

FARM: A FEEDBACK-BASED ADAPTIVE RESOURCE MANAGEMENT FOR COMPLEX REAL-TIME SYSTEMS AND APPLICATION TO SENSOR WEB*

S. SWAMINATHAN AND G. MANIMARAN †

Abstract.

The increasing reliance on real-time technologies for building complex systems calls for efficient dynamic resource management strategies. These strategies aim to achieve high resource utilization and allow graceful degradation in the face of unpredictable workload. In this paper, we present a novel resource management methodology, called Feedback-based Adaptive Resource Management (FARM), which is suited for complex real-time applications that has dynamic workload. We present a case study of one such complex system: Autonomous Hot Spot Convergence System (AHSCS) for sensor web, and apply our resource management solution to AHSCS. The FARM methodology combines the advantages of feedback-controlled scheduling, value-based scheduling, and path-based paradigm to provide a predictable, reliable, and robust services to complex real-time systems, such as AHSCS.

Key words. adaptive resource management, feedback-based scheduling, value-based scheduling, path-based paradigm.

1. Introduction. The advancements in computing technology coupled with growing needs for building complex real-time systems such as sensor web [1, 2], autonomous vehicle controller, and missile control systems. A common characteristic of these systems is that the workload is dynamic and unpredictable, hence any static resource management scheme will lead to pessimistic usage of resources. Thus a resource management scheme for these complex real-time systems needs to exhibit following characteristics: (i) *Graceful Degradation* - the system needs to gracefully degrade during overloads by executing critical tasks, (ii) *Adaptation* - the system needs to adapt to change in workload, enabling better resource usage than static resource management schemes and (iii) *Robustness* - system needs to maintain stability amidst uncertainties in workload. In this paper, we present a new feedback-based adaptive resource management (FARM) methodology that has all the above desired characteristics and discuss its application to a complex real-time system: Autonomous Hot Spot Convergence System (AHSCS) using sensor web. We would like to point out that this methodology was developed while designing a solution for the problem of resource management in AHSCS. Hence in this paper, we present a brief overview of the system, discuss the characteristics of this complex real-time system. Then, we describe the shortcomings of the existing resource management approaches if directly applied to build this system and present our resource management methodology solution for designing AHSCS. We believe that FARM can be applied to many other complex real-time systems involving dynamic and unpredictable workload to obtain graceful degradation and robust stability.

1.1. Sensor Web - Background. NASA seeks to transition from large, instrument-jammed observatories to cheaper and lighter space-based platforms. Thus, the Earth Science Vision Initiative has been formulated. The cornerstone of the Vision is the *sensor web*, [1, 2, 3] a closely integrated constellation of earth observing satellites that collectively monitor the conditions of our planet through a vast array of instruments.

* This research was supported in part by the NSF under grant CCR-0098354.

† Dependable Computing & Networking Laboratory, Dept. of Electrical and Computer Engineering, Iowa State University, Ames, IA 50011, Email: {*swamis, gman*}@iastate.edu .

This network of satellites can act autonomously in controlling instruments and spacecraft, while also responding immediately to (1) the commands of the user interested in measuring specific events and (2) important terrestrial events (e.g., a volcanic event). This approach allows easy deployment of new technology and moreover is scalable.

What is Sensor Web?

The concept of integrating a constellation of Earth observing satellites into a cohesive network of measurement instruments is called the *Sensor Web*. In concept it is similar to the Internet in that scientists and other users will have access to any on-orbit sensors and be able to direct and control those sensors in the same manner as the information is accessed in today's Internet.

The Sensor Web concept also will take full advantage of the revolution that is taking place in information and telecommunications technologies for direct delivery of space-based Earth observations to the end-user at the cost of placing a long distance telephone call.

Application Scenario

The following scenario illustrates the operation of a constellation of earth observing (EO) satellites. Several sentinel satellites provide a line-of-sight view of all instrumented satellites in the constellation; each sentinel knows the precise location of all members in the constellation. The scenario involves the handling of a significant terrestrial event. A synthetic aperture radar satellite detects a volcanic event. The autonomous satellite brings the event into focus by rotating its instruments and altering its coverage area; on-board feature detectors analyze the data and assign priorities to different parts of the image; data compression is employed; to communicate the data to the ground station, another member of the constellation must be used as a relay, since the ground station is out of view; more important portions of data are sent back first, at high resolution; less important parts are sent back last; scientists are alerted and assume control of the spacecraft; they direct its instruments for specific follow-up measurements.

Technology Challenges

Providing a technology to handle a scenario, as above, requires research and development in information technology, application domain, and inter-disciplinary areas. Thus, the technology challenges posed by the sensor web are as follows:

1. Information Technology

- (a) Resource management mechanisms are needed to manage the various instruments, computing, and communication resources in such a way that efficient (i) monitoring of the activities inside the "spot beam" (coverage area), (ii) analyzing of the perceived scientific events (e.g., hot-spots), and (iii) decision making and communicating information about these events to the end-user and Earth control center, are achieved. Specifically, the resource manager is required to address dependability (real-time, reliability/availability, and security) requirements.
- (b) Algorithms and techniques for on-board data processing. This includes data aggregation, data compression, and signal and image processing.
- (c) Reconfigurable architectures and algorithms for various on-board data processing.
- (d) Energy-efficient architectures and algorithms for on-board computing and communication.
- (e) System modelling, verification, and validation techniques and tools.

2. Satellite Communication Technology

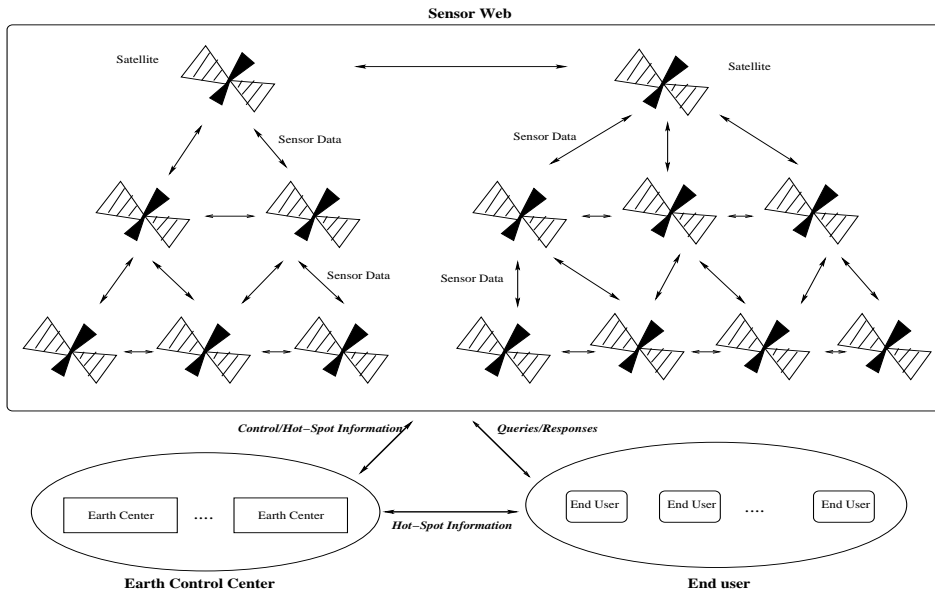


FIG. 1. *Sensor Web for AHSC System*

- (a) Dynamic control and reconfiguration of satellite instruments.
- (b) Satellite computing, communication, and instrumentation technologies.
- (c) Global real-time on-board navigation capability for earth science remote sensing.

3. Domain-specific Technology

- (a) Domain-specific algorithms and techniques. For example, algorithms for water-level monitoring in Polar Ice caps and for detecting cloud contamination with atmospheric correction.
- (b) Modelling and analysis of applications.

System Model

Figure 1 depicts the sensor web system environment wherein the satellites, Earth control center, and the end user are identified. The following assumptions describe the configuration and capabilities of these entities.

1. Satellite Capabilities

- (a) The system (i.e., sensor web) is a constellation of Earth observing satellites that coordinate among themselves for distributed monitoring, processing, and decision making.
- (b) The satellites orbit around the earth at different orbits: low earth orbit (small satellites), geostationary (geostationary satellites) orbit, and L1 and L2 orbits (sentinel satellites). Since the spot beam of a satellite depends on its orbit level, the accuracy of sensor data also depends on it. Smaller the spot beam, higher the accuracy of the sensor data.
- (c) Each satellite is equipped with a set of instruments and sensors for measurement.
- (d) Each satellite is equipped with on-board data processing capabilities that enable it to act autonomously, reacting to significant measurement

events on and above the Earth, increasing precision and coverage where needed without human intervention.

- (e) Each satellite is equipped with on-board communication capabilities for communicating with other satellites, Earth control center, or the end user. Inter-satellite communication involves exchanging of hot-spot information (sensor data), current utilization of on-board computing capability, spot beams, and the current instrument configuration.
2. **Earth control center** is assumed to have necessary computing facilities for off-board processing and archiving, and communication facilities to communicate with the sensor web.
3. **End user workstation** is assumed to have the necessary computing and communication facilities for querying the sensor web.

In this paper, we specifically focus on the resource management issue of the AHSCS, which involves efficient management of computing and communication resources such that the real-time, reliability, availability, and security requirements of the sensor web are satisfied. We use our novel resource management methodology that is capable of maintaining system stability and robustness in spite of dynamic and uncertain workload. A shorten version of the proposed methodology can be found at [4].

The rest of the paper is organized as follows: In section 2, we present AHSCS's analysis and design and discuss its workload characteristics, motivate the need for for new resource management methodology in section 3. We present our new resource management methodology, FARM, in section 4 and conclude in section 5.

2. AHSCS: System Analysis and Design. The sensor web for the Autonomous Hotspot Convergence System (AHSCS) can be viewed as a distributed real-time system with different computational capabilities collaborating to solve a problem (problem of analyzing a hot-spot together). Figure 2 shows the high level schematic of the AHSCS at each node (satellite), wherein four key interfaces to the system and relevant databases are identified.

- *Sensor Interface*: An interface for interacting with the sensor instruments used to collect the data about hot-spots and for reconfiguring the sensors.
- *Earth Control Center Interface* : An interface for communicating with Earth Control Center(ECC) and for decoding the control signals sent by the Earth Control Center.
- *End-User Interface* : An interface for interacting with end-user (EU) to understand the queries and to return the query results.
- *Interface for Inter-satellite Communication* : An interface for communicating with other satellites in the network.
- *Shared Tables*: The various functional modules (identified in Section 2.1) of AHSCS interact through a set of shared tables.

2.1. Functional Modules and Tables. The AHSCS can be represented in the form of modules and the functional level view (data and control flow) of the AHSCS is shown in Figure 3. The various functional modules are as follows: (1) Sensor Reading Module [SRM], (2) Human Interface Module [HIM] (interface to earth station and end-user), (3) Hot-Spot Identifier [HSI], (4) Hot-Spot Analyzer [HSA], (5) Overload Handler [OH],(6) Communication Modules [GC-I], [GC-II] (communicating to other satellites for exchanging sensor data, and for communicating load, topology, and coverage information to other satellites), (7) Resource Reconfigurator [RR] (to dynamically reconfigure the resources such as sensor instruments) and (8) Query Processor [QP] (to perform end-user queries).

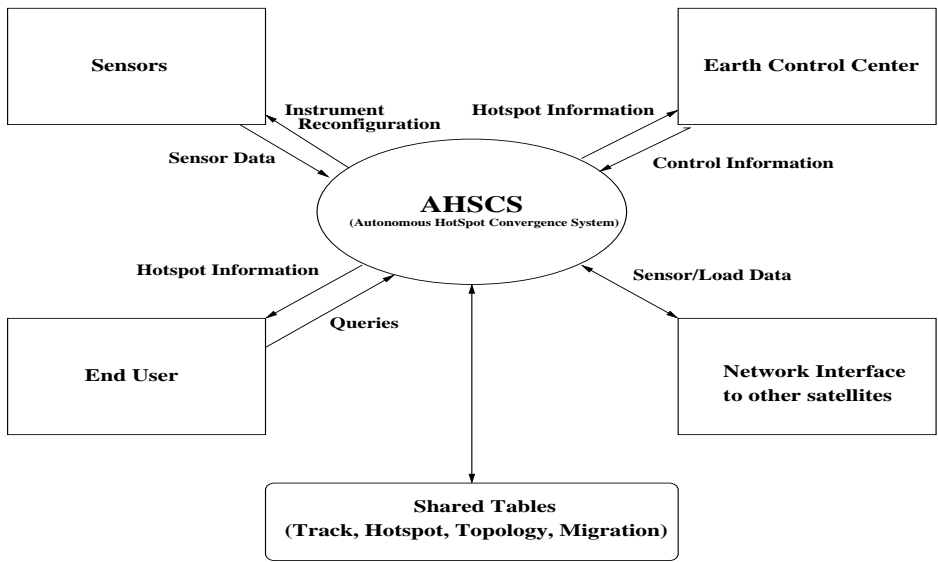


FIG. 2. A high-level schematic of AHSC system

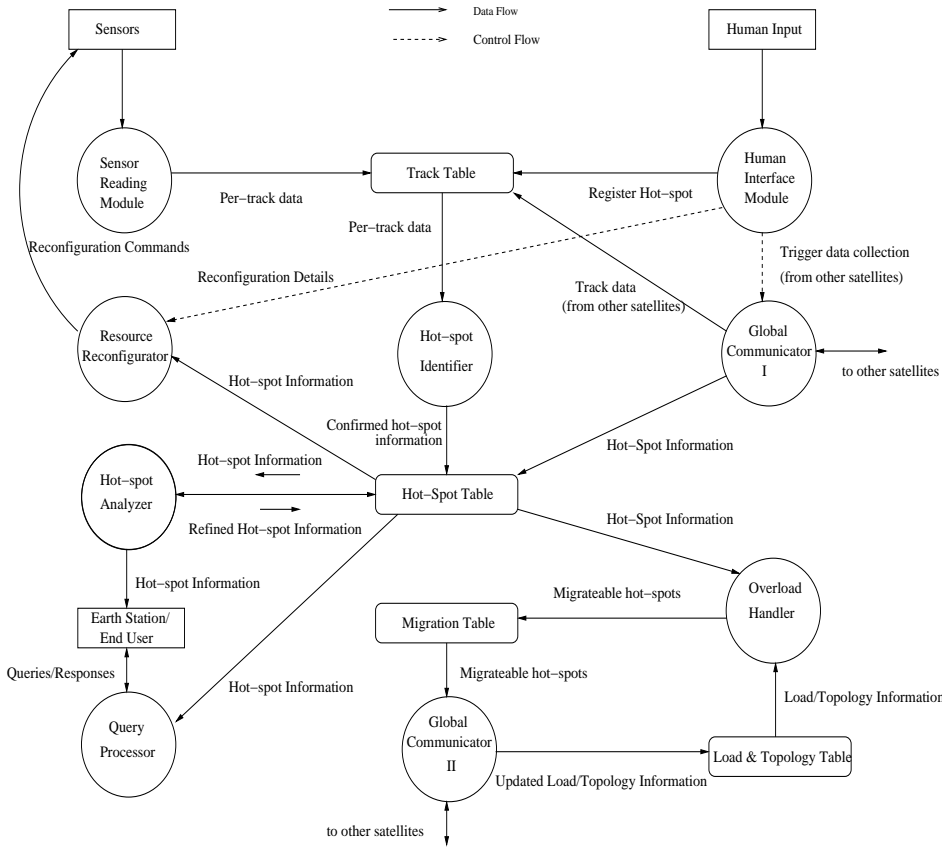


FIG. 3. Functional Diagram of AHSC System

The list of shared tables used by the various modules of AHSCS is as follows: (1) Track Table (for storing track data), (2) Hotspot Table (for storing data about identified hotspots), (3) Load and topology Table (for storing the system load data, and satellite topology related data), and (4) Migration Table (for storing details about migratable hotspots).

The details of the functional modules and shared tables are presented in Appendix.

3. Workload Model and Motivations. In this section, we first discuss the system model describing the system capabilities and workload, then provide the motivation for developing a new resource management methodology.

3.1. Workload Characteristics.

1. The computing workload for satellite on-board processing typically comes from various sources: Sensor data processing (*periodic workload*), hot-spot analysis (*aperiodic workload*), controls from Earth control center (aperiodic), and end user queries (aperiodic).
2. The typical communication workload composed of: inter-satellite communication (both periodic and aperiodic), satellite to Earth control center (both periodic and aperiodic), and satellite to end user communication (aperiodic).
3. The system must handle dynamically varying workloads in response to perceived scientific events (e.g., hot-spots), the spacecraft environment, spacecraft anomalies and user commands.
4. The dynamic workload also depends on the nature of the observed hot-spot. They could be:
 - *Dynamically arriving periodic workload* to monitor “fixed hot-spots” (e.g., volcano).
 - *Aperiodic workload* to monitor “moving hot-spots” (e.g., tornado) which requires continuous monitoring with possible hand-off to other satellite and reconfiguration of instrumentation and computing resources.
5. In summary, there is a great deal of dynamics and uncertainty in the computing and communication workloads.

Since all situations and possible uses of sensors cannot be anticipated during design phases, an approach for dynamically adapting the allocation of distributed computational and communication resources is needed. Even if it were possible to anticipate all possible use case scenarios of future sensor webs, dedicating computational and communication resources based on maximum requirements will be far more expensive than employing an adaptive resource management system.

3.2. Motivations for New Resource Management Methodology. We first justify the suitability of path-based paradigm and feedback control based paradigm for resource management in AHSCS, then we motivate the need for our new resource management methodology that combines these paradigms.

3.2.1. Motivation for Path Based Paradigm. The dynamic path paradigm [6, 7] is convenient for specifying end-to-end system objectives and for analyzing timeliness, dependability, and scalability of distributed real-time systems. The notion of a dynamic path is also useful to describe systems with dynamic variability, wherein the behavioral characteristics of the various tasks cannot be determined statically.

A dynamic path is an abstraction of larger granularity when compared with the traditional task level abstraction. The AHSCS is highly suitable for applying the dynamic path paradigm for the following reasons:

- The AHSCS system is a highly dynamic real-time system, whose load can vary significantly over time with no upper bound. Therefore a system description that can accommodate this dynamic variability is required and any static resource allocation scheme (based on an artificially imposed upper bound) will lead to poor resource utilization.
- Most of the system specifications directly translate into timeliness constraints (deadlines) only for higher level execution paths. Therefore a task-based description of the system will require imposing artificial deadlines on individual tasks whereas the problem specification only imposes a timeliness constraint on the execution of a sequence of tasks.
- Finally, many of the situations in the AHSCS are such that most of the tasks or functions do not have pre-specified deadlines. In such cases, the dynamic path paradigm is an appropriate mechanism for describing the system specifications.
- Paths being an abstraction of larger granularity than tasks, the number of scheduling entities to be handled by the scheduling algorithm is smaller in the case of a path-based scheduler. This results in lower scheduling cost.

3.2.2. Motivation for Feedback Control Based Scheduling. It is clear that the workload in AHSCS is highly dynamic and unpredictable. Most of the scheduling paradigms that have been proposed [8] – static priority-driven preemptive scheduling (e.g., RMS and EDF), static table-driven scheduling (e.g., MARUTI), dynamic planning based scheduling (e.g., Spring and DeSiDeRaTA [9]), and dynamic best-effort scheduling – are “open-loop” strategies that are effective when the workload can be accurately modeled. However, these paradigms are inadequate for many real world problems, including the AHSCS, wherein the workload cannot be accurately modelled, and these paradigms do not provide graceful degradation under overloads. Thus, there is a need for efficient methodology for resource management where predictable performance guarantees can be obtained in the presence of uncertainty.

Feedback control theory [10] has been central to modelling systems operating in uncertain environments. In the past few decades, this theory has made impressive strides in this direction. Correct adaptation as illustrated by feedback control theory will yield significant dividends with respect to robustness. In recent years, many researchers have adopted feedback control theory for scheduling in real-time systems [11, 12, 13, 14]. These work assume task level abstraction for scheduling which does not necessarily capture the semantics of the application.

3.2.3. Motivation for the New Feedback Control Path Based Paradigm.

In this paper, we develop a novel resource management methodology and solution for the AHSCS that combines feedback control based scheduling and path-based based scheduling (we call it, *Feedback-based Adaptive Resource Management (FARM)*) in order to provide robust performance in the presence of uncertainty in workload and also to provide graceful degradation during overloads. The proposed FARM methodology is a powerful paradigm that combines the benefits of the component strategies: scheduling based on application semantics (through path-based scheduling) and providing robustness (through feedback strategy). We would like to point out here, to the best of our knowledge, ours is the first work that combines path-based paradigm and feedback control strategies for resource management.

A resource management middleware developed on a project called SWARM [9], was proposed for complex applications such as sensor web, which is based on path-based paradigm. Our FARM methodology is more powerful than SWARM in provid-

ing robustness and graceful degradation, in addition to being flexible for expressing timing constraints.

4. Proposed Feedback-based Adaptive Resource Management. In this section, we first provide an overview on dynamic paths, then identify the schedulable entities (i.e., the paths) and their characteristics, and finally propose our novel resource management methodology as a resource management solution for the AHSCS.

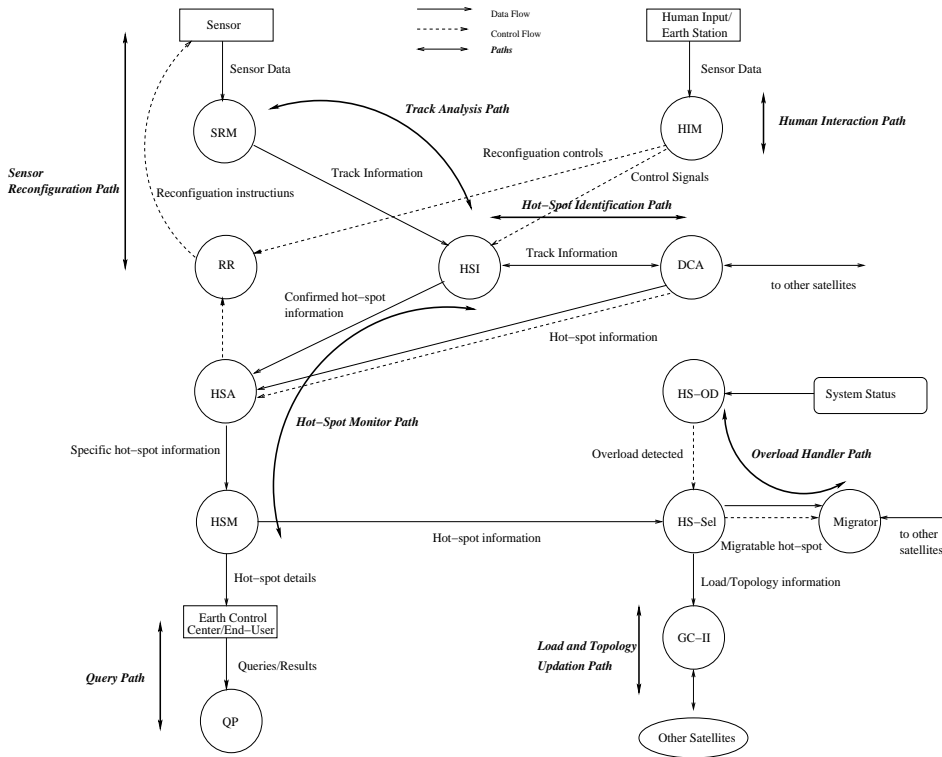


FIG. 4. Data and Control Flow Diagram of AHSC System (Path level)

4.1. Background on Dynamic Paths. As defined in [6, 7], a dynamic path consists of a data/event source, a data/event stream and a data/event consumer. The data/event source produces data/events, which cause the data/event consumer to perform processing. Based on the nature of the source and consumer, dynamic paths can be classified into the following categories:

- **Transient Path (TP):** This consists of an event source, an event stream, and an event consumer. A TP is activated by an event, which causes the consumer to initiate an action. The response to the event must occur within a specified amount of time, which is the *required latency* or *activation deadline* of this path. Usually, timeliness is very critical for TP's and this deadline is hard.
- **Continuous Path (CP):** This consists of a data source, a data stream, and a data consumer. The data stream is a set of data elements with attributes. The data source cyclically produces updates about the attributes of the elements in the data stream which are processed by the data consumer. The set

of elements in the data stream can change with time. Such a CP can be characterized by its *cycle deadline* which is the time by which all the elements in the data stream must be processed once. *Tactical throughput* (number of elements processed per unit time) and *Data Interprocessing Time* (time interval between successive updates to a data element) are measures that characterize the timeliness of the CP.

- **Quasi-Continuous Path (QCP):** This is activated and deactivated by events. However, between the activation and deactivation events, a QCP behaves like a CP, cyclically processing the items in a data stream. Each QCP has a *deactivation deadline* which is the time by which it must be deactivated. This deadline can usually be determined upon activation.

4.2. Dynamic Paths in AHSCS. The task level diagram of the AHSCS given in Figure 4 identifies the dynamic paths in the system. The dynamic path characteristics such as its type, data source (DSrc), streams (DS) and consumers (DC) in case of Continuous and Quasi-continuous paths and triggering events (TE) and ending events (EE) in case of transient paths, are given in Table 1.

Path No.	Path Name	Path Type	Path Characteristics
1	Track Analysis	Continuous	DSrc - Sensor DS - Table Entries DC - HSI task
2	Hot-Spot Identification	Quasi-Continuous	DSrc - Track Table DS - Table Entries DC - HSA task
3	Hot-Spot Monitor	Quasi-Continuous	DSrc - Hot-Spot Table DS - Table Entries DC - ECC/EU
4	Query Processor	Transient	TE - Queries from ECC/EU EE - Results returned to ECC/EU
5	Sensor Reconfiguration	Transient	TE - Reconfiguration info. from HSA or HIM EE - Reconfiguration of resources
6	Load and Topology Updation	Continuous	DSrc - LTM task DS - Load and Topology Table DC - HS-Sel task
7	Overload Handler Path	Transient	TE - Overload detection EE - Hot-Spot Monitor Migration
8	Earth Center Interface Path	Transient	TE - ECC EE - Trigger HSI and/or RR

TABLE 1
Paths Characteristics in AHSCS

4.3. Our FARM Methodology. In this section, we present our FARM methodology for resource management in complex real-time systems, such as AHSCS. The

FARM methodology combines feedback control based scheduling, path based paradigm, and value based scheduling [15, 16, 17]. Figure 5 shows the schematic of the proposed methodology wherein the control variables, namely, the manipulated variable, controlled variable, set point, and the actuation mechanisms are identified. Table 2 summarizes the parameters used for each of these control variables.

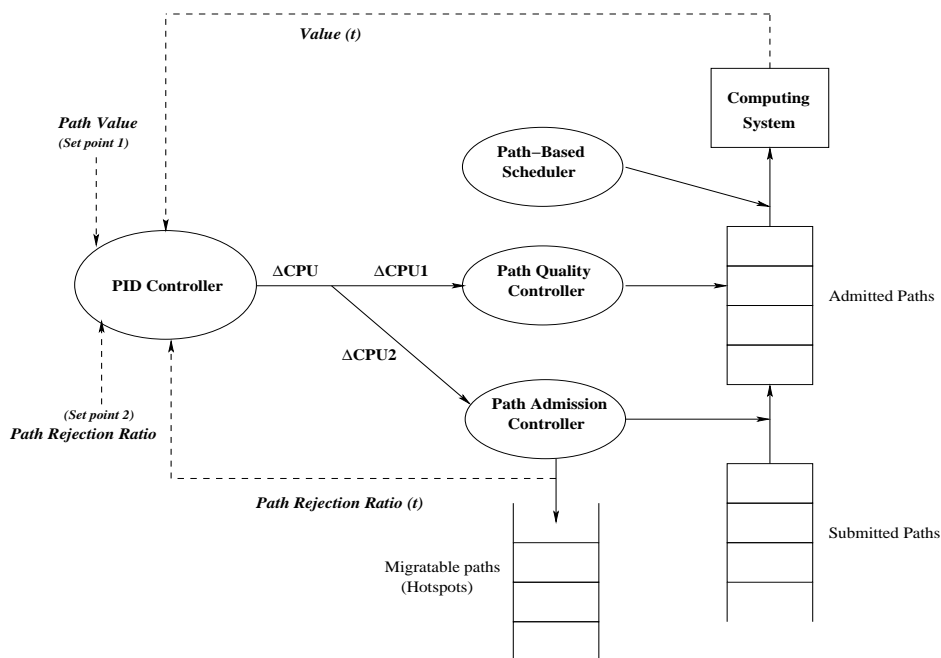


FIG. 5. Schematic of FARM methodology

Variable name	AHSCS parameter
Controlled variable	Number of hotspots rejected Number of end-user queries rejected Average <i>value</i> of hotspots
Set point	Maximum number of hotspots rejectable Maximum number of end-user queries rejectable Minimum acceptable value of hotspots
Manipulated variable	Resource (CPU) utilization
Actuation mechanism	Path admission controller Path quality controller

TABLE 2
Feedback control variables in AHSCS

The workload to the resource manager is the execution of various types of paths of which the continuous paths form the static workload, and the transient or quasi-continuous paths form the dynamic (variable) workload. For instance, the dynamic workload is generated when a new hotspot is identified and it needs to be processed, or when a query is generated from an end-user which needs to be responded.

It is assumed that the static workload is scheduled onto the computing resources offline, and the dynamic workload is scheduled online. The typical functioning of the FARM resource manager, to handle dynamic workload, is as follows:

- The dynamic workload arrives at the *submitted path queue*.
- The *path admission controller* admits some paths into *accepted hotspot queue*; and rejects others and puts them into *migratable hotspot queues*, which will potentially be migrated to other suitable satellites.
The admission decision is an actuation mechanism which is dictated by the action of the PID controller.
- The *path quality controller* adjusts the quality of the path processing, i.e., increases/decreases the computation times of the paths relating to hotspot processing. For example, the computation time of the hotspot monitor path can be adjusted without violating the minimum computational quality. However, it is to be noted that the paths relating to query processing and overload handling may not be amenable for quality adaptation. The quality adaptation decision is also an actuation mechanism which is dictated by the action of the PID controller.
- The *dynamic path based scheduler* schedules the accepted paths onto the computing resources. Path-based scheduling algorithms such as the one proposed in [6] can be used for this purpose.
- The paths are executed by the computing system.
- The controlled variables are observed and periodically fed back to the *PID controller* [10]. The number of hotspots/queries rejected is fed back by the path admission controller, and the controller obtains the average hotspot value by periodically reading the hot-spot table. It must be noted that the hot-spot table is updated by hot-spot analyzer.
- The PID controller compares the monitored value of controlled variables against their corresponding set points and computes the “error”. The error is used to determine the amount of change that needs to be brought in the value of manipulated variable. For example, the change may be increasing/decreasing the current CPU utilization by some amount, say ΔCPU . This ΔCPU is being split into $\Delta CPU1$ and $\Delta CPU2$, and these splits in CPU utilization are effectuated by the path quality controller and path admission controller, respectively.

The proposed resource management scheme aims to maintain high system value and high schedulability by aiming to maintain the path value and path rejection ratio to the desired setpoint values. For example, when the system is overloaded, the system is bound to experience high path rejection ratio. The controller which monitors (and responsible for controlling) this parameter will instruct the path quality controller to reduce the quality of computation, such that CPU utilization gain is $\Delta CPU1$. The path quality controller sets all tasks in lower quality levels (that consumes less computation time), and hence will decrease path rejection ratio, thus allowing graceful degradation of the system. In the other scenario, if the path value decreases less than the setpoint, then the system instructs the path admission controller to reject few tasks to gain $\Delta CPU2$, thereby allowing critical tasks to execute in an acceptable quality. Thus, the controller periodically monitors the path rejection ratio and system value, and determines the action that needs to be taken, if any. The issues of determining how to determine $\Delta CPU1$ and $\Delta CPU2$ from ΔCPU , the sampling/feedback rate should be fixed by the system designer, based on the required response time and

sensitivity to the workload.

4.4. Satellite Coordination and Load Balancing. The coverage areas under the control of adjacent satellites in the same level or across levels may have regions of overlap. In such cases, tracks/hotspots that occur in this region will be detected by both satellites. Coordination involves ensuring that the two satellites arrive at a consistent decision about which one of them is to take further control of the common tracks/hotspots. We will first define the *load index* at any given satellite as the number of active tracks/hotspots being handled by that satellite. The following actions take place during coordination:

- A satellite at a given level transmits its load index and proximity to the track/hotspot to three other types of satellites:
 - Satellites that are in the same level.
 - Satellites that are in the lower levels which fall under the spot beam of the given satellite.
 - Satellites that are in the higher levels whose spot beam covers the given satellite.
- The selection of a satellite to monitor a common track/hotspot is done based on the following criteria:
 - *Quality based coordination:* If the hotspot needs fine-grain (i.e., accurate) analysis, the satellite that is at the lowest level is chosen among the eligible satellites. If the selected satellite is overloaded, then the next lower level satellite is tried.
 - *Load based coordination:* If a satellite (sender) is overloaded, it initiates a load (hotspot processing) migration to a lower/higher/same level satellite (receiver) that is lightly loaded. The selection should also consider the quality of the hotspot processing at the remote satellite, if it were to be migrated.

In both the above cases, if there are no potential receiver satellite, the sender could initiate a “reconfiguration request” to an underloaded satellite requesting it to reconfigure its instrumentation to cover the required hotspots, and then the migration is achieved.

It is evident from the above discussion that efficient coordination and load balancing, involves several decision making which are collectively known as global scheduling in the distributed real-time scheduling literature [18]. A typical global scheduling is achieved by the following four inter-related policies:

- *Information Policy:* This dictates how information exchange among different nodes should be carried out, i.e., what information is to be collected, and when it is to be collected.
- *Transfer Policy:* This decides when loads are to be migrated from a given node.
- *Selection Policy:* This decides which loads are to be migrated from a heavily loaded node.
- *Location Policy:* This governs the decision about choosing a suitable receiver for the load migration.

Table 3 summarizes our proposed approach for each of the global scheduling policies to achieve efficient coordination and load balancing. The details of each of these policies can be found in [18].

5. Conclusions. In this paper, we presented a new feedback-based adaptive resource management methodology (FARM) and presented its application to Au-

Policy	Approach suggested
Information policy	Periodic policy
Transfer policy	Threshold policy (Path admission controller)
Selection policy	Value-based policy (migrate least valued hotspots)
Location policy	Sender initiated

TABLE 3
Global scheduling policies for AHSCS

onomous Hot-Spot Convergence System (AHSCS) that uses sensor web. FARM combines the advantages of feedback control scheduling (to handle uncertainty in workload), path-based scheduling paradigm (to flexibly express timing constraints), and value based scheduling (to allow graceful degradation). We studied AHSCS system and its workload characteristics, and applied FARM-based resource management solution to it. We believe that FARM can be suitable for many contemporary/future complex real-time systems.

REFERENCES

- [1] J.L. Paul, "Smart Sensor Web: Web-based exploitation of sensor fusion for visualization of the tactical battlefield," *IEEE Aerospace and Electronics Systems Magazine*, vol.16, no.5, pp.29-36, May 2001.
- [2] J.L. Paul, "Smart Sensor Web: Web-based exploitation of sensor fusion for visualization of the tactical battlefield," In Proc. *Intl. Conference on Information Fusion*, 2000.
- [3] K.A. Delin, and S.P Jackson, "Sensor Web for in situ exploration of gaseous biosignatures," In Proc. *IEEE Aerospace Conference*, vol.7, pp.465-472, 2000.
- [4] S. Swaminathan and G. Manimaran, "FARM: A Feedback-based Adaptive Resource Management for Autonomous Hot-Spot Convergence System," In Proc. *Intl. Workshop on Parallel and Distributed Real-Time Systems*, Florida, April 2002 (a preliminary version of this paper).
- [5] "Exploring Our Home Planet Earth Science Enterprise Strategic Plan," NASA, (<http://www.earth.nasa.gov/visions/stratplan>).
- [6] L.R. Welch, "Large-grain, dynamic control system architectures," In Proc. *Workshop on Parallel and Distributed Real-Time Systems*, 1997.
- [7] L.R. Welch, B. Ravindran, B.A. Shirazi, and C. Bruggeman, "Specification and modeling of dynamic, distributed real-time systems," *Real-Time Systems Symposium*, pp.72-81, 1998.
- [8] K. Ramamritham and J. A. Stankovic, "Scheduling algorithms and operating systems support for real-time systems," *Proc. IEEE*, vol.82, no.1, pp.55-67, Jan. 1994.
- [9] R. Detter, L.R. Welch, B. Pfarr, B. Tjaden, and E.-N. Huh, "Adaptive management of computing and network resources for spacecraft systems," In Proc. *Annual Military and Aerospace Applications of Programmable Devices and Technologies Conference (MAPLD)*, 2000.
- [10] S.P. Boyd and C.H. Barratt, *Linear Controller Design: Limits of Performance*, Prentice Hall, Englewood Cliffs, New Jersey, 1991.
- [11] J.A. Stankovic, C. Lu, S.H. Son, and G. Tao, "The case for feedback control real-time scheduling," in *Proc. EuroMicro Conf. on Real-Time Systems*, 1999.
- [12] C. Lu, J.A. Stankovic, G. Tao, and S.H. Son, "Design and evaluation of feedback control EDF scheduling algorithm," in *Proc. Real-Time Systems Symp.*, 1999.
- [13] L.R. Welch, D.A. Lawrence, and D.R. Alexander, "Feedback control resource management using a Posteriori workload characterizations," In Proc. *IEEE Conference on Decision and Control*, 2000.
- [14] R. Omari, G. Manimaran, M.V. Salapaka, and A.K. Somani, "New algorithms for open-loop and closed-loop scheduling of real-time tasks based on execution time estimation," Submitted to *IEEE Trans. Computers*, 2001.
- [15] A. Burns, D. Prasad, A. Bondavalli, F. Di Giandomenico, K. Ramamritham, J. Stankovic, and L. Stringini, "The meaning and role of value in scheduling flexible real-time systems,"

Journal of Systems Architecture, 2000.

- [16] L.R. Welch and S. Brandt, "The value of benefit in real-time computing," In Proc. *Intl. Workshop on Parallel and Distributed Real-Time Systems*, 2001.
- [17] S. Swaminathan and G. Manimaran, "A new algorithm for value based scheduling in dynamic real-time systems," In Proc. *Intl. Workshop on Parallel and Distributed Real-Time Systems*, Florida, 2002.
- [18] C. Siva Ram Murthy and G. Manimaran, *Resource Management in Real-time Systems and Networks*, MIT Press, 2001.
- [19] F. Wang, K. Ramamritham, and J.A. Stankovic, "Determining redundancy levels for fault-tolerant real-time systems," *IEEE Trans. Computers*, vol.44, no.2, pp.292-301, Feb. 1995.
- [20] B. Tjaden, L.R. Welch, S. Ostermann, D. Chelberg, R. Balupari, M. Bykova, A. Mitchell, and L. Tong, "SECURE-RM: Security and Resource Management for Dynamic Real-Time Systems," In Proc. *Real-time System Security Minisymposium, Southern Conference on Computing (SCC)*, 2000.

Appendix: Functional Modules and Tables.

Functional Modules.

1. **Sensor Reading Module:** The sensor data collected by the sensors are read by this module and is stored in the *track table*. This module thus consists of only one task, SRM, which reads the data from the sensor and stores it in the track table.
2. **Human Interface Module:** This module is responsible for collecting information from Earth Control Center, deciphering the control signals sent by the Earth Control Center and communicating with the control center. Thus, the tasks in this module are the HIM (Human Interface Module), responsible for reading and deciphering the control signals sent by the Earth Control Center. This task, based on the control signals sent by them, it updates the track table for triggering further action such as identifying a hot-spot etc. Further, this monitor can also trigger the resource reconfigurator module when reconfiguration control commands are sent by the Earth Control Center. Further, this module triggers the Global Communicator I module, which triggers the other satellites when a scientist in the Earth Control Center becomes aware of an hot-spot without knowing its exact location.
3. **Hot-Spot Identifier:** This module consists of a task HSI, which is responsible for analyzing the sensor data read by the sensor reading module (by reading from the track table) and identify if there are any hot-spots (by correlating the current observed sensor data with previously observed sensor data). If a new hot-spot is detected based on the analysis of the track information (obtained from its own sensors and also from other satellites through Global Communication I) , then it creates a new hot-spot information entry in the *hot-spot table*.
4. **Hot-Spot Analyzer:** This module consists of HSA task for analyzing a particular hot-spot and finding a suitable node for monitoring the hot-spot. If the current node is capable of monitoring the hot-spot, then HSA triggers HSM task, which is created for each hot-spot monitored in the system, and is responsible for monitoring the information stored in the hot-spot table for its hot-spot, observing the quality of the information perceived by the satellite and communicates it to Earth Control Center or the end-user. HSM can trigger Overload Handler module, if it observes that the quality of information perceived about a hot-spot is less than acceptable quality and needs to be migrated to some other node which can monitor this to provide a better quality. This situation is possible, if the observed hot-spot is dynamic in nature and needs constant migration of hot-spot observer task for efficient monitoring. Further, this module can also trigger the resource reconfigurator module, if the instruments need to be reconfigured dynamically to obtain more value for the collected information (such as to improve the focus or resolution of the perceived image).
5. **Overload Handler:** This module consists of tasks that are responsible for monitoring and maintaining the computational load of a system and the quality of

the data perceived by the monitors analyzing the hot-spots. So, if the system is either overloaded or if the quality of an observed hot-spot information is not of acceptable quality, then HS-OD (Hot-Spot Overload Detector) task, which identifies this scenario from reading the hot-spot table, triggers HS-Sel task. The HS-Sel is responsible for selection of hot-spot to be migrated and also for selection of a suitable computing node that can monitor the hot-spot to be migrated with acceptable quality. HS-Sel selects a hot-spot to be migrated based on the value of the observed hot-spot data, by reading from the hot-spot table. The selection of the satellite to be migrated is done by consulting the *Load and Topology migration table*, to check if the selected node has the necessary computational capabilities and coverage capabilities to observe the hot-spot to be migrated. Thus, the task selects the hot-spot to be migrated to a selected node and writes these information to *Migration table*, along with hot-spot information collected till then. The exact details of overload handling and maintaining the quality of hotspot processing is explained in Section 4.

6. **Resource Reconfigurator:** This module consists of RR task, which is responsible for reconfiguring the instruments to change the spot beam (coverage area), based on the control signals received by the HIM or HSM. This module enables dynamic reconfiguration of sensor instruments based on the requirement of the system.
7. **Query Processor:** This module is responsible for interaction with Earth Control Center or the end-user. This involves communication with end-user, process the queries and to reply the results (by reading the information from the hot-spot table).
8. **Global Communicator I:** This module consists of DCA task responsible for communicating and collecting the sensor information from other satellites regarding hot-spots and tracks (possible hot-spots). The observed information is written to hot-spot table and track table. The information written by these tasks in the hot-spot table is used by HSM to refine the hot-spot information stored in the hot-spot table.
9. **Global Communicator II:** This module consists of GC-II task, which is responsible for updating the *Load and Topology Table*, with load and spot beam information of various satellites in the sensor web. Further, this module also consists of a migrator task, which reads the hot-spots to be migrated to other satellites from *Migration table* and transfers the particular hot-spot information to the selected satellite.

Shared Database Tables. The following are the list of shared tables used by the system for storing and exchanging data between modules:

1. **Track Table:** This table is used for writing the track (possible hot-spot) information read by the sensor reading module and has the following fields: Trigger time, Track ID, Source, Track Results, Track Longitude, Track Latitude, Track type, end-user location, track information, satellite id.
2. **Hot-Spot Table:** This table is used for storing the hot-spot information written by the hot-spot identifier and analyzer modules. The fields in this table can be: Hot-Spot ID, Hot-Spot type, its longitude, its latitude, source, consumer location, hot-spot information, value (quality of the perceived hot-spot information).
3. **Load and Topology Table:** This table is used for storing the load and coverage information of various satellites in the sensor web. The fields in this table are: Satellite ID, Latitude range, Longitude range, computational load index.
4. **Migration Table:** This table is used for storing the details of the hot-spot to be migrated to some other satellite either for balancing the workload or for obtaining better quality perception. The fields in this table are: Hot-Spot ID, Destination Satellite ID, Hot-Spot Information.