & Ingrand 1989; Myers 1993), a hierarchical reactive control system, is used as both an executor for plans, and a high-level controller for the overall system. The Multiagent Planning Architecture (Wilkins & Myers 1998) provides distributed communication and plan storage services.

CPEF supports both *direct* models of execution, for which process actions are performed by the system itself, and *indirect* models of execution for which the system supervises execution of plans by a collection of distributed execution entities. The indirect model of execution is essential for domains where direct software control of plan entities is impossible, including many classes of WFM problems.

CPEF employs a *procedure library* that stores both plans and operators. Elements of the library span multiple abstraction levels and are usable for both plan generation and execution, thus supporting smooth transitions between them. In particular, plan generation can proceed to arbitrary levels of refinement, with the executor applying additional procedures at runtime to refine tasks to executable activities. Planning and execution operate asynchronously, in a loosely coupled fashion, with agents communicating domain knowledge, plans, requests, and situation enformation as required to fulfill their respective responsibilities.

The creation and deployment of *monitors* is a critical part of CPEF. Users can define a wide range of monitors; additionally, certain kinds of monitors are created automatically for generated plans, as a way of detecting situation changes that may invalidate a plan. One research focus for CPEF is to develop more flexible and powerful models of failure detection and recovery. For example, CPEF supports the specification, monitoring, and repair of the following generalized types of failures.

Unattributable Failures occur when no action has failed or assumption been violated, yet some assessment (human or automated) has deemed the current plan inadequate. Unattributable failures arise because planning operators don't model the real world with sufficient fidelity.

Aggregate Failures require the failure of a set of related activities. Aggregation is important for failure identification when processes include redundant actions as a way of improving their robustness.

To date, repair in CPEF has focused on minimal-perturbation dependency structure methods with extensions to accommodate our generalized failure models. We intend to augment these methods with heuristic local repairs in the near future.

CPEF, while domain-independent technology, is being developed within the context of supporting a Joint Forces Air Component Commander (JFACC) with the management of air campaigns (Lee 1998). CPEF has been successfully managed the generation, repair, and execution of large-scale air superiority plans (with several thousand nodes) while remaining responsive to changes in guidance and tasking within a simulated operating environment.

SWIM

In the SWIM project, we are developing a WFM system (jointly with CIRL, University of Oregon) to support the management of assets and resources for advanced Intelligence, Surveillance and Reconnaissance (ISR) capabilities. On a daily basis, intelligence planners are faced with the task of coordinating multiple ISR assets to maximize available information about the battlefield in order to increase the effectiveness of the deployed forces . Effective integration of the automated information discovery, acquisition, exploitation and dissemination with multi-asset synchronization within ISR requires some form of intelligent process management. SWIM aims to provide a highly adaptive WFM system that will enable more effective and efficient management of available assets and information. The workflow manager must address traditional workflow uncertainties, a volatile operating environment, frequently changing goals and operating practices, and the unexpected addition and subtraction of processing agents during runtime.

Our contribution is focused on reactive control and scheduling. It will leverage many of the reactive control capabilities from CPEF, augmenting them with advanced resource allocation, process reasoning, and scheduling capabilities.

The expressive activity representations of CPEF will be used to capture the activity, capability, and information product knowledge required to reason about workflow processes, while the monitoring, reactive execution, and dynamic repair capabilities will be employed to support adaptivity of active processes. The ability to efficiently combine declarative and procedural knowledge will allow a reactive controller exploit knowledge about the domain. At later stages of the project, hierarchical planning techniques will be employed to provide automatic process generation. In addition techniques for process combination and consolidation will be explored in an effort to increase both long and short term adaption of processes.

Advanced resource allocation/scheduling techniques (based on Adaptive Constraint Satisfaction (Borrett, Tsang, & Walsh 1996)) and fast anytime algorithms (Joslin & Clements 1998) combined with capacity analysis (Berry 1992) will be used to task agents. The tasks in the activity list, will exist at different levels of abstraction (Berry, Choueiry, & Friha 1992). Some might be to achieve strategic objectives, while others might be to perform a specific set of collection tasks within a set time horizon. The activity manager will apply the most appropriate algorithms, given the agents involved and abstraction level. There will always be a legal schedule of activities ready for distribution. However, the system will constantly adapt the schedule to reflect activities and changes in the world. Incoming information that affects the current schedule will also exist spanning multiple abstraction levels, and possibly different temporal intervals. Triggered by monitors, the activity manager will select the strategy most appropriate to the new situation and evolve the current schedule appropriately.

Conclusions

The AI communities of reactive control, planning, and scheduling have much to contribute to the development of powerful, adaptive workflow engines. Reactive control provides a rich legacy of work on process representation, monitoring, execution management, and adaptivity that provides a powerful starting point for constructing workflow engines. Scheduling provides the basis for resource and task allocation, as well as methods for incremental repair in response to dynamics within a given domain. Planning provides methods both for process synthesis and adaptation of processes in response to new requirements or operating conditions.

Certainly, workflow engines require more than just a direct integration of state-of-the-art capabilities within these fields. First, advances are required in terms of the capabilities supported by each of the technologies, primarily to support richer process descriptions and improved adaptability. In addition, workflow has special characteristics and requirements that have not been considered in detail by members of these AI communities. Included among these are the need for rich organizational models, security, scalability, and interoperability with legacy systems. Beyond the capabilities of the individual technologies, better integration frameworks are needed that provide richer interoperability of functionality, as required to support adaptivity and responsiveness for highly dynamic environments.

To date, there has been only minimal contact between the workflow community and the AI communities of reactive control, planning, and scheduling (for example, (Drabble, Lydiard, & Tate 1998)). The broader exchange of problems and techniques between these two groups would lead to more rapid development of the kinds of adaptive, intelligent systems required by both.

References

Ambite, J. L., and Knoblock, C. A. 1997. Planning by rewriting: Efficiently generating high-quality plans. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*.

Attie, P.; Singh, M.; Emerson, E.; Sheth, A.; and Rusinkiewicz, M. 1996. Scheduling workflows by enforcing intertask dependencies. *Distributed Systems Engineering Journal* 3.

Berry, P. M.; Choueiry, B. Y.; and Friha, L. 1992. A distributed approach to dynamic resource management based on temporal abstractions. *Journal of Intelligent Systems Engineering* 7:227–242.

Berry, P. M. 1992. The PCP: A predictive model for satisfying conflicting objectives in scheduling problems. *Artificial Intelligence in Engineering* 7:227–242.

Borrett, J. E.; Tsang, E. P. K.; and Walsh, N. R. 1996. Adaptive constraint satisfaction: The quickest first principle. In *Proceedings of the 12th European Conference on Artificial Intelligence*.

Cichocki, A.; Helal, A. S.; Rusinkiewicz, M.; and Woelk, D. 1998. *Workflow and Process Automation: Concepts and Technology*. Kluwer.

Drabble, B.; Lydiard, T.; and Tate, A. 1998. Workflow support in the air campaign planning process. In *Proceedings of the AIPS Workshop on Interactive and Collaborative Planning*.

Drummond, M.; Swanston, K.; Bresina, J.; and Levinson, R. 1993. Reaction-first search. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, 1408–1414.

Drummond, M. 1989. Situated control rules. In Brachman, R., and Levesque, H., eds., *Principles of Knowledge Representation and Reasoning: Proceedings of the First International Conference*. Morgan Kaufmann.

Eder, J., and Liebhart, W. 1995. The workflow activity model WAMO. In *Proceedings of Third International Conference on Cooperative Information Systems*.

Erol, K.; Hendler, J.; and Nau, D. S. 1994. Semantics for hierarchical task-network planning. Technical Report CS-TR-3239, Computer Science Department, University of Maryland.

Firby, R. J. 1994. Task networks for controlling continuous processes. In *Proceedings of the Second International Conference on AI Planning Systems*. Menlo Park, CA: AAAI Press.

Garcia-Molina, H.; D. Gawlick, J. K.; Kleissner, K.; and Salem, K. 1990. Coordinating multi-transaction activities. Technical Report CS-TR-247-90, Princeton University.

Gasser, L., and Huhns, M. N. 1989. *Distributed Artificial Intelligence*. Morgan Kaufmann.

Georgeff, M. P., and Ingrand, F. F. 1989. Decision-making in an embedded reasoning system. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*.

Godefroid, P., and Kabanza, F. 1991. An efficient reactive planner for synthesizing reactive plans. In *Proceedings of the Ninth National Conference on Artificial Intelligence*.

Hammond, K. 1989. *Case-Based Planning: Viewing Planning as a Memory Task*. Academic Press.

Hammond, K. 1990. Explaining and repairing plans that fail. *Artificial Intelligence* 45:173–228.

Hollingsworth, D. 1994. The workflow reference model. Technical Report TC00-1003, Workflow Management Coalition.

Howe, A. E., and Cohen, P. R. 1991. Failure recovery: A model and experiments. In *Proceedings of the Ninth National Conference on Artificial Intelligence*.

Howe, A. E. 1995. Improving the reliability of artificial intelligence planning systems by analyzing their failure recovery. *IEEE Transactions on Knowledge and Data Engineering, Special Issue on Dependability of AI Systems*.

Joslin, D., and Clements, D. 1998. Squeaky wheel optimization. In *Proceedings of the 15th National Conference on Artificial Intelligence*. AAAI Press.

Kambhampati, S., and Hendler, J. 1992. A validation-structure-based theory of plan modification and reuse. *Artificial Intelligence* 55(2):192–258.

Kerr, R., and Walker, R. 1989. A job shop scheduling system based on fuzzy arithmetic. In *Proceedings of the 3rd International Conference on Expert Systems and the Leading Edge in Production Planning and Control*, 433–450.

Lawrance, P. 1997. *WfMC Workflow Handbook 1997*. Wiley.

Lee, T. J. 1998. The air campaign planning knowledge base. Technical report, Advanced Automation Technology Center, Menlo Park, CA.

Lesser, V., and Corkill, D. 1983. The distributed vehicle monitoring testbed: A tool for investigating distributed problem solving networks. *AI Magazine* 4(3):15–33.

Lui, J.-S., and Sycara, K. P. 1998. Multiagent coordination in tightly coupled task scheduling. In Huhns, M. N., and Singh, M. P., eds., *Readings in Agents*. Morgan Kaufmann Publishers. chapter 2, 164–171.

Mohan, C.; Agrawal, D.; Alonso, G.; Abbadi, A. E.; Gunthor, R.; and Kamath, M. 1995. Exotica: A project on advanced transaction management and workflow systems. *ACM SIGOIS Bulletin* 16(1).

Musliner, D. J.; Durfee, E. H.; and Shin, K. G. 1993. CIRCA: A cooperative intelligent real-time control architecture. *IEEE Transactions on Systems, Man, and Cybernetics* 23(6).

Musliner, D. J.; Durfee, E. H.; and Shin, K. G. 1995. World modeling for the dynamic construction of real-time control plans. *Artificial Intelligence* 74(1).

Myers, K. L. 1993. *User's Guide for the Procedural Reasoning System*. Artificial Intelligence Center, SRI International, Menlo Park, CA.

Myers, K. L. 1996a. A procedural knowledge approach to task-level control. In *Proceedings of the Third International Conference on AI Planning Systems*. AAAI Press.

Myers, K. L. 1996b. Strategic advice for hierarchical planners. In Aiello, L. C.; Doyle, J.; and Shapiro, S. C., eds., *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifth International Conference (KR '96)*. Morgan Kaufmann Publishers.

Myers, K. L. 1998. Towards a framework for continuous planning and execution. In *Proceedings of the AAAI 1998 Fall Symposium on Distributed Continual Planning*. Menlo Park, CA: AAAI Press.

Pryor, L., and Collins, G. 1996. Planning for contingencies: A decision-based approach. *Journal of Artificial Intelligence Research* 4:287–339.

Reuter, A. 1989. Contracts: A means for extending control beyond transaction boundaries. In *In Proceedings of Third International Workshop on High Performance Systems*.

Sadeh, N. 1994. Mirco-opportunistic scheduling. In Zweben, M., and Fox, M., eds., *Intelligent Scheduling*. Morgan Kaufmann Publishers. chapter 4.

Smith, S. F. 1994. Reactive scheduling systems. In Brown, D., and Scherer, W., eds., *Intelligent Scheduling Systems*. Kluwer Publishing.

Sycara, K., and Miyashita, K. 1992. Incremental schedule modification. In *Proceedings of the AAAI Spring Symposium on Computational Considerations in Supporting Incremental Modification and Reuse*.

Turksen, I. 1997. Three fuzzy theory approaches for scheduling system design. In *Proceedings of the First International Workshop on Approximate Reasoning In Scheduling (ARS'97)*.

Veloso, M. M.; Pollack, M. E.; and Cox, M. T. 1998. Rationale-based monitoring for planning in dynamic environments. In *Proceedings of the Fourth International Conference on AI Planning Systems*.

Veloso, M. M. 1992. *Learning by Analogical Reasoning in General Problem Solving*. Ph.D. Dissertation, Carnegie Mellon University, Pittsburgh, PA.

Wilkins, D. E., and Myers, K. L. 1998. A multiagent planning architecture. In *Proceedings of the Fourth International Conference on AI Planning Systems*.

Wilkins, D. E. 1985. Recovering from execution errors in SIPE. *Computational Intelligence* 1(1):33–45.

Wilkins, D. E. 1988. *Practical Planning: Extending the Classical AI Planning Paradigm*. Morgan Kaufmann.

Winograd, T., and Flores, F. 1987. *Understanding Computers and Cognition*. Addison-Wesley.

Zlotkin, G., and Rosenschein, J. 1989. Negotiation and task sharing among autonomous agents in cooperative domains. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*.

Zweben, M.; Daun, B.; Davis, E.; and Deale, M. 1994. Scheduling and rescheduling with iterative repair. In Zweben, and Fox., eds., *Intelligent Scheduling*. Morgan Kaufmann Publishers. chapter 8.