

FARM: A Feedback-based Adaptive Resource Management for Autonomous Hot-Spot Convergence System

S. Swaminathan and G. Manimaran
Dependable Computing & Networking Laboratory
Dept. of Electrical and Computer Engineering
Iowa State University, Ames, IA 50011, USA
{swamis, gmani}@iastate.edu

Abstract— In this paper, we present a novel and comprehensive resource management solution for the autonomous hot-spot convergence system (AHSCS) that uses sensor web. This solution is in response to a call for solution at the WPDRTS 2002. The proposed solution involves system analysis and design and developing a new resource management methodology, which we call Feedback-based Adaptive Resource Management (FARM). The FARM methodology combines the advantages of feedback control scheduling, path-based scheduling, value-based scheduling, and survivability strategies to provide dependable (predictable, reliable, and secure) services to the AHSCS.

I. PROBLEM STATEMENT

Processing, archiving, accessing, visualizing and communicating Earth Science data from space and ground-based sensors to the research community presents many challenging problems in distributed, real-time computing. The problem seeking solutions that will be presented at the special session of the 2002 Workshop on Parallel and Distributed Real-Time Systems (WPDRTS), is taken from this domain. The solutions to the problem statement were expected should be in the form of a design, a prototype or an architecture similar to the ones [1], [2] presented in the previous years. This year's problem statement in short is as follows [3]:

According to NASA's Earth Strategic Plan [4], "Advances in Earth observing and information technologies, research, and modeling are all required to fulfill our long term vision for Earth System prediction." The observing system of the future will include satellites in a variety of orbits. These will include a *sensor web* of small smart satellites in low Earth orbit, large aperture sensors in geostationary orbits, and sentinel satellites at L1 and L2 orbits. The problem to be solved is based on this sensor web and is defined as follows: "A scientist on Earth becomes aware of a "hot-spot" on the earth -an erupting volcano, a forming hurricane, etc. There are many research satellites in orbit - some will be able to observe this hot-spot, some will not. Determining which satellites can observe this hot-spot, notifying each of the event, and modifying each observing schedule is a difficult, time consuming job. It may take so long that the hot-spot has disappeared. The scientist desires to notify just one satellite in orbit, whichever comes over the nearest ground station first. Then the on-orbit satellites perform *collaborative problem solving*, working together autonomously to perform the notification, selection and rescheduling necessary to observe this hot-spot with as many resources as possible, and send all the data to the scientist. The on-board real-time systems of the future should enable collaborative problem solving."

II. BACKGROUND AND SOLUTION OVERVIEW

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*, [5], [6], [7] a closely integrated constellation of earth observing satellites that collectively monitor the conditions of our planet through a vast array of instruments. 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.

A. 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.

B. 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.

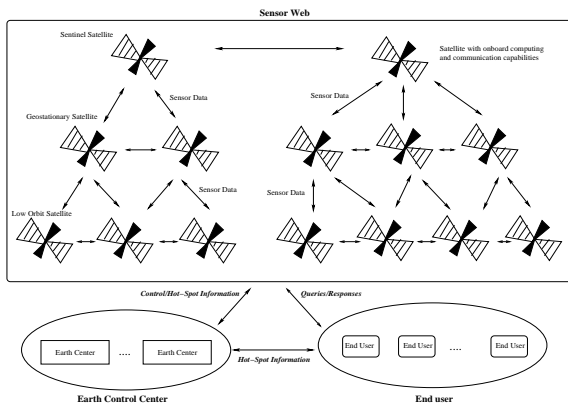


Fig. 1. Sensor Web for AHSC System

C. 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

- 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.
- Algorithms and techniques for on-board data processing. This includes data aggregation, data compression, and signal and image processing.
- Reconfigurable architectures and algorithms for various on-board data processing.
- Energy-efficient architectures and algorithms for on-board computing and communication.
- System modeling, verification, and validation techniques and tools.

2. Satellite Communication Technology

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

3. Domain-specific Technology

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

D. 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

- The system (i.e., sensor web) is a constellation of Earth observing satellites that coordinate among themselves for distributed monitoring, processing, and decision making.
 - 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.
 - Each satellite is equipped with a set of instruments and sensors for measurement.
 - 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.
 - 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.
- 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.
 - 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. To accomplish this, it is necessary to allow dynamic configuration of sensor webs of space-based resources and to insure that data flows smoothly from the satellite-based instrument to the ground-based archive.

The rest of the paper is organized as follows: Section III presents system analysis and design, Section IV characterizes the workload and motivates the need for new resource management methodology, Section V proposes a new resource management methodology (FARM), Section VI discusses the survivability issues, and finally Section VII concludes the paper.

III. 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

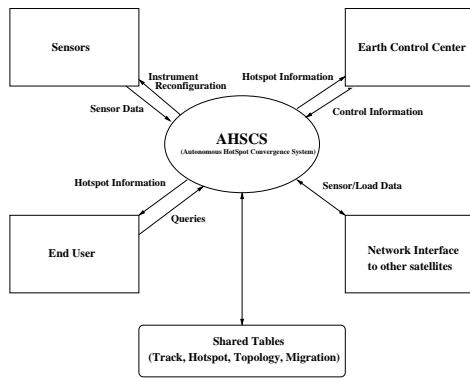


Fig. 2. A high-level schematic of AHSCS system

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 and for decoding the control signals sent by the Earth Control Center.
- *End-User Interface*: An interface for interacting with end-user 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 3.1) of AHSCS interact through a set of shared tables.

A. 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).

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).

Table I shows the list of database tables with their corresponding reader and writer modules.

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

IV. RESOURCE MANAGEMENT – 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.

A. 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.

B. 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 developing a new resource management methodology combining them.

B.1 Motivation for Path Based Paradigm

The dynamic path paradigm [8], [9] 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:

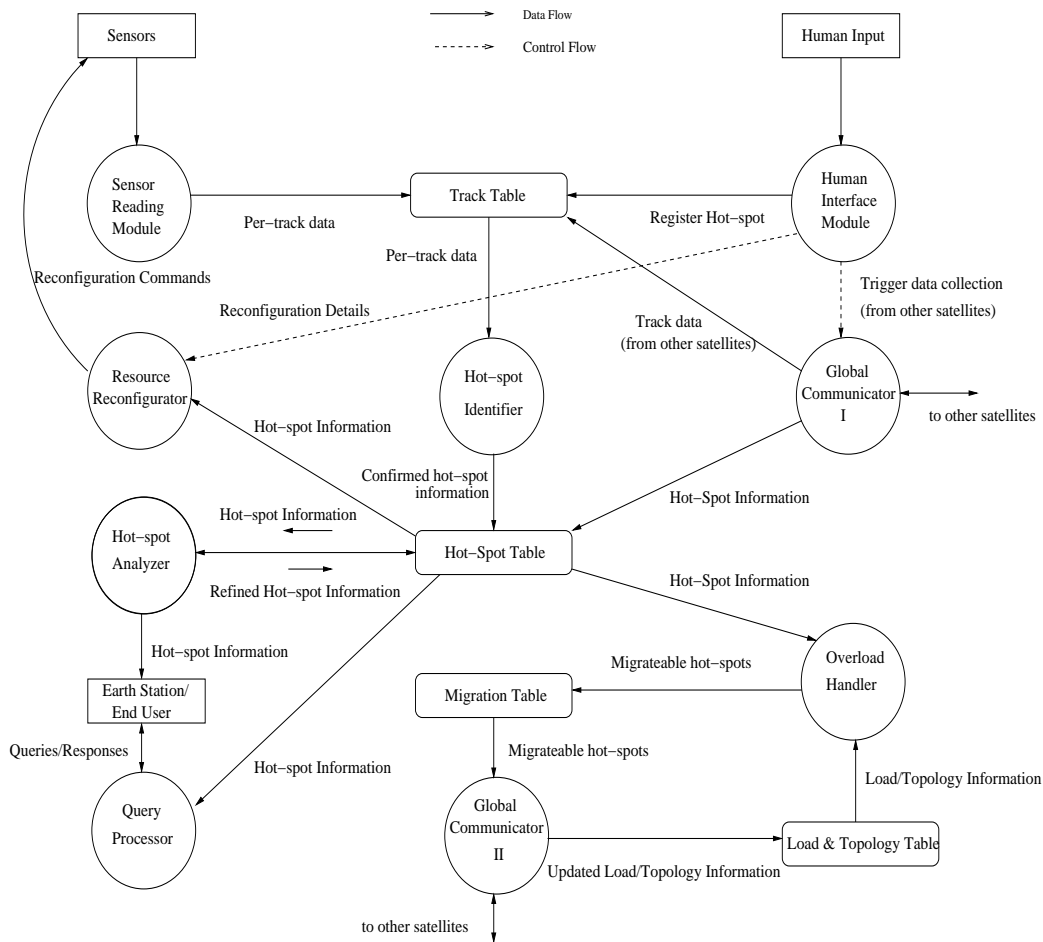


Fig. 3. Functional Diagram of AHSC System

Table No.	Table Name	Readers	Writers
1	Track Table	HSI	SRM, HIM, HSI
2	Hot-Spot Table	HSA, HSI, GC-I, QP	HSI, HSA, GC-I
3	Load and Topology Table	OH	GC-II
4	Migration Table	GC-II	OH

TABLE I
DATABASE TABLES IN AHSCS

- 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.

B.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 [10] – 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 [11]), 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 modeled, 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 [12] has been central to modeling 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 [13], [14], [15], [16]. These works assume task level abstraction for scheduling which does not necessarily capture the semantics of the application.

B.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, called SWARM [11], 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 providing robustness and graceful degradation, in addition to being flexible for expressing timing constraints.

V. 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 a novel resource management methodology with solution for the AHSCS.

A. Background on Dynamic Paths

As defined in [8], [9], 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.

B. 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, streams and consumers are given in Table II.

C. 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 [17], [18], [19]. 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 III summarizes the parameters used for each of these control variables.

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

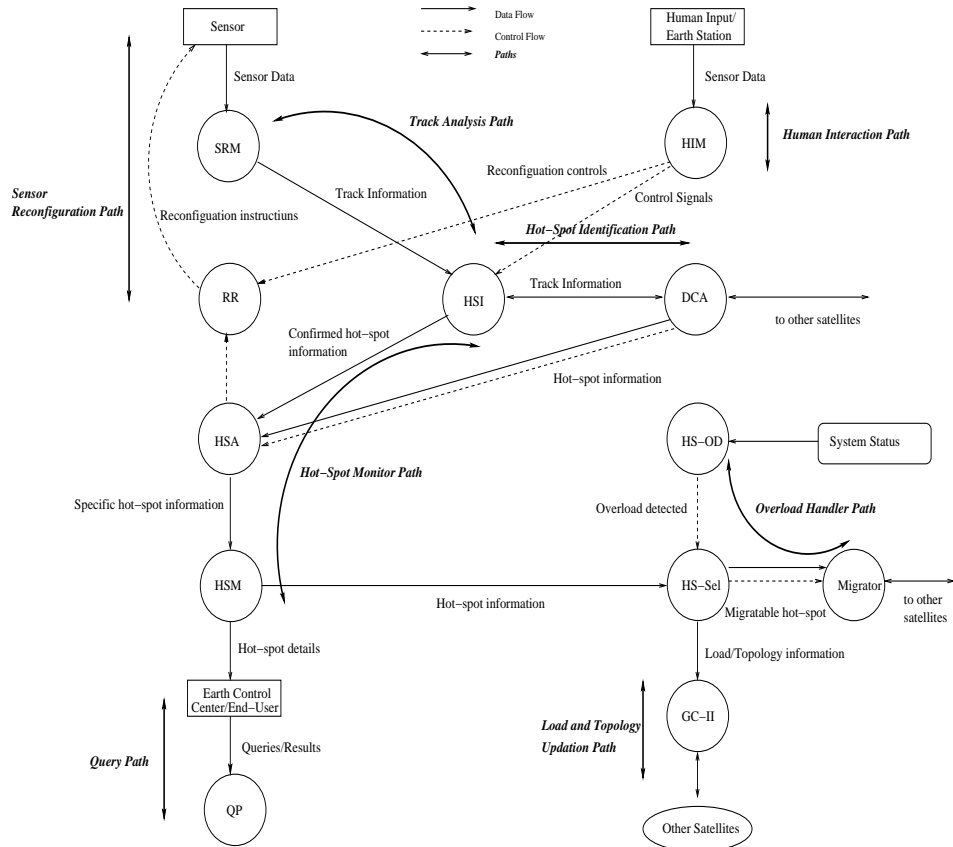


Fig. 4. Data and Control Flow Diagram of AHSC System (Path level)

Path No.	Path Name	Path Type	Path Characteristics
1	Track Analysis	Continuous	Data Source - Sensor Data Streams - Table Entries Data Consumer - HSI task
2	Hot-Spot Identification	Quasi-Continuous	Data Source - Track Table Data Streams - Table Entries Data Consumer - HSA task
3	Hot-Spot Monitor	Quasi-Continuous	Data Source - Hot-Spot Table Data Streams - Table Entries Data Consumer - Earth Center/End User
4	Query Processor	Transient	Triggering event - Queries from Earth Control Center/End User Ending event - Results returned to Earth Control Center/End User
5	Sensor Reconfiguration	Transient	Triggering event - Reconfiguration information from HSA or HIM task Ending Event - Reconfiguration of resources
6	Load and Topology Update	Continuous	Data Source - LTM task Data Streams - Load and Topology Table Data Consumer - HS-Sel task
7	Overload Handler Path	Transient	Triggering Event - Overload detection
8	Earth Center Interface Path	Transient	Ending Event - Hot-Spot Monitor Migration Triggering Event - Earth center control Ending Event - Trigger HSI and/or RR

TABLE II
PATHS CHARACTERISTICS IN AHSCS

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 III
FEEDBACK CONTROL VARIABLES IN AHSCS

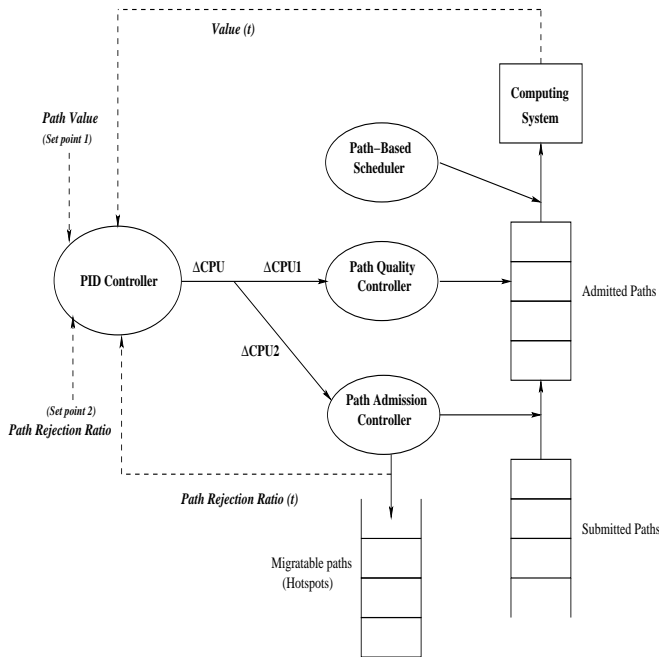


Fig. 5. Schematic of FARM methodology

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 [8] 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* [12]. 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.

D. 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-

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 IV
GLOBAL SCHEDULING POLICIES FOR AHSCS

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 [20]. A typical global scheduling has four inter-related policies: Information, Transfer, Selection and Location policies.

Table IV 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 [20].

VI. SURVIVABILITY

Survivability of computing and communication infrastructures is very critical to the dependable operation of the AHSCS. The potential threats could be from (1) faults in hardware or software, (2) malicious attacks from individual hackers, criminals and terrorists or warring nations, (3) human error, and (4) technology and service design deficiencies, scaling deficiencies associated with the congestion of systems and services, protocol inefficiencies, and the like. The dependable information infrastructure must deliver secure, reliable and predictable services in the face such threats. Since there cannot be a single technique that can protect the system against all the possible threats, a combination of techniques needs to be employed. A combination protection and restoration strategies need to be employed to tolerate both known and unknown threats. The protection strategy requires redundancy in spatial, temporal, and quality dimensions; and the restoration strategy requires efficient mechanisms for fault detection and location, and system reconfiguration [20].

To achieve dependable AHSCS, the following subsystems

Fault-tolerant Technique	Path Names
N-Version Programming	Load and Topology Updation
Recovery Block	Sensor Reconfiguration Overload Handler Query Processor Earth Center Interface
Imprecise Computation	Track Analysis Hot-spot Identifier Hot-spot Monitor

TABLE V
FAULT-TOLERANCE TECHNIQUES FOR PATHS IN AHSCS

need to be designed dependable:

- *Infrastructure protection:* Hardware redundancy techniques (such as TMR) need to be used to tolerate hardware faults, firewall technology has to be used to protect against communication-induced threats.
- *Secure and fault-tolerant communication:* Inter-satellite communication needs to be encrypted and authenticated to protect against malicious activities that include insertion, modification, and fabrication of messages. The same thing is true with the communication between sensor web and earth control center/end-user, and between end user and earth control center. Redundant communication channels need to be employed between all the critical communication entities to tolerate channel failures.
- *Fault-tolerant path execution:* Critical paths need to be identified and are to be scheduled with multiple versions to provide fault-tolerance. Depending upon the nature of the path, the appropriate fault-tolerant technique needs to be employed. Fault-tolerant techniques include N-Version Programming (NVP), Recovery Block (RB), and Imprecise Computation (IC). Paths that are amenable for acceptance test can use RB technique; paths that involve iterative computation, such as tracking, can use IC technique to allow graceful degradation in result quality in the presence of overload; and other critical paths can use NVP technique. Table V summarizes suggested fault-tolerance techniques for each path in the AHSCS.

A. Value-based Performability Index and Scheduling

As NVP and PB require multiple versions to achieve reliable execution of paths, the schedulability of the system may be degraded. Thus, there exists a tradeoff between schedulability and reliability. This tradeoff is often captured by defining a *performability* measure [20], [21] that captures both schedulability and reliability in a single value function which represents the overall utility of the system. The performability measure is usually defined for individual tasks and it needs to be extended to paths.

We propose a path-based performability measures wherein a value function is defined for each path to capture the value of the path to the system. Since AHSCS has many transient and Quasi-continuous paths, dynamic value-based fault-tolerant scheduling (such as the one proposed in [19]) needs to be developed.

The value of a path is characterized by: *reward*, if the path executes successfully; *penalty*, if an admitted path fails to complete before its deadline (due to overload or faults) or if a path is rejected by the admission controller. The path-based fault-tolerant scheduler determines the best redundancy level (i.e., the number of versions) for each path such that the overall performability is maximized.

B. Security

Security in AHSCS becomes an important issue as the satellites can be accessed by users over the Internet. This global web technology makes the sensor web vulnerable to ever changing array of attacks for which current defense mechanisms are inadequate. Hence, we suggest using a security scheme similar to the one proposed in [22], which is an integrated secure framework that combines an intrusion detection system and resource management framework. The suggested security framework will have the following two components:

- *Network Intrusion Detection System*: This can perform both intrusion and anomaly detection. The system records its current network access information and user access patterns, compares it with history access profile and analyzes the profile information to detect misuse or anomaly.
- *Resource Manager*: If an intrusion or anomaly is detected by the system, then the resource manager (Path quality controller in our system) does the appropriate resource reallocation such as dropping a suspicious path (e.g., user query path), or possibly reducing the amount of resources allocated to it if suspicion decreases.

Thus, a security framework with the above two components can enable the detection of intrusion or attacks on the sensor web and can plan appropriate resource management actions to build a secure real-time applications over the sensor web.

VII. CONCLUSIONS

In this paper, we presented a novel and comprehensive resource management solution for the autonomous hot-spot convergence system (AHSCS) that uses sensor web. The solution is arrived by realizing the following subgoals: (1) defining the system model and workload characteristics, (2) system analysis and design of the AHSCS, (3) resource management using new methodology, called feedback-based adaptive resource management (FARM), that combines the advantages of feedback control scheduling (to handle uncertainty in workload), path-based scheduling paradigm (to flexibly express application's timing constraints), and value based scheduling (to allow graceful degradation), and (4) survivability and security strategies. The authors believe that the proposed solution provides new avenues for further research and new insights into the design and development of AHSCS.

REFERENCES

[1] A. Manikuttu, M. Shashidhar, G. Manimaran, and C. Siva Ram Murthy, "DREAD: Distributed REal-time Air Defense System," In Proc. *Intl. Workshop on Parallel and Distributed Real-time Systems*, pp.108-118, Apr. 1997.

[2] R. Sriram, G. Manimaran, and C. Siva Ram Murthy, "Air Traffic Control System," In Proc. *Intl. Workshop on Parallel and Distributed Real-time Systems*, 1998.

[3] Problem Statement for the 10th Workshop on Parallel and Distributed Real-Time Systems – Autonomous Hot-Spot Convergence, 2002. (<http://comp.uark.edu/aapon/wpdrts2002/problem2002.pdf>)

[4] "Exploring Our Home Planet Earth Science Enterprise Strategic Plan," NASA, (<http://www.earth.nasa.gov/visions/stratplan>).

[5] 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.

[6] 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.

[7] 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.

[8] L.R. Welch, "Large-grain, dynamic control system architectures," In Proc. *Workshop on Parallel and Distributed Real-Time Systems*, 1997.

[9] 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.

[10] 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.

[11] 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.

[12] S.P. Boyd and C.H. Barratt, *Linear Controller Design: Limits of Performance*, Prentice Hall, Englewood Cliffs, New Jersey, 1991.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] S. Swaminathan and G. Manimaran, "A reliability-aware value-based scheduler for dynamic multiprocessor real-time systems," accepted for publication at *Intl. Workshop on Parallel and Distributed Real-Time Systems*, 2002.

[20] C. Siva Ram Murthy and G. Manimaran, *Resource Management in Real-time Systems and Networks*, MIT Press, 2001.

[21] 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.

[22] 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. 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 or Resource Reconfigurator module in case of overload or when the need for resource/instrument reconfiguration arises.
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 identifies this scenario by reading the hot-spot table and triggers HS-Sel task which selects the hot-spot to be migrated and suitable computing node that can monitor the hot-spot to be migrated with acceptable quality, by consulting the *Load and Migration Table* to ensure if it has enough capabilities to monitor the hot-spot and writes to *Migration Table*. *Migration Table*.
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 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 migrated to some other satellite either for balancing the workload or for obtained better quality perception. The fields in this table are: Hot-Spot ID, Destination Satellite ID, Hot-Spot Information.