# Overview of Adaptable Situation-aware Secure Service-based (AS$^3$) Systems

[1]S. S. Yau, [1]H. Davulcu, [2]S. Mukhopadhyay, [1]D. Huang, [1]Y. Yao, and [1]H. Gong

*[1]Arizona State University, Tempe, AZ 85287-8809, USA*
*[1]{yau, hasan.davulcu, dazhi.huang, yisheng.yao, haishan.gong}@asu.edu}*

*[2]West Virginia University, Morgantown, WV, USA*
*[2]supratik.mukhophadyay@mail.wvu.edu*

## Abstract

*Service-based systems are distributed systems which have the major advantage of enabling rapid composition of distributed applications, regardless of the programming languages and platforms used in developing and running different components of the applications. In these systems, various capabilities are provided by different organizations as services interconnected by various types of networks. The services can be integrated following a specific workflow to achieve a mission goal for users. For large-scale service-based systems involving multiple organizations, high confidence and adaptability are of prime concern in order to ensure that users can use these systems anywhere, anytime with various devices, knowing that their confidentiality and privacy are well protected and the systems will adapt to satisfy their needs in various situations. Hence, these systems must be adaptable, situation-aware and secure. In this paper, an approach to rapid development of adaptable situation-aware secure service-based (AS$^3$) systems is presented. Our approach enables users to rapidly generate, discover, compose services into processes to achieve their goals based on the situation and adapt the composed processes when situation changes.*

## Keyword:

Adaptable situation-aware secure service-based systems, adaptive workflow synthesis, AS$^3$ logic, AS$^3$ calculus, distributed trust management, hierarchical situation-awareness, security, service-oriented architecture

## 1. Introduction

Service-based systems are distributed computing systems, which have the major advantage of enabling rapid composition of distributed applications, regardless of programming languages and platforms used in developing and running different components of the applications. In service-based systems, various capabilities are provided by different organizations as *services*, which are software/hardware entities with well-defined interfaces to provide certain capability over wired or wireless networks. The services can be integrated following a specific *workflow*, which is a series of cooperating and coordinated activities designed to carry out a well-defined process to achieve a mission goal for users. Our MURI project, "Adaptable Situation-Aware Secure Service-based (AS$^3$) Systems" aims at conducting basic research on automating the development, deployment and operations of robust and secure service-based systems to achieve declaratively specified mission goals with multiple QoS requirements in dynamic and unreliable environments.

The motivation of our project is to facilitate the rapid adoption of service-based systems in many large-scale distributed systems, such as Grid and Global Information Grid (GIG), for various distributed applications including collaborative scientific and engineering work, e-business, healthcare, military, and homeland security. For these large-scale service-based systems, *high confidence* and *adaptability* are of prime concern. It is very important to ensure that users can use these systems anywhere, any time using various devices (ranging from handheld devices to PCs), knowing that their confidentiality and privacy are well protected and the systems will adapt to their needs in various situations. Therefore, these systems must have the following properties:

(1) *Adaptability*. In these systems, services may become unavailable due to distributed denial-of-service attacks or system failures, and new processes may be created in runtime to fulfill users' new mission goals. Hence, the systems must have the capabilities to change their configurations to provide high availability, or to adapt their behavior to satisfy the new goals in dynamic environments.

(2) *Situation-awareness* (SAW). SAW is the capability of being aware of situations and adapting the system's behavior based on situation changes [Yau02a-b]. SAW is essential for high confidence and adaptable service-based systems since it is needed for determining adaptive processes to achieve users' goals, and enforcing flexible security policies.
(3) *Security*. To provide high confidence to users, these systems must have the capabilities of authenticating users and service providers, verifying the integrity of services, protecting the confidentiality of information, controlling the access to services based on security policies, and detecting malicious services and users.

Although various techniques have been proposed to improve the security and provide dynamic service composition of service-based systems [IBM04a-c, OAS04a-b], so far there are no effective enabling techniques for developing *Adaptable Situation-aware Secure Service-based* (AS$^3$) *Systems*. In our MURI project, we are developing a declarative unifying logic-based approach to developing service-based systems with situation-awareness, distributed security policy management and enforcement, and adaptive workflow management while preserving overall correctness and consistency of the systems. During the first year of this project, we have accomplished the following:

(a) An AS$^3$ logic for supporting the specification, verification and synthesis of AS$^3$ systems with various QoS requirements, such as security, SAW and real-time.
(b) An AS$^3$ calculus for providing a formal programming model for AS$^3$ systems.
(c) Declarative models of hierarchical SAW and security policies, and mappings between model representations and AS$^3$ logic specifications for supporting requirement analysis and converting requirements to AS$^3$ logic specifications.
(d) An agent-based distributed trust management approach for efficiently and effectively managing and enforcing situation-aware security policies.
(e) A preliminary version of adaptive workflow synthesis using domain-specific knowledge as the situation evolves during the workflow execution.

## 2. An Example

Consider an AS$^3$ system, as shown in Figure 1, connecting the Intelligent Transportation System (**ITS**), Police Departments (**PD**), Fire Departments (**FD**), and Ambulance Services (**AMS**), for maintaining traffic safety and coordinating various responders (**ITS**, **PD**, **FD** and **AMS**) in emergency situations.

**ITS**, **PD**, **FD** and **AMS** provide various capabilities as services in the system. A Mission Planner (**MP**) service in the system is used to automatically synthesize workflows using these services to fulfill various mission goals. The following *Accident Response* scenario illustrates the need for adaptability, situation-awareness and security provided by AS$^3$ system.

A 911 call center gets a report that there is an accident at location **L** during the rush hour. In response to such a situation, the following workflow (as shown in Figure 2) is automatically generated by the **MP** to coordinate field rescue operations and mitigate the effects of the accident. The following is the step-by-step description of the control flow logic in the workflow:

(1) A nearby Helicopter (**H**) provided by **PD** is identified and sent to **L**. Meanwhile:
   - Two Police Patrol Cars (**CAR**), one Fire Engine (**FE**) and one Ambulance (**AMB**) closest to **L** are also identified and sent to **L**.
   - An Emergency Road Closure (**ERC**) service provided by the **ITS** is also invoked to *display road closure messages* on the big screens along the affected roads.
(2) Once **H** arrives at **L**, it starts to *serve as a base station* for all communications among responders at the accident site.
(3) Upon arriving at **L**, the police officers *set up a perimeter* to secure the accident site.
(4) After the police perimeter is set up, the police inform the **FE** and **AMB**, and the **FE** and **AMB** *enter the accident site*.
(5) The fire fighters start to *rescue* the passengers trapped in the damaged vehicles. Meanwhile:
   - The paramedics on the AMB start to *assess the status* of the injured passengers for deciding the appropriate medical care for them.
(6) After the fire fighters get the passengers out of the damaged vehicles, the paramedics carefully put the injured passengers on the **AMB** and take them to a nearby hospital. Meanwhile:
   - The **FE** also leaves **L** after the fire fighters finish their work.
(7) The police officers remove the perimeter, and the **CAR**s leaves **L**. Meanwhile:
   - The **ERC** provided by the **ITS** is invoked again to notify drivers in nearby roads that the road closure has ended.

In the AS[3] system, a set of coordination agents will be automatically generated to monitor the situations, execute the abovementioned workflow, and adapt the workflow when necessary. The constraints that the workflow needs to satisfy include the following:

- ❖ All the responders should arrive at the accident site within fifteen minutes.
- ❖ Any **CAR**, **FE**, **AMB**, or **H** that are serving at one accident site should not be dispatched to another accident site before completing their jobs at the accident site.
- ❖ Injured passengers in critical conditions should be brought to a nearby hospital within fifteen minutes after they are rescued from their damaged vehicles.
- ❖ Any coordination **agent** should only follow the commands from a trusted **MP**, being authenticated and delegated by a trusted party (the proper authority). Only after **CARs** leaves from **L**, **ERC** can end the road closure.

In a perfect world, the above workflow would execute successfully and perform all the needed tasks to complete the rescue operations. However, various things may go wrong during execution, For examples, a service may fail to terminate successfully, an unperceived exception condition may arise at run-time or more resources may be needed in order to fulfill a user's goal. Since it is almost impossible to identify all control and correction steps before execution time, the system must provide the capability to adapt the workflow at run-time with the following *dynamic reconfiguration constraints:*

- ❖ **Resource failure**: An ambulance can transport at most two injured passengers at the same time, and hence the **MP** should send another ambulance within five minutes to carry additional injured passengers.
- ❖ **Service failure:** If the police fail to set up a perimeter within fifteen minutes after the 911 call center gets an accident report, FE and AMB can enter the accident site regardless a police perimeter has been set up or not.
- ❖ **Exception Condition:** If the paramedics determine that one of the injured passengers is in *critical*

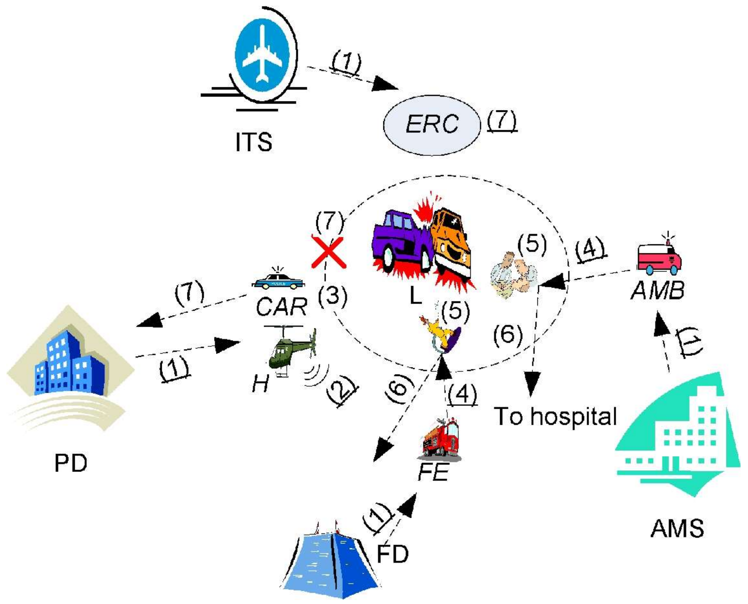

**Figure 1.** The motivating example
Underlined steps are automatically initiated by the coordinating agents, and other steps are manually initiated by human. Entities with italic notations are services.

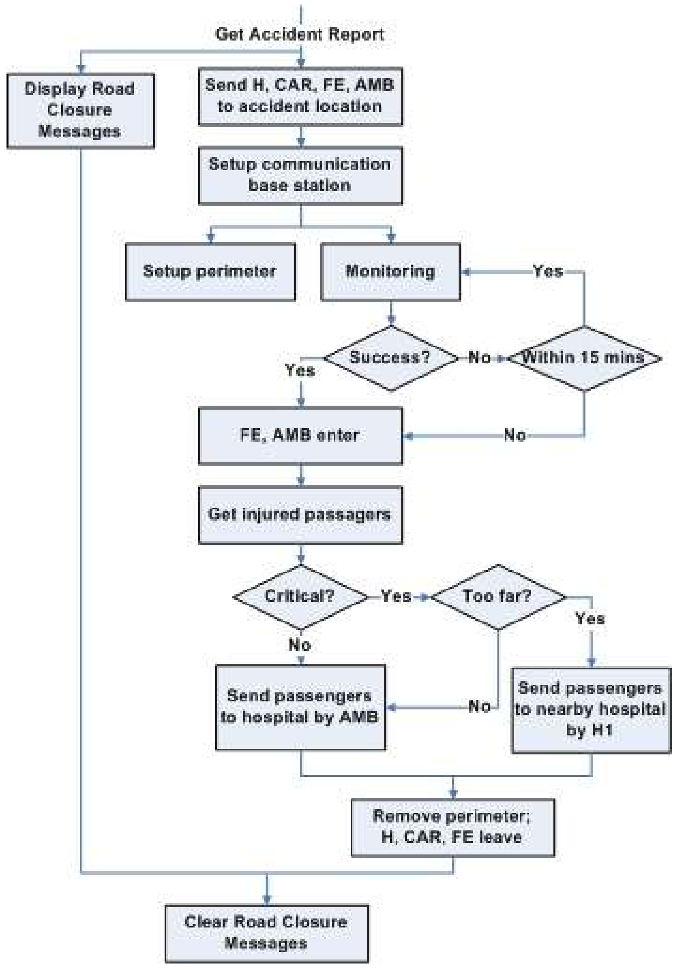
Get Accident Report
Display Road Closure Messages
Send H, CAR, FE, AMB to accident location
Setup communication base station
Setup perimeter
Monitoring
Yes
Success?
No
Within 15 mins
Yes
FE, AMB enter
No
Get injured passagers
Critical?
Yes
Too far?
Yes
No
Send passengers to hospital by AMB
No
Send passengers to nearby hospital by H1
Remove perimeter; H, CAR, FE leave
Clear Road Closure Messages

**Figure 2.** The workflow in the motivating example

*condition* in Step (6), another helicopter (H1) is discovered and used to transport the passenger in critical condition to the hospital.

## 3. Background

The interdisciplinary research in this project is related to the following research areas: deliberative and reactive systems, languages and formalisms for service modeling and composition, distributed trust management, situation-awareness, and workflow planning and scheduling. In this section, the background of these related research areas will be discussed.

### 3.1 Deliberative and Reactive Systems

Deliberative systems (also referred to as deliberative agents) are systems based on the "Sense-Plan-Act" (SPA) model [Ark98], in which agents sense the environment, jointly plan their actions, and act cooperatively to achieve a well-defined goal [Dav94, Dor97, Nam01]. Such systems usually consist of a planner and a world model [Dav94, Dor97, Nam01]. In the world model, the actions and events in the world are represented symbolically, usually using some logic like first order logic based situation calculus [McC69]. The planner utilizes some AI planning techniques [Abe04, Bac01, Cha87] to generate a sequence of actions that needs to be performed to achieve the goal based on the world model. Although deliberative systems can generate complicated coordinated actions and allow learning and prediction, the difficulty in modeling the world and keeping the world model up-to-date and the high computational complexity in planning make deliberative systems not suitable for applications requiring real-time response in dynamic environments.

Reactive systems (also referred to as reactive agents), on the contrary, do not require any world model and planning. A reactive system works in a stimulus-response manner, for which a set of "Sense-Act" rules are defined to control the reaction of the system to the event sensed by the system [Bro91ab, Nie03]. This type of systems utilizes simple low-level reactions based on the information collected from environments to obtain complex, goal-related and intentional behavior [Bro91ab, Nie03]. Reactive systems can provide very fast response to the events detected. But due to the lack of a world model and a planner, reactive systems can only act according to the pre-defined rules, and cannot plan actions ahead of time. Also, pure reactive systems do not provide any learning capability. Hence, reactive systems are not adaptive.

Much research has been done to develop hybrid (of reactive and deliberative) systems to take advantages of both types of systems and overcome their limitations [Bly93, Nwa96, Nou97, Woo02, Urd03, Vas04]. In hybrid systems, techniques of deliberative systems are used at the top layer to perform high-level planning and learning, while techniques of reactive systems are used at the bottom layer to drive the low-level reactions to environment changes. However, an intermediate layer must be added to mediate reactive and deliberative layers due to the significant difference between two types of techniques. Currently, there is no unifying formal approach for developing such kind of systems.

The AS$^3$ systems proposed in our project can be considered as hybrid systems. In this project, we will develop a declarative unifying logic-based approach to developing AS$^3$ systems with hierarchical SAW for reactive behavior and adaptive workflow management for deliberative actions.

### 3.2 Languages and Formalisms for Service Modeling and Composition

Service-Oriented Architecture (SOA) aims at facilitating seamless integration, interoperabilty and deployment of distributed applications. SOA is increasingly deployed for building distributed service-oriented systems. SOA promotes component reuse as well as provides separation of concerns by separating the business logic of an application from the implementation details of the individual interactions of the components. A service can be viewed as a stand-alone software/hardware module performing a specific function. For example, a sensor can be viewed as a stand-alone hardware module that provides the service of "sensing its environment". Services can run on spatially distributed individual nodes of a network. The functionalities provided by a service are implemented by one or more methods that may themselves use functionalities of other services. Services can be dynamically discovered and invoked. A service has an interface that is exposed to the environment and is used by other modules for invoking functionalities provided by it. The implementation details of the service as well as its location are supposed to be hidden from the users. A service registers itself with a service registry that allows other modules to locate the service as well as obtain information about its interface. New services having complex functionalities can be created by composing existing services. SOA loosely couples its constituent individual modules. Services and the clients that can access them can communicate among each other either asynchronously (documented-oriented) or synchronously (remote procedure call).

In web-service-based SOA, services communicate with clients through well established XML-based protocols like SOAP (Simple Object Access Protocol). The enabling technologies for developing such SOA include XML, WSDL (Web Services Definition Language) [W3C], SOAP (Simple Object Access Protocol) [New02], and UDDI (Universal Description Discovery and Integration) [New02]. The XML framework provides a language and techniques for defining and processing semi-structured data. WSDL provides an XML-based language for describing web services interfaces. A WSDL file not only describes a service interface, but also provides a set of locations for accessing the service. SOAP provides a framework for XML-based messaging. It is essentially a protocol for one way message transmission, but a remote procedure call-like mechanism can be emulated. UDDI provides a directory for registering and discovering web services. Web services register themselves with a UDDI registry by sending a SOAP message. Once registered, the UDDI registry includes the URL of a WSDL file that contains the description of the web service. When a client queries the UDDI registry using a SOAP message, it receives the WSDL file corresponding to the web service that directs it how to format a SOAP message that reaches the service. On receiving the SOAP message, the SOAP processor at the service end can map it to an input to the method implementing the required functionality referred to in the message. The output of the method (if any) is formatted as a SOAP message and sent back to the client. An alternative to using WSDL, SOAP and UDDI is ebXML (Electronic Business XML) that provides a framework for negotiation between services.

Other popular languages for describing web services include ontology-based languages like RDF (Resource Description Framework). RDF follows XML syntax and provides a data model for representing services/resources.

## 3.3 Situation-awareness

SAW is the capability of being aware of situations and adapting the system's behavior based on situation changes [Yau02a-b]. A *situation* is a set of context attributes over a period of time that is relevant to future system behavior. A *context* is any instantaneous, detectable, and relevant property of the environment, the system or users, such as time, location, wind velocity, temperature, available bandwidth, invocation of action, and a user's schedule [Yau02a-b]. *Hierarchical situation-awareness (hierarchical SAW)* is the capability of being aware of situations in different abstract levels and properly reacting to situation changes. In service-based system, the purpose is to minimize human effort for developing and maintaining distributed applications, regardless of programming languages and platforms used. When apply SAW to distributed service-based systems, such as AS$^3$ systems, it raises a number of challenging issues, such as dynamic system infrastructure, high scalability of distributed interacting entities, and heterogeneous resources. Therefore, SAW in AS$^3$ systems should be flexible, scalable, reliable and easy to develop, as well as provide support for dynamic service discovery and execution.

Situation-awareness has been studied in artificial intelligence [Rus03], human-computer interactions [Car83] and data fusion community [Hal01]. Existing work could be divided into two categories. One category focuses on modeling and reasoning SAW [Mcc69, Pin94, Mcc00, Mat03a-b, Pla03], and the other focuses on providing toolkit, framework or middleware for development and runtime support for SAW [Dey01, Rom02, Ran03, Cha03, Yau02a-b]. These notable work can hardly meet the challenges due to lack of a systematic way to deal with the increasingly dynamic operating environments of service-based systems. In our approach, a declarative model is developed to help application developers analyze and specify SAW requirements. A logic is developed to formally specify the declarative model and support the automated synthesis of calculus terms through theorem proving. The calculus terms will be further translated into codes, running as distributed entities (typically, SAW agents) to perform runtime context acquisition and situation analysis.

## 3.4 Distributed Trust Management

Trust is "the capacity to commit oneself to fulfilling the legitimate expectations of others, is both the constitutive virtue of, and the key causal precondition for the existence of any society" [Dun84]. In our daily life, we frequently make trust decisions, directly or indirectly. For example, when we take a taxi, we trust the taxi driver will take us to the destination location safely and charge us a reasonable fee. In AS$^3$ systems, service providers and service users form virtual societies, in which we face information overloads, increased uncertainty and risk of using services provided by third-parties. Therefore, trust plays an essential role for ensuring successful and secure interactions among members. We refer *trust management* to an approach to collecting, specifying, analyzing, and presenting evidences to make decisions on whether a principal is trusted. When applied to AS$^3$ systems, evidences include (1) s*ecurity policies*, which are detail statements that embody the goals of protecting the security-critical services, (2) *credentials*, which represent delegations of trust among principals, and (3) c*urrent system situation*: for example, a system has been compromised. A *principal* is an entity with a unique identity in AS$^3$ system that requests permissions. An *entity* is any concrete or abstract object in AS$^3$ system, including a user, a process, a service, or a computation/communication resource. In AS$^3$ system, each principal may potentially interact with any other principal. For better manageability, a set of principals with some common properties (such as roles) may be organized as a *group*. A *group* may be externally identified as a single organizational principal.

The trust management for AS$^3$ system should be flexible, scalable and adaptable to the changing environments and user requirements and provide security support for dynamic service discovery and execution. Existing security solutions for service-based systems [Nak02, Nae03, IBM04c, OAS04a] and trust management approaches [Bla96,99, Chu97, Jim01, Li03, Gav04] in distributed systems can hardly meet this challenge due to lack of a systematic way to deal with the increasingly dynamic operating environments of service-based systems. A distributed trust management (DTM) framework, together with a formal model, is essential to provide a systematic approach to specifying and enforcing security policies, including authentication, access control and delegation policies, which allow direct authorization of security-critical actions in the dynamic environments of AS$^3$ systems.

## 3.5 Workflow Planning and Scheduling

Planning techniques [Sri04] have been used to schedule dynamically changing workflows. The classical planning problem [Nau04] is specified by describing the initial state of the world, the desired goal state, and a set of deterministic actions. The objective is to find a sequence of these actions, which, when executed from the initial state, lead the agent to the goal state. Each action has preconditions to be satisfied in order to execute it, and has effects which are released after the execution of that action. For instance, in the "travel" domain, **fly(Phoenix, San Diego)** is an action. A person must be located in Phoenix to do that action as a precondition, and after flying to San Diego, the person's location will be changed to San Diego as the effect. A problem is composed of the initial state and the goal state [Kam97].

Classical planners are simply state transition systems with the restrictive assumption of implicit time and resources. This restricted model is quite useful for studying the logics and computational aspects of planning with simple state-transition operators. However, in many applications, this resticted model is not realistic for the following reasons:

**Quality of Service Requirements:** In reality, actions do occur over a time span and there exists certain resources. Furthermore, often goals in a plan are meaningful only if they are achieved within a time limit. In a dynamic environment, various events may be expected to occur at future time periods. Hence, actions have to be located in time with respect to expected events and to goals [Bac01].

**Lack of Domain Knowledge:** Classical planners are also domain independent, that is, they cannot take advantage of domain knowledge [Nau99]. In planning problems, search spaces are all exponential in size, and blind search in any of them is ineffective. Hence, a key problem facing planning systems is that of guiding or controlling search. Domain knowledge is the way that we can embed *control information* [Bac00].

**Execution Time Problems:** Since classical planning techniques assume static goals and environment with complete and reliable domain information, in real-world problems most of the generated plans will fail. The reasons for the failures are due to various execution time problems such as dynamic external environment [Abe04], incomplete domain information [Gar02], and unreliable situation information [Kro03, 04].

For workflows with timing and resource constraints, the workflows need to be properly scheduled with appropriate resources being assigned to them. Classical scheduling algorithms, such as Rate Monotonic and Earliest Deadline First, depend on *a priori* knowledge of workload and systems, and hence they can only be used in predictable environments [Liu73, Sta88]. Dynamic scheduling systems, such as Spring, provide performance guarantees upon new task arrivals with on-line admission control, but still require *a priori* task set characterizations [Zha87]. Feedback control real-time scheduling can provide robust performance guarantees in unpredictable environments, but also require *a priori* task set characterizations and are not designed for workflow scheduling [Abd97, Cac00, Lu01, Lu02]. Logic-based workflow scheduling techniques can be used to handle multiple constraints on workflow execution, but cannot deal with unpredictable environments [Sen02, Dav98]. Grid workflow scheduling techniques can provide performance guarantees for workflows in Grid, but are specially designed for scientific computing (computation intensive, high parallelism, little coordination) [Yan03, Her04, Kea04, Wic04, Yu04].

## 4. Overview of Our Approach to AS$^3$ Systems

The concept of AS$^3$ systems is to address these requirements using a declarative unifying logic-based approach, which can also be applied to general service-based systems. In this section, we will give an overview of our approach and highlight our accomplishments in the first year of the project.

As depicted by **Figure 3**, our overall approach to developing, deploying and operating AS$^3$ systems consists of the following steps:

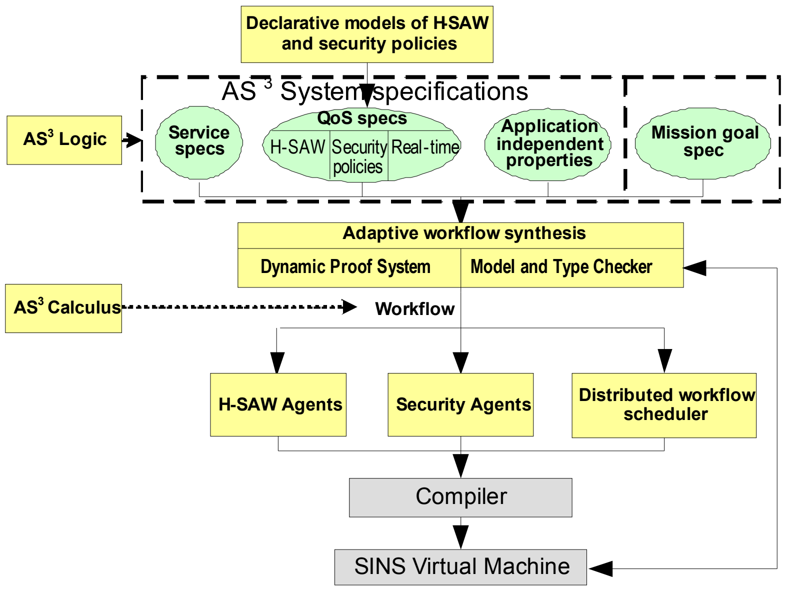

**Figure 3**. Our overall approach and research aspects

(1) **Develop a formal programming and specification model, involving a calculus and a logic, to support the specification, verification and synthesis of AS$^3$ systems with various QoS requirements, such as security, SAW and real-time.**

We have developed an AS$^3$ calculus to provide a formal programming model for AS$^3$ systems. The AS$^3$ calculus can model timeouts and failures, and has well-defined operational semantics that involves interactions of external actions and internal computations. The external actions include communication between processes, leaving and joining groups/domains. The internal computations are method calls of named services. The AS$^3$ calculus allows modeling various QoS requirements and dynamic adaptation at runtime by processes. In particular, it provides a hierarchical domain-based security model, which requires two processes to move into the same named domain before they are allowed to communicate with each other. The AS$^3$ calculus also has a well-defined equational theory that allows for modeling redundancy for fault-tolerance as well as formal reasoning using a simulation relation.

We have also developed an AS$^3$ logic, a hybrid normal modal logic [Bla03] for specifying AS$^3$ systems. The logic has both temporal and spatial modalities for expressing situation information as well as modalities for expressing communication, service invocation, joining and leaving groups. It provides atomic formulas for expressing relations among variables and nominals for identifying agents. The vocabulary of the logic does not include function symbols, but has nominals that identify agents and constant symbols that are interpreted over a domain. Models for the logic are (annotated) processes in the AS$^3$ calculus. The AS$^3$ logic allows declarative specification of QoS requirements, such as security, situation-awareness and real-time requirements. A novel proof system of AS$^3$ logic allows the synthesis of AS$^3$ calculus terms from declarative specifications in AS$^3$ logic as well as allowing checking consistency of specifications. The model checking problem for AS$^3$ logic is decidable for image-finite processes, and hence it allows verification of application-independent properties, like deadlock freedom.

(2) **Develop declarative models of QoS requirements for AS$^3$ systems, and mappings between model representations and AS$^3$ logic specifications.**

Since the AS$^3$ logic developed in (1) is a modal logic, which is difficult to use for developers who are not experts in logics, it is necessary that the developers can analyze and specify their requirements using some declarative models to achieve rapid development of AS$^3$ systems. The model representations of the requirements should then be automatically mapped to appropriate AS$^3$ logic specifications. So far, we have developed models of hierarchical SAW [Yau05c] and security policies [Yau05a-b], and the associated mappings to AS$^3$ logic specifications.

❖ **A declarative model for hierarchical SAW in AS$^3$ systems.**

*Situation-Awareness (SAW)* is essential for AS$^3$ systems because it is needed for enforcing security policies and for adapting workflows when situation changes. *Hierarchical situation-awareness (hierarchical SAW)* is the capability of being aware of situations in different abstract levels, which correspond to a command and control hierarchy formed by users or agents, and properly reacting to situation changes. We consider a service as a process, which can accept inputs from other processes and produce outputs. Hence, a service-based system can be considered as a collection of parallel processes, each of which can send/retrieve data to/from other processes. Therefore, modeling hierarchical SAW in AS$^3$ systems includes two aspects: (1) modeling situations, and (2) modeling the relations between situations and processes.

In our model for hierarchical SAW, a situation is either an ***atomic situation*** or a ***composite situation***. An ***atomic situation*** is defined as a term composed of operations on context instances and constants. Since context acquisition and operations on contexts are highly domain-specific and often involve low-level system processes, our model does not include the ways contexts are collected and the semantics of operations on contexts. Instead, we assume that each context is collected periodically by invoking at least one service in a service-based system, and a service can collect a context and also implement operations for preprocessing this context. A ***composite situation*** is composed of other situations through the usage of situation operators. The situation operators in our model include not only normal logical connectives (negation, conjunction, disjunction), but also temporal ("ever" and "always") and knowledge ("know") operations. The incorporation of these operators enables developers to model complex situations requiring reasoning on time and knowledge of agents. In particular, our model can be used to express the situation that timestamped common knowledge [Hal90] is attained among distributed processes. In our model, four relations between situations and processes are defined, which allow service providers and developers to define the situations that trigger, allow or prohibit the execution of processes in AS$^3$ systems, and also enable them to model control structures in processes, which are commonly used in service coordination. We have also developed a mapping between our declarative model and AS$^3$ logic specifications for hierarchical SAW, so that the AS$^3$ logic specifications can be automatically generated once the model representations for hierarchical SAW requirements are generated by developers.

- **A declarative model for security policies in AS$^3$ systems.**

Our security policy model is a declarative model that abstracts the concepts, such as "service", "Principal", "Delegation", "Permission" and "TrustPolicy", and relations among concepts, such as "Trust", "CanAccess", and "Delegate", in the security policies. We have also identified a fragment of AS$^3$ logic for specifying security policies, and developed algorithms for checking consistency, redundancy, and service accessibility. This model is essential to provide a logic-based approach to specifying and reasoning security policies, including authentication, access control and delegation policies, that allows direct evaluation of trust for security-critical actions in the distributed environments of AS$^3$ systems. To provide dynamic and flexible trust management in AS$^3$ systems, we have incorporated the following in our security policy model: (i) context and action history in dynamic evaluation of trust relationships among entities [Yau05a]; (ii) an automatic mapping from our security policy model to AS$^3$ logic such that security policies can be enforced by dynamically allocating entities in named domain hierarchy. The key insight of our security policy model comes from policy-based trust management [Bla96,99, Chu97, Jim01, Li03, Gav04], behavior history-based (also called reputation-based) trust management [Abd00, Che01, Yu02, Xio04, Shm05], and ambient calculus [Car00].

(3) **Develop the following enabling techniques for building and operating AS$^3$ systems based on the results from steps (1) and (2).**

AS3 systems require adaptive situation-aware behavior in the presence of system failures, overload, or damages and rapid reconfiguration in order to achieve users' dynamic mission goals. AS$^3$ systems also need to secure access to critical information infrastructure of distributed services based on flexible security policies. Our approach is based on a declarative unifying AS$^3$ logic for extending service-oriented architecture with automated hierarchical situation-awareness, distributed trust management, and adaptive workflow management while preserving the overall correctness and consistency of the specifications.

3a) ***Deductive Knowledge-Based Adaptive Planner with AS$^3$ logic.*** Knowledge-based planning systems are based on a philosophy of using whatever domain knowledge is available to solve the planning problem. This knowledge may include (i) services and goal structures, (ii) various kinds of QoS and security constraints, (iii) situation awareness, (iv) search control techniques, and (v) interaction with humans when needed to use their expertise [Liu03].

In our previous work [Dav04], we demonstrated the use of logic for modeling and reasoning about Web service contracts. Specifically, we proposed a logic, called CTR-S, which captures the dynamic aspects of contracting for services which involve two or more parties in a potentially adversarial situation. Our previous work assumed that we started with a complete specification of the workflow plan and the services comprised. However, our framework enabled reasoning with contract execution, which amounts to enforcement of dynamically evolving contractual temporal constraints. In our current work, we utilize the AS$^3$ logic, which is more expressive, and we relax our assumptions along two important dimensions in order to enable synthesis and reasoning with adaptive workflows:

(i) **Mining Incomplete Workflow Specifications using the Event Log**: We have developed techniques for mining incomplete workflow templates, in a logical language, that correspond to possibly incomplete specifications of workflow plans. Such plans can significantly reduce the search for a proof since a deductive planner only needs a proof for incomplete parts of the workflow plan. High Level Task (HTL) engineering using event log mining is based on finding frequent regular services invocation patterns from sequences corresponding to occurrence of the events as stored in an event log. We have recently developed frequent sequence pattern mining algorithms that will be tailored for this purpose.

(ii) **Enforcing Customizable Failure Semantics using Dynamic Reconfiguration Constraints**: We utilize AS$^3$ logic as a programming language for expressing knowledge about services behavior as well as domain specific reconfiguration constraints that must be enforced during adaptation from run-time failures.

(iii) **Dynamic Proof Theory**: We are developing a dynamic proof theory of AS$^3$ logic in order to efficiently and correctly enforce dynamic reconfiguration constraints on adaptive workflow templates. The dynamic proof theory will be based on a static proof theory of AS$^3$ logic that we have already developed.

3b) Hierarchical SAW agents for adaptable service coordination. Hierarchical SAW agents are automatically synthesized from AS$^3$ logic specifications of hierarchical SAW requirements using the proof system of AS$^3$ logic. Hierarchical SAW agents perform the following tasks: (i) Distributed context acquisition and processing by invoking the services that provide and process contexts. (ii) Distributed situation analysis. Based on our model for hierarchical SAW, situations are organized in a hierarchical structure that reflects the composition relations among situations. Hence, distributed SAW agents are also organized hierarchically. Situations analyzed by low-level SAW agents are used by high-level agents to compose more abstract situations. This not only helps us obtain a view of current situation at different abstract levels, but also facilitates situation information sharing among SAW agents. (iii) Autonomous service coordination by invoking certain services based on recognized situations.

3c) Agent-based distributed trust management for efficiently and effectively managing and enforcing security policies. Our agent-based distributed trust management technique is to synthesize security agents and interceptors from the security policy specifications using the proof system of AS$^3$ logic, and deploy the security agents and interceptors on a secure agent platform, such as Secure Infrastructure for Networked Systems (SINS) developed at the US Naval Research Laboratory (NRL) [Bha03], to enforce security policies. The interceptors will intercept the service requests and invoke the corresponding security agents to perform trust decision evaluation and enforce the trust decisions. SINS is currently used in our prototype system. It is based on distributed agent technology and a synchronous programming language, called Secure Operations Language (SOL) also developed at NRL [Bha02].

3d) Distributed workflow scheduling technique for efficiently scheduling, deploying and executing workflow with dependencies on situations and security policies.

## 5. Discussion and Future Work

For hierarchical SAW, our future research will be on analyzing the expressiveness of our declarative model, and proving the soundness and consistency of the model representations. For DTM, our future research will be on analysis of our trust management model, such as expressiveness, consistency and soundness. For adaptive workflow synthesis and workflow scheduling, the following innovative features will be incorporated into our dynamic proof theory to reason with incomplete workflow template execution and adaptation:

1) Online Planning – Online Planning refers to the idea of postponement of composition of detailed workflow structure until the time to execute the portion of the workflow that has not been composed yet, i.e. the workflow initially is abstractly defined and is fabricated on demand. This feature helps us recover failures due to the change in the external environment.
2) Online Resource Management and Scheduling – Failure sometimes can be caused due to unreliable situation information or changing resource requirements. To accommodate such dynamic information during the execution of multiple existing workflow instances, we will develop an approach to transforming original workflow specifications into new workflow specifications that incorporate all resource management and timing constraints into the control flow graph itself such that a constraint solver can be used at run-time to allocate and reallocate resources as the resource requirements and situation information changes at run-time, without necessarily modifying the control-flow strategies. This approach consists of (a) a formal model for timing and resource constraints, (b) distributed schedulers in AS$^3$ calculus, (c) transformation rules for generating distributed schedulers, and (d) a workflow scheduling service on SINS platform.

## Acknowledgment

This work is supported by the DoD/ONR under the Multidisciplinary Research Program of the University Research Initiative, Contract No. N00014-04-1-0723. The authors appreciate many useful discussions with Ramesh Bharadwaj of Naval Research Laboratory.

## References

[Abd97] T. F. Abdelzaher, E. M. Atkins, and K. G. Shin, "QoS Negotiation in Real-Time Systems and its Application to Automatic Flight Control," *IEEE Real-Time Technology and Applications Symposium*, June 1997.

[Abd00] A. Abdul-Rahman and S. Hailes, "Supporting Trust in Virtual Communities," *Proc. the 33rd Hawaii International Conference on System Sciences*, 2000, pp. 6007-6015.

[Abe04] D. Aberdeen, S. Thiébaux, and L. Zhang, "Decision-Theoretic Military Operations Planning," *ICAPS-04*, 2004.

[Ark98] R.C. Arkin, "Behavior-based Robotics", MIT Press, 1998.

[Bac00] F. Bacchus and F. Kabanza, "Using Temporal Logics to Express Search Control Knowledge for Planning," *Artificial Intelligence*, vol. 116(1-2), 2000, pp. 123-191.

[Bac01] F. Bacchus and M. Ady, "Planning with Resources and Concurrency: A Forward Chaining Approach," *Int'l Joint Conf. on Artificial Intelligence (IJCAI-2001)*, 2001, pp. 417-424.

[Bha02] R. Bharadwaj, "SOL: A Verifiable Synchronous Language for Reactive Systems," *Proc. Synchronous Languages, Applications, and Programming (SLAP' 02),* http://chacs.nrl.navy.mil/publications/ CHACS/2002/2002bharadwaj-entcs.pdf

[Bha03] R. Bharadwaj, "Secure Middleware for Situation-Aware Naval C$^2$ and combat Systems," *Proc. 9th Int'l Workshop on Future Trends of Distributed Computing System (FTDCS 2003)*, 2003, pp. 233-240.

[Bla96] M. Blaze, et al., "Decentralized Trust Management," *Proc. IEEE Symposium on Privacy and Security*, Oakland, 1996, pp. 164-173.

[Bla99] M. Blaze, et al., "The KeyNote Trust Management System (version 2)," *RFC2704*, 1999.

[Bla03] P. Blackburn, M. de Rijke and Y. Venema, "Modal Logic", Cambridge University Press, 2003.

[Bly93] Jim Blythe and W. Scott Reilly, "Integrating Reactive and Deliberative Planning in a Household Robot", Technical Report CMU-CS-93-155, Carnegie Mellon University, School of Computer Science, May 1993.

[Bro91a] Rodney Brooks, "Integrated systems based on behaviors", In Proceedings of AAAI Spring Symposium on Integrated Intelligent Architectures, Stanford University, March 1991. Available in SIGART Bulletin, Volume 2, Number 4, August 1991.

[Bro91b] Rodney Brooks, "Intelligence without reason", In Proc. of IJCAI-91. Morgan Kaufmann, San Mateo, 1991.

[Cac00] M. Caccamo, G. Buttazzo, and L. Sha, "Capacity Sharing for Overrun Control," *IEEE Real-Time Systems Symposium*, Orlando, FL, December 2000.

[Car83] S. Card, T. Moran and A. Newell, "The Psychology of Human-Computer Interaction," *Lawrence Erlbraum Associates*, 1983.

[Car00] L. Cardelli, A. D. Gordon, "Mobile ambients," *Theor. Comput. Sci.*, 240(1), 2000, pp. 177-213.

[Cha87] David Chapman. "Planning for conjunctive goals", Artificial Intelligence, 32:333–378, 1987.

[Cha03] A. T .S. Chan and Siu Nam Chuang, "MobiPADS: A Reflective Middleware for Context-Aware Computing," In *IEEE Transactions on Software Engineering*, vol. 29(12), Dec 2003, pp. 1072-1085.

[Che01] R. Chen and W. Yeager, "Poblano - a distributed trust model for peer-to-peer networks," *Technical report*, Sun Microsystems, 2001.

[Chu97] Y. Chu, et al., "REFEREE: Trust Management for Web Applications," *World Wide Web J.*, vol.2(3),1997,pp.127-139.

[Dav94] Davidsson, P., "Concepts and autonomous agents", LU--CS--TR: 94--124, Department of computer science, Lund University, 1994

[Dav98] H. Davulcu, M. Kifer, C.R. Ramakrishnan, and I.V. Ramakrishnan. "Logic based modeling and analysis of work flows," *Proc. ACM Symp. on Principles of Database Systems*, June 1998, pp. 25-33.

[Dav04] H. Davulcu, et al., "CTR-S: A Logic for Specifying Contracts in Semantic Web Services," *Proc. 13th Int'l WWW Conf.*, 2004, pp.144-153.

[Dey01] A. K. Dey and G. D. Abowd, "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications," *Human-Computer Interaction*, vol. 16(2-4), 2001, pp. 97-166.

[Dor97] J. E. Doran, et al, "On Cooperation in Multi-Agent Systems", The Knowledge Engineering Review, 12(3), 1997.

[Gar02] A.Garland and N. Lesh, "Continuous Plan Evaluation with Incomplete Action Descriptions," *Proc. 3rd Int'l NASA Workshop on Planning and Scheduling for Space*, Houston, 2002.

[Gav04] R. Gavriloaie, W. Nejdl, D. Olmedilla, K. Seamons, and M. Winslett, "No registration needed: How to use declarative policies and negotiation to access sensitive resources on the semantic web", In *1st First European Semantic Web Symposium*, 2004

[Hal90] J. Halpern and Y. Moses, "Knowledge and common knowledge in a distributed enviroment," *J. ACM*, vol. 37(3), 1990, pp. 549-587.

[Hal01] David L. Hall and James Llina, "Handbook of Multisensor Data Fusion," *CRC Press*, 2001.

[Her04] R. Hernandez, D. Vanderster and N. Dimopoulos, "Resource Management and Knapsack Formulations on the Grid," *Proc. 5th Int'l Workshop on Grid Computing*, November 2004, pp. 95-101.

[IBM04a] Business Process Execution Language for Web Services. http://www-128.ibm.com/developerworks/library/specification/ws-bpel/

[IBM04b] WS-Security. http://www-106.ibm.com/developerworks/webservices/library/ws-secure/

[IBM04c] WS Security Policy. http://www-106.ibm.com/developerworks/library/ws-secpol/
[Jim01] T. Jim, "SD3: A Trust Management SystemWith Certified Evaluation", In *IEEE Symposium on Security and Privacy*, May 2001.
[Kam97] S. Kambhampati, "Refinement Planning as a Unifying Framework for Plan Synthesis," *AI Magazine*, vol. 18(2), 1997, pp. 67-97.
[Kea04] K. Keahey, K. Doering, and I. Foster, "From Sandbox to Playground: Dynamic Virtual Environments in the Grid," *Proc. 5th Int'l Workshop on Grid Computing*, November 2004, pp. 34-42.
[Kro03] R. van der Krogt, M. de Weerdt, and C. Witteveen, "A resource based framework for planning and replanning," *Web Intelligence and Agent Systems*, vol. 1(3/4), 2003, pp. 173-186.
[Kro04] R. van der Krogt and M. de Weerdt, "The two faces of plan repair," *Proc. 16th Belgium-Netherlands Conf. on Artificial Intelligence (BNAIC-04)*, 2004, pp. 147-154.
[Li03] N. Li and J. Mitchell, "RT: A Role-Based Trust Management Framework," *Proc. 3rd DARPA Information Survivability Conf. and Exposition* (DISCEX '03), Apr. 2003.
[Liu73] C. L. Liu and J. W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real-time Environment," *J. ACM*, vol. 20(1), 1973, pp. 46-61.
[Liu03] Donghong Liu, "Knowledge-based planning systems and HTN planners," 2003.
[Lu01] C. Lu, "Feedback Control Real-time Scheduling", *Ph.D. thesis*, University of Virginia, May 2001.
[Lu02] C. Lu, J. A. Stankovic, G. Tao and S. H. Son, "Feedback Control Real-Time Scheduling: Framework, Modeling, and Algorithms," *Real-Time Systems Journal*, vol. 23(1/2), 2002, pp. 85-126.
[Mat03a] C. J. Matheus, et al., "A Core Ontology for Situation Awareness," *Proc. 6th Int'l Conf. on Information Fusion*, 2003, pp. 545-552.
[Mat03b] C. J. Matheus, et al., "Constructing RuleML-Based Domain Theories on top of OWL Ontologies," *Proc. 2nd Int'l Workshop on Rules and Rule Markup Languages for the Semantic We*b, 2003, pp. 81–94.
[McC69] J. McCarthy and P. J. Hayes, "Some Philosophical Problems from the Standpoint of Artificial Intelligence," *Machine Intelligence 4*, 1969, pp. 463-502.
[McC00] J. McCarthy., "Situation Calculus with Concurrent Events and Narrative", *http://wwwformal.stanford.edu/jmc/narrative/ narrative.html*, 2000.
[Nae03] Naedele M., "Standards for XML and Web services security," *Computer*, Vol.36, Iss.4, 2003, pp.96-98.
[Nak02] Nakamur Y., Hada S. and Neyama R., "Towards the integration of Web services security on enterprise environments," *Proc. of Symp. on Applications and the Internet (SAINT) Workshops (2002)*, 2002, pp.166-175
[Nam01] Brian Mac Namee, Pádraig Cunningham, "A Proposal for an Agent Architecture for Proactive Persistent Non Player Characters", *Proc. 12th Irish Conference on Artificial Intelligence and Cognitive* Science pp. 221-232, 2001.
[Nau99] D. Nau, Y. Cao, A. Lotem, and H. Muñoz-Avila, "SHOP: Simple Hierarchical Ordered Planner," *Proc. 16th Int'l Joint Conf. on Artificial Intelligence (IJCAI 99)*, 1999, pp. 968-975.
[Nau04] D. Nau, M. Ghallab, and P. Traverso, "Automated Planning: Theory and Practice," Morgan Kaufmann, 2004.
[New02] E. Newcomer, "Understanding Web Services," Addison Wesley, 2002.
[Nie03] Niederberger C., Gross M. , "Hierarchical and Heterogeneous Reactive Agents for Real-Time Applications", Computer Graphics Forum, September 2003, vol. 22, no. 3, pp. 323-331
[Nou97] Nourredine Bensaid and Philippe Mathieu, "A hybrid architecture for hierarchical agents", pages 91-95. Griffith University, Gold-Coast, Australia, February 1997.
[Nwa96] Hyacinth S. Nwana, "Software Agents: An Overview", *Knowledge Engineering Review*, Vol. 11, No 3, pp. 205-244, October/November 1996.
[OAS04a] "OASIS eXtensible Access Control Markup Language (XACML) TC," http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml.
[OAS04b] OASIS Security Services TC, "Security Assertion Markup Language (SAML)," http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security.
[Pin94] J. A. Pinto, "Temporal Reasoning in the Situation Calculus," *PhD Thesis*, University of Toronto, 1994.
[Pla03] D. Plaisted, "A Hierarchical Situation Calculus," *J. Computing Research Repository (CoRR)*, 2003.
[Ran03] A. Ranganathan and R. H. Campbell, "A Middleware for Context-Aware Agents in Ubiquitous Computing Environments," *Proc. ACM/IFIP/USENIX Int'l Middleware Conf.*, 2003, pp. 143-161.
[Rom02] M. Roman, C. Hess, R. Cerqueira, A. Ranganathan, R. Campbell and K. Nahrstedt, "A middleware infrastructure for active spaces," *IEEE Pervasive Computing*, vol. 1(4), 2002. pp. 74–83.
[Rus03] S. Russell and P. Norvig, "Artificial Intelligence: A modern Approach," 2nd ed., *Prentice Hall*, 2003.
[Sen02] P. Senkul, M. Kifer, and I. H. Toroslu, "A Logical Framework for Scheduling Workflows under Resource Allocation Constraints," *Proc. 28th Int'l Conf. on Very Large Data Bases (VLDB'02)*, 2002, pp. 694-705.
[Shm05] V. Shmatikov and C. Talcott, "Reputation-Based Trust Management," *J. Computer Security (Special issue on selected papers of WITS '03)*, vol. 13, no. 1, 2005, pp.167-190.
[Sri04] Biplav Srivastava and Jana Koehler. Planning with Workflows - An Emerging Paradigm for Web Service Composition. Workshop on Planning and Scheduling for Web and Grid Services held in conjunction with (ICAPS 2004), Canada, June 2004.
[Sta88] J. A. Stankovic and K. Ramamrithitham (Eds), "Hard Real-Time Systems," *IEEE Press*, 1988.
[Urd03] C. Urdiales, et al, "Hierarchical planning in a mobile robot for map learning and navigation", in *Autonomous Robotic Systems - Soft Computing and Hard Computing Methodologies and Applications*, D. Maravall, D. Ruan and C. Zhou (eds), Springer Verlag Pub pp. 165-188, 2003

[Vas04] Vasco Pires, Miguel Arroz, Luis Custódio, Logic Based Hybrid Decision System for a Multi-robot Team, *8th Conference on Intelligent Autonomous Systems*, 2004

[W3C] W3C, "Web Services Description Language (WSDL) 1.1," *http://www.w3.org/TR/wsdl*

[Wic04] A. Bose, B. Wickman, and C. Wood, "MARS: A Metascheduler for Distributed Resources in Campus Grids," *Proc. 5th Int'l Workshop on Grid Computing*, November 2004, pp. 110-118.

[Woo02] Mike Wooldridge, "An Introduction to Multiagent Systems by Michael Wooldridge", ISBN 0 47149691X, John Wiley & Sons (Chichester, England), February 2002

[Xio04] L. Xiong and L. Liu, "PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities," *IEEE Trans. Knowl. Data Eng*. Vol. 16 No.7, 2004, pp.843-857.

[Yan03] L. Yang, J. M. Schopf and I. Foster, "Conservative Scheduling: Using Predicted Variance to Improve Scheduling Decisions in Dynamic Environments," *Proc. Supercomputing 2003*, November 2003.

[Yau02a] S. S. Yau, et al., "Development of Situation-Aware Application Software for Ubiquitous Computing Environments," *Proc. 26th Ann. Int'l Computer Software and Applications Conf. (COMPSAC 2002)*, 2002, pp. 233-238.

[Yau02b] S. S. Yau, et al., "Reconfigurable Context-Sensitive Middleware for Pervasive Computing," *IEEE Pervasive Computing*, vol. 1(3), 2002, pp. 33-40.

[Yau05a] S. S. Yau, et al., "Situation-Aware Access Control for Service-Oriented Autonomous Decentralized Systems," *7th Int'l Symp. on Autonomous Decentralized Systems*, 2005, pp.17-24.

[Yau05b] S. S. Yau, et al., "An Adaptable Security Framework for Service-based Systems," *10th IEEE Int'l Workshop on Object-oriented Real-time Dependable Systems*, pp.28-35.

[Yau05c] S. S. Yau, D. Huang, H. Gong and H. Davulcu, "Situation-awareness for Adaptable Service Coordination in Service-based Systems," *Proc. 29th Annual Int'l Computer Software and Applications Conference (COMPSAC 2005)*, 2005, pp. 107-112.

[Yu02] B. Yu and M. P. Singh, "An evidential model of distributed reputation management," *1st Int'l Joint Conference on Autonomous Agents and MultiAgent Systems*, 2002.

[Yu04] J. Yu and R. Buyya, "A Novel Architecture for Realizing GridWorkflow using Tuple Spaces," *Proc. 5th Int'l Workshop on Grid Computing*, November 2004, pp. 119-128.

[Zha87] W. Zhao, K. Ramamritham and J. A. Stankovic, "Preemptive Scheduling Under Time and Resource Constraints," *IEEE Transactions on Computers*, vol. 36(8), 1987.