NETWORKED INFOMECHANICAL SYSTEMS: A MOBILE EMBEDDED NETWORKED SENSOR PLATFORM

Richard Pon[1,2], Maxim A. Batalin[1,3], Jason Gordon[1,2], Aman Kansal[1,2], Duo Liu[1,2],
Mohammad Rahimi[1,3], Lisa Shirachi[1,2], Yan Yu[1,4], Mark Hansen[1,5], William J. Kaiser[1,2],
Mani Srivastava[1,2], Gaurav Sukhatme[1,3], Deborah Estrin[1,4]
[1]Center for Embedded Networked Sensing, University of California, Los Angeles, CA 90095
[2]Electrical Engineering Department, University of California, Los Angeles, CA 90095
[3]Computer Science Department, University of Southern California, Los Angeles, CA
[4]Computer Science Department, University of California, Los Angeles, CA 90095
[5]Statistics Department, University of California, Los Angeles, CA 90095

***Abstract*— Networked Infomechanical Systems (NIMS) introduces a new actuation capability for embedded networked sensing. By exploiting a constrained actuation method based on rapidly deployable infrastructure, NIMS suspends a network of wireless mobile and fixed sensor nodes in three-dimensional space. This permits run-time adaptation with variable sensing location, perspective, and even sensor type. Discoveries in NIMS environmental investigations have raised requirements for 1) new embedded platforms integrating many diverse sensors with actuators, and 2) advances for in-network sensor data processing. This is addressed with a new and generally applicable processor-preprocessor architecture described in this paper. Also this paper describes the successful integration of R, a powerful statistical computing environment, into the embedded NIMS node platform.**

Keywords- Embedded; Networked; Sensor; Actuation; System; Mobility

I. INTRODUCTION

Advances in embedded networked sensor systems (ENS) have enabled the first deployments of these devices in many environments [1,2]. Applications for ENS devices now appear in public health, security, and environmental monitoring [3]. However, the first deployment of ENS devices reveal new challenges, to be discussed below, associated with operation of static sensor networks and raise requirements for new capabilities [4]. These include precise sampling of dynamic phenomena, deployment of diverse sensor types in three-dimensional environments, long term and constantly available operation, as well as on-demand high performance computing and communication to remote users.

First, the physical configuration including the distribution of sensing elements for a static ENS network is determined at deployment time. While the ENS network may be optimized for sensing fidelity based upon the initial state of the environment, the inevitable and unpredictable time evolution of environmental phenomena may introduce obstacles to sensing, introduce sources of distortion or interference, or cause the spatial distribution of events to depart from the design-time distribution. In particular, the spatiotemporal sampling rate required to reconstruct an environmental model with a required level of fidelity may evolve with time according to rapidly changing of phenomena. Thus, a specified design-time distribution of ENS devices may not provide the required sampling rate. For ENS systems that must detect events, the obstacles present in environments may evolve and obscure events again rendering a design-time solution for deployment to be suboptimal or inapplicable at run-time.

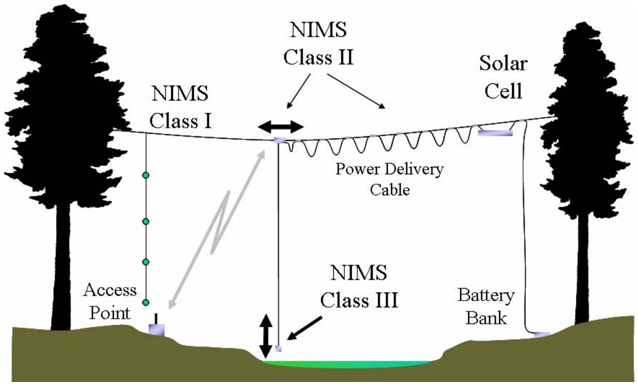

Figure 1. NIMS Class I, II, and III architectures are shown in schematic view, deployed in a forest environment for microclimate and water system monitoring. NIMS Class II devices move horizontally and control elevation of attached NIMS Class III nodes. Class I nodes suspended from the infrastructure are shown as well.

Many important phenomena require extended periods of observation over multiple seasonal cycles. Not only are conventional battery sources inadequate for long term support of many sensor element types, but also, energy harvesting methods for static nodes may not provide sustainability over the required time frame. Specifically, the energy sources (e.g. solar radiation) available for harvesting may not be spatially distributed according to the ENS node energy consumption requirements. Finally, many important environments for ENS monitoring are characteristically large, three-dimensional spaces where measurements must be performed.

This paper describes the Networked Infomechanical Systems (NIMS) ENS platform and its contribution to address the fundamental and important challenges described above [5]. As will be discussed, many deployments in the natural environment of ENS devices now show that a wide range of sensors and now actuator systems are required for characterization of the phenomena important to environmental science. Node and system architecture must then address these requirements. The NIMS architecture described here has been shown to address sensing uncertainty [6], adaptive sampling requirements [7,8], and operation in large three-dimensional environments with energy harvesting capability [5]. A series of NIMS architecture classes have been developed and deployed (as shown in Figures 1 and 2).

This material is based upon work supported by the National Science Foundation (NSF) under Grant No. ANI-00331481. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

0-7803-9201-9/05/$20.00 ©2005 IEEE

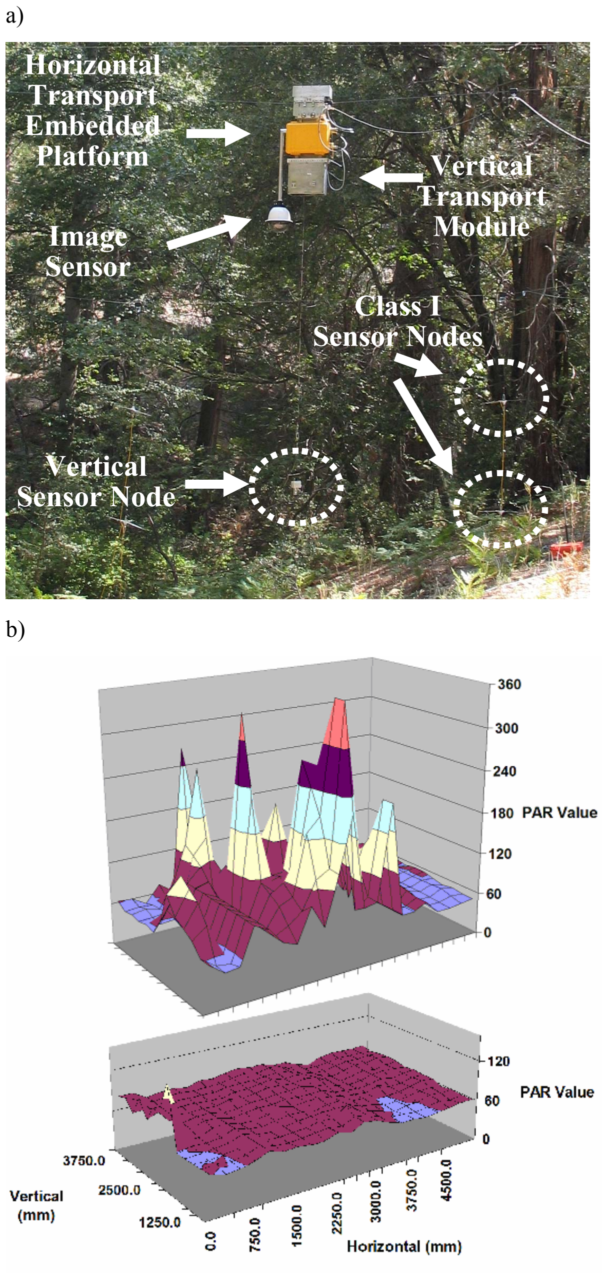

Figure 2. a) A NIMS platform designed for characterization of forest environment microclimate. The vertically actuated sensor node and vertically-suspended, static sensor nodes are also shown. All nodes are linked over wireless networks. b) Acquired maps of photosynthetically active radiation (PAR) solar light intensity. These were acquired by actuating the node within the transect plane (coordinates indicated as horizontal and vertical displacements in millimeters) while sampling PAR sensors using the systems to be described below. These maps were acquired at intervals of one hour during normal daylight conditions. These reveal the typical dramatic change in light distribution occurring in the environment due to solar illumination angle change and forest canopy physical structure.

NIMS embedded system implementation raises several important challenges: 1) The NIMS system requires embedded platforms that incorporate diverse sensors and actuators, accommodate runtime configuration, and provide status verification of these devices. 2) The NIMS capability to extensively and intensively probe environments raises the requirements for in-network sensor data processing with capabilities beyond those typically encountered for static embedded sensors. This includes embedded adaptive sampling algorithms that require capable statistical computing analysis of field variables.

Previous work [5-8] has described the first applications of NIMS. However, this paper describes new embedded platform features that address these new challenges. This paper will briefly introduce the new NIMS classes, and then will include a description of the new processor-preprocessor hardware and software architecture based on EmStar [9] design principles. This paper will further describe the successful development of methods for an embedded statistical computing environment based on R systems [10,11]. Along with other NIMS developments, these are the subject of an open-source release and will be of general interest to ENS developers.

II. NIMS ACTUATION FOR DISTRIBUTED SENSING

NIMS is based on a hierarchical architecture including fixed, mobile, and actuated sensor nodes. An aerial cableway infrastructure physically supports nodes in three dimensional volumes, provides a medium for information and energy transport, and enables low power, precise, and deterministic actuation. Actuation enables self-reconfiguration for adaptation to environmental dynamics. The aerial perspective further enables a wide range of sensing locations and sensor viewpoints that may be applied for enhancing sensing fidelity. Also, by operating above the surface, the NIMS system enables avoidance of typical surface-based obstacles found in natural environments that would otherwise render mobility to be either ineffective or uncertain in nature. By lowering and elevating devices from the suspended NIMS node structure, shown in Figures 1 and 2, measurements can be acquired in an entire transect plane. The cableway infrastructure enables NIMS to determine its precise location as it relies on feedback mechanisms calibrated to the infrastructure. The infrastructure further enables the mobile node to select a static, aerial perspective without power dissipation for maintaining its elevated station.

It is useful to compare NIMS methods with conventional actuation. Specifically, other types of mobile systems include aerial and ground based vehicles. Aerial vehicles offer the capability to range rapidly over large three-dimensional environments. However, navigation of such vehicles in complex environments is challenging. Moreover, many measurements require long residency times at specified locations with low acoustic emission; characteristics incompatible with conventional aerial vehicles. Ground-based vehicles may also traverse large areas; however, they may be incapable of navigating complex terrain or may impact the environment in undesirable ways.

III. NIMS ARCHITECTURE

NIMS architecture has been designed to address the unique set of requirements associated with environmental monitoring. NIMS systems may be divided into four classes distinguished according to whether the ENS nodes and their supporting infrastructure are static or physically reconfigurable.

Class I systems are composed of static sensor nodes supported by static infrastructure. Here, rapidly deployable cable infrastructure supports sensors within 3D environments as well as providing energy distribution to nodes. Class II systems are composed of mobile nodes that propagate on fixed infrastructure. Infrastructure may be rigid or flexible, composed of rails, tracks, or cables. Class III systems are composed of mobile infrastructure with fixed nodes. The mobile infrastructure imparts mobility to otherwise static nodes for either vertical or horizontal motion. Class IV systems are composed of both mobile infrastructure and mobile nodes. The combination of multiple NIMS classes enables NIMS to achieve complex motion including vertically suspended elements that have three spatial degrees of freedom.

An example of a NIMS field system, shown in Figure 2, is a second generation, all weather, and robust NIMS node that is currently deployed in the James San Jacinto Mountains Reserve [12]. This has been developed for the characterization of fundamental microclimate phenomena characterization. The specific system shown in Figure 2 has been operating since March 2004. It continuously collects data between brief offline periods used for scheduled maintenance and upgrades that have occupied less than one percent of the total operating period. This system is composed of Class II and III modules. The horizontal transport is a Class II module, containing a servo motor actuator, actuation power train, and vertical node transport in a sealed package. The horizontal transport also contains an actuated imager. The vertical sensor payload supports temperature, relative humidity, and photosynthetic active radiation (PAR) optical sensors.

NIMS Class I nodes, referred to as "sensor strands" are suspended as shown in Figure 2. These share the same processor-preprocessor architecture (to be discussed) as that of the mobile NIMS node. These provide high rate sensor event detection that has been important for enabling event aware task allocation [8].

NIMS may scale down to compact devices operating in indoor environments. NIMS Laboratory System (NIMS-LS) is an extensible, rapidly deployable Class III system that employs a pair of motor actuators that may retract or extend cables from a pair of cable spools. Wireless network access permits a mobile NIMS node to control its position through communication with an embedded motor actuator controller. Algorithms including adaptive sampling [7] and task allocation [8] have been verified on NIMS-LS here before operation on the field systems shown in Figure 2.

IV. NIMS EMBEDDED PLATFORM HARDWARE AND SOFTWARE ARCHITECTURE

A. *Design Requirements*

A typical NIMS deployment incorporates multiple NIMS node classes with many sensors, actuators, embedded processors, and network interfaces. In addition, diverse algorithms are supported ranging from motion control, imaging, adaptive sampling, and task allocation. This, in turn, also requires the support of multiple programming languages and models. Finally, another class of sensing applications requires high throughput computing capability for in network processing, as will be described further. For example, image processing capability is required for many NIMS applications. Also, powerful statistical computing methods are required as well for adaptive sampling and related functions.

These combined requirements are addressed by an embedded platform architecture that provides access to a wide range of peripheral devices accessible over digital interfaces as well as analog interfaces and requiring real-time control. This implies the need for a dedicated processor that specifically supports these many devices and real-time tasks.

B. *NIMS Platform Hardware Architecture*

Typical applications of the NIMS embedded sensor platform require sensor interfaces for environmental sensing as well as for platform status verification including position, actuator state, and contact limit detection sensors. Characteristically, this class of device interfaces presents demands for low latency service, while at the same time, offering a low bandwidth load. This includes digital and analog sensor interfaces for both environmental monitoring and for critical fault detection sensors that may indicate collision events. A second class of device interfaces presents a latency tolerant demand for service, but, present bursts of wide bandwidth load. This includes the examples of imaging and communication interfaces (both wireless and wireline). It is desired, therefore, to formulate an architecture that is matched to these diverse demands and yet also may achieve low energy operation.

A preprocessor – processor architecture, shown in Figure 3, is adopted for NIMS with an approach that directs the preprocessor to real-time support of sensor systems for both environmental monitoring and system health monitoring. The preprocessor demands are adequately met by typical 16-bit micropower microcontrollers. The processor, in contrast, is selected from the class of 32-bit systems, providing not only high performance, but, also support for the Linux operating system. This latter choice provides many design options to be discussed.

Two high capability 32-bit processors are used in the NIMS 1 system. The vertical sensor node, shown in figure 2a, contains an Intel Stargate platform hosting a PXA-255 X-Scale processor[9]. The Stargate platform obtains 802.11b wireless access from a standard PCMCIA wireless card. It communicates to the preprocessor via its Ethernet interface. The horizontal transport, also shown in figure 2, houses a more capable 1.6GHz Intel Pentium-M processor on an Advanced Digital Logic MSM855 PC104 Plus board. The Pentium M provides higher performance than the PXA-255 processor with hardware floating point support and cache size important for sensor and image data processing hosted on the horizontal node. The PC104 is compact, expandable, and supports the Linux operating system installed on a compact hard disk drive. Its peripherals include PCMCIA, enabling the use standard 802.11b wireless interfaces, a Sensoray 311 framegrabber for acquiring images from the image sensor, and a Connecttech Xtreme-8 port serial card for supporting the many required serial interfaces.

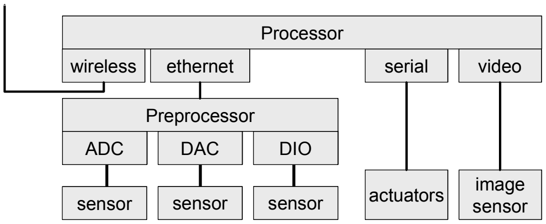

Figure 3. The NIMS Processor/Preprocessor architecture showing sensor interfaces supported by the preprocessor and actuator and image sensor interfaces supported by the processor.

Note that devices offering low bandwidth demand are serviced directly by the preprocessor. Interfaces presenting wide bandwidth demand (including network access and imaging) are serviced directly by the high capability processor. Together, this enables the processor to operate episodically, on demand. However, the preprocessor may operate continuously and at low average energy while providing low latency communication support to sensors and actuators.

The NIMS platform preprocessor implementation is a ZWorld BL2600 single board computer. It contains 8 11-bit analog input channels, 16 digital inputs, 4 high current digital outputs, 5 serial ports, and an Ethernet interface. The single ended analog sensing channels include configurable sample periods and adjustable gain for voltage ranges of 0-1V to 0-20V. The preprocessor utilizes ethernet to interface to the high capability processor.

The NIMS sensor suite includes a Licor LI190-SZ PAR quantum light sensor and a Campbell Scientific hmp-45c combination temperature and humidity sensor with a radiation shield. In addition, the vertical node uses a Finemec piezoelectric acoustic range sensor and a Newark microswitch contact sensor for its fault detection devices. The image sensor is a Canon VC-C4R pan, tilt, zoom camera with full motion video and serial controlled actuator. The actuator system is composed of Parker Hannifin SM233BR-NMSN servo motors with integrated 3:1 gearheads and IP65 wash down grade seals, controlled by Vix 500IE servo motor controls serviced over serial links. Additional systems include a National Control Devices serial interface relay power switching system, controlled by the processor, to enable the individual control of power state of each system module including the sensors, actuators, and vertical node system.

C. *Network Access and Network Architecture*

NIMS systems utilize wireless networks for internode communication. The NIMS 1 system uses standard PCMCIA IEEE 802.11b wireless interfaces with either high gain dipole or dielectric patch antennas. For rapid delivery of urgent measurement data conveying motion and sensing coordination, the nodes are interconnected via an IEEE 802.11 infrastructure-mode network with an Internet accessible access point (basestation). SSH protocol transport provides authentication and security for remote access with typical data transfer rates of 200 kbps between nodes and 40kbps rate to remote Internet hosts.

For the rapid deployment and configuration of NIMS, each fixed and mobile node is accessible to the NIMS horizontal node, forming a single network cluster. Network access to both the vertically actuated sensor as well as to fixed sensors also exploits the EmStar system. For each node, the preprocessor establishes a TCP/IP link over the 802.11 wireless interfaces to the EmStar client (to be discussed) hosted on each node.

D. *NIMS Energy Usage and Transport*

Typical measurement requirements include continuous monitoring over extended periods that includes the support of high power dissipation sensors. Thus, NIMS systems have been designed to exploit infrastructure for both energy harvesting and transport. The first NIMS field deployment has included suspended solar cells, for energy harvesting, as shown in Figure 1. The current NIMS 1 system relies on the James Reserve's energy harvesting system for sustainable operation. The system includes a solar cell array and battery bank (A standby generator is available for emergency use). Siemens SP75 solar panels, a typical panel used in deployment, can harvest an average of approximately 250 Watt hours of energy per day. This number is heavily dependent upon weather conditions, location, and season. High voltage transport is leveraged to reduce resistive transmission line energy loss. Standard inverter systems convert 12VDC sources and provide 110VAC distributed through a networked enabled uniterruptable power supply (UPS) to the node via a folding festooning cable, extending along the node support cable. The vertical node is powered by conducting cables running parallel to the stainless steel support cable. An electrically conducting slipring transfers energy from the horizontal node to the spooled vertical node wires. The slipring ensures that the wires do not twist as the spool rotates to raise and lower the vertical node.

Each primary subsystem component may be individually controlled in its power usage. Power demand includes: 1) Wireless interface with power in idle, receive, and transmit draws power of 690mW, 870 mW, and 1110mW, respectively, 2) The camera system draws power of 7.6W, 3) The horizontal node with computing platforms, thermal energy management, power control system, draws 15W, 3) The vertical node system draws 15W, 4) The actuator system power in standby is 35W with additional 5W-12W applied for horizontal motion and 5W – 15W applied for vertical motion. Although the motor system consumes 35W of standby power due the Vix motor drivers and power supply, the system can be shutdown while not moving. High ratio worm gears, incorporated in both the horizontal and vertical power trains, keep the node stationary even with the motor system powered down. Typical motion speed is 0.2 m/sec.

E. *NIMS Platform Software Architecture*

The NIMS software architecture has been developed to meet the diverse demands presented to the NIMS development and runtime environment. In addition to the sensor and actuator interfaces discussed above, it must also support a diversity of applications including those for adaptive sampling, task allocation, and image processing for landmark-based geolocation, object detection, and tracking. Finally, an important additional design requirement is that each NIMS system type, whether an external field environment or indoor testbed system, must support the same algorithm implementations independent of specific selections of processor platform, sensor, or actuator systems. Thus, the development, simulation, and emulation environments must enable the same application source code to be hosted on each platform to enable development and verification in the laboratory of systems that will be deployed in the field.

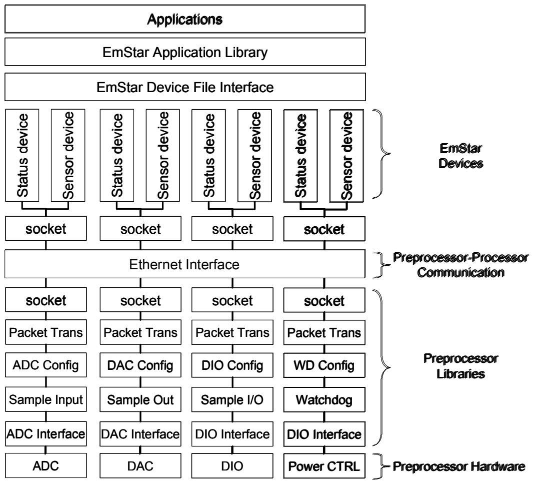

Figure 4. The NIMS Processor-Preprocessor software architecture is shown. This architecture exposes the standard EmStar status and sensor device file interfaces to each of the analog input (ADC), analog output (DAC), and digital input/output (DIO) to applications. An additional interface may be added as well for watchdog fault recovery functionality.

F. *NIMS EmStar Systems*

The EmStar [9] development and runtime environment for embedded networked sensor systems has been applied to NIMS design and implementation. NIMS design follows an event driven programming model, naturally supported by EmStar. This reactive design approach is applied to the processor-preprocessor architecture described above, providing methods for servicing unscheduled events and managing out-of-order operations, typical of unattended operations in complex environments. The EmStar Framework for User Space Device Drivers (FUSD) device file interface is used for both sensor and actuator interfaces and maintains the same interfaces across all NIMS platform types [10]. The use of standard device file interfaces allows visibility into internal device state and convenient manual or autonomous configuration. This facilitates in-field testing without interruption of system operation.

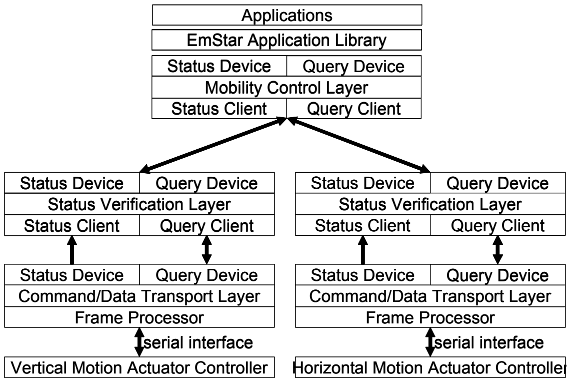

Figure 5. The NIMS EmStar actuator multilayer driver architecture providing configuration and control interfaces for the primary NIMS horizontal and vertical transport.

G. Processor-Preprocessor Software Architecture

The preprocessor-processor system (shown schematically in Figure 4), is based on multiple EmStar client processes communicating over socket interfaces to multiple preprocessor servers dedicated to each sensor device interface. EmStar constructs status and sensor devices for each channel, enabling a user to conveniently control each channel via a standard device interface API, adjusting sampling rate, gain of each input channel, or controlling an analog or digital output. Applications control the preprocessor by reading and writing to the standard EmStar status and sensor devices. These systems are deployed in the field and are operating continuously and autonomously in support of environmental measurements. Unique capabilities that leverage this same design paradigm are being added, including low energy platform scheduling and watchdog error recovery functionality.

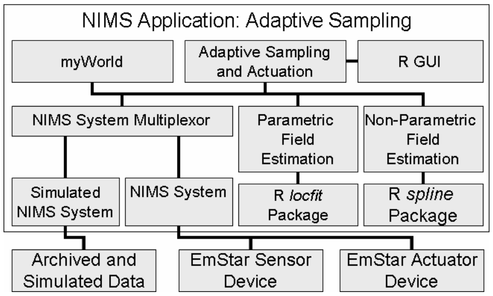

Figure 6. The architecture for the embedded R statistical computing environment on the NIMS node is shown. NIMS R code applications may access sensors and actuators in real-time field, simulated, or archived data creating the myWorld application environment.

H. Actuator Control Software Architecture

The EmStar system enables the development of a layered actuator driver architecture, as shown in Figure 5. The actuator driver implementation relies on many libraries and device patterns in EmStar, such as the status device, query device, and frame processor device patterns. It exploits the status and query client libraries in EmStar to simplify the interaction between actuator driver layers. For example, the semantics of the query device promote a modular design, such that each layer is simplified and self-contained. In the three-layer actuator architecture illustrated in Figure 5, the lower Command/Data transport layer driver interacts directly with the actuator controller through a serial communication port exchanging command and status data. The middle Status Verification layer driver achieves a complex functionality level, including continuous verification of the actuator status. Finally, the upper Mobility Control layer driver is responsible for node actuation control in a 2D space. It is at this layer that specific mapping of actuator motion to NIMS node motion is managed. For example, it is here where the motion control for field NIMS (Figure 2) and NIMS-LS are managed. This enables diverse platforms having various electromechanical systems to show the same application layer interface for node motion control. It is also at this upper layer that motion limit verification is performed. Any one of the three layers may be conveniently reconfigured at runtime. In addition, the lowest driver layer may be independently adapted to specific actuators.

V. NIMS EMBEDDED R SYSTEMS FOR IN-NETWORK STATISTICAL COMPUTING

The first investigations of natural environment phenomena with NIMS have revealed new characteristics of fundamental field variables. An example is the spatiotemporal distribution of solar illumination, as shown in Figure 2b. The complexity and rapid evolution of this phenomena immediately demonstrated that fixed sensors or even simple actuated sensor scanning methods would be inadequate for high fidelity mapping. Thus, statistical computing tools are required for in-network processing of this complex data source in order to enable algorithms needed for adaptive robotic sampling [7,8]. Thus, a means is required to host the statistical computing environment directly by the NIMS node system. A new embedded platform capability based on the R statistical computing environment has been developed to support this requirement and is reported here.

The R language [11] is a GNU GPL implementation of the S language and computing environment, originally developed at Bell Laboratories [12]. Similar to the familiar Matlab script structure, the R language has as its basic data types, vectors and matrices. However, in contrast to other languages, R provides access to a wide range of both classical and modern statistical methodologies. A large community of research statisticians are actively extending the functionality of R by contributing numerous computational and visualization techniques. Both S and R were designed to be extensible, either by creating new functions (methods and classes) directly in the R language or by linking code written in other languages.

S, and hence R, have inherited much from John Tukey's fundamental formulation of exploratory data analysis. Specifically, this iterative process of modeling and inference demands that languages and environments for statistics computation allow users to manipulate data through higher-level objects and to further extend the system to include their own computations and graphical displays. R finds its strength in the rapid auditioning of statistical and graphical methods. This represents an ideal computing environment for statistics research that incorporates sensor data sources and data acquisition based on actuated sensors in the environment. Thus, R has been chosen as the statistical computing environment for in-network sensor data processing on the NIMS node.

The use of R in the NIMS framework has precedents in other efforts to "export" or share statistical functionality in familiar server, database, and language tools [13]. This is motivated by developer requirements for statistical computations. Embedding R exposes its functionality, avoiding recoding of complex methodologies in enclosing applications. For example, as will be discussed, nonparametric estimation schemes based on splines, wavelets, local polynomials, and radial based functions have all been implemented and tested in R as components in the development of NIMS adaptive sampling. The built-in graphical routines have also aided algorithm refinement and evaluation. In short, R has aided the data analysis on the results of past, or even currently executing, experiments.

The usage of embedded R also introduces challenges. In particular, proper coding methods are important in R to take advantage of its built-in vectorization of operations and ameliorate its otherwise reduced performance for certain computations. Although a large external database may be available, algorithms requiring adaptive behavior to environmental inputs must be hosted on the local node since network connectivity to a remote host is not guaranteed.

The structure of complete applications hosted on the NIMS node, adaptive sampling for example, is shown in Figure 6. Here, a set of NIMS classes and libraries have been developed providing navigation and sampling strategies. These libraries rely on standard R packages. For example, our standard spatiotemporal field estimation relies on the "Local Regression and Likelihood" package [15]. This enables the simultaneous generation and estimation of the measurement field and the collection of data. In addition we use these standard packages to generate derivatives and gradient of the estimated field. This information can be visualized as a contour map or as a perspective view (provided by R) for real time field visualization. In addition, this information is used in guiding the robot to navigate intelligently across the field.

R applications rely on actuators and measurement instrument system interfaces provided by Emstar. The appropriate R device is responsible for writing proper actuation information to the EmStar actuation device. Similarly, sensor data is acquired by reading the proper file system device corresponding to the desired sensor. These devices enable hosting of R code in real time applications.

VI. LESSONS LEARNED

The NIMS system described here is a second generation prototype node, designed and implemented based upon lessons learned from the first NIMS node deployment at the Wind River Canopy Crane Research Facility [15] in September 2003. That first deployment was a short term, fair weather, monitored test of NIMS systems, software, and capabilities. This first and subsequent deployments at the San Jacinto James Reserve [16] have provided essential data, affecting our designs in the following ways:

1) Data acquired from suspended mobile NIMS systems have inspired the need for expanded experimental capabilities including precise imaging, gas phase sampling, and water system characterization. These requirements have driven the development of the current NIMS system with its ability of rapid incorporation of new software and hardware modules via EmStar methods.

2) Reliable operation depends not only on dedicated embedded computing modules implemented expressly for NIMS, but, also on peripherals including sensor and actuator devices. These devices exhibit fault behaviors driving them into unstable or nonfunctional states. Recovery requires individual module level power control of every component as implemented in this NIMS system. These methods have enabled the use of commercial off-the-shelf components for rapid addition of systems that meet new user demands.

3) Remote nodes require methods for reliable configuration management and debugging methods. The use of platforms supporting the Linux operating system has enabled configuration management, on-line debugging, and the support of multiple concurrent users. Management tools have proven to be required and effective with bandwidth efficient transport of summary status information including, for example image and sensor data, and system information including kernel ring buffer contents.

4) Wireless link characteristics have benefited from the incorporation of high gain antennas into NIMS systems and these in turn enable connectivity to distributed fixed nodes.

5) Development and testing of algorithms on remote systems is costly in time. Experience in this area has lead to the development of the emulation systems of Section V and the NIMS-LS verification system of Section III. Further, developing sophisticated algorithms needs a simulation environment that enables the developer to test the algorithm rapidly and frequently. The integration of the R environment with NIMS systems not only enables the user to communicate with the real node, but also lets the application communicate with archived or synthetic data sets. The porting of code from the emulation to field node has proceeded with minimal modification for both adaptive sampling and task allocation algorithms.

VII. CONCLUSIONS

NIMS introduces an embedded networked sensor architecture exploiting infrastructure for constrained actuation. Precise, wide-range actuation enables NIMS systems design to address fundamental problems associated with traditional static sensor networks, including sustainable adaptation to environmental dynamics and detection and active reduction of sensing uncertainty. A processor-preprocessor hardware architecture along with a matching software architecture based on the EmStar system has provided the support required for many novel NIMS applications. This architecture is generally applicable to other embedded robotic sensor systems that face similar challenges for support of diverse sensors, actuators, and high throughput computing demand. Also, the NIMS software architecture now includes an embedded statistical computing environment based on R. This enables applications developed with the full R capability to directly access both actuation and sensing. The combined NIMS hardware and software platforms will be released as an open source contribution accompanying their supporting EmStar framework [17]. Also, the encapsulation of basic R operations into an object-oriented programming structure will be released as an R package of general interest to robotic sensor system developers.

VIII. REFERENCES

[1] D. Estrin, G.J. Pottie, M. Srivastava, "Instrumenting the world with wireless sensor networks," ICASSP 2001, 2001.

[2] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, Kristofer Pister. "System Architecture Directions for Network Sensors". Proceedings of ASPLOS, 2000.

[3] C. M. P. Ozanne, D. Anhuf, S. L. Boulter, M. Keller, R. L. Kitching, C. Korner, F. C. Meinzer, A. W. Mitchell, T. Nakashizuka, P. L. Silva Dias, N. E. Stork, S. J. Wright, M. Yoshimura, "Biodiversity meets the atmosphere: A global view of forest canopies," Science, vol. 301, pp. 183-186, July 2003.

[4] D. Estrin, L. Girod, G. Pottie, M. Srivastava, "Next century challenges: Scalable coordination in sensor networks," Mobicom, 1999.

[5] W. Kaiser, G. Pottie, M. Srivastava, G. Sukhatme, J. Villasenor, D. Estrin, "Networked Infomechanical systems (NIMS) for Ambient Intelligence," Center for Embedded Networked Sensing Technical Report, No. 31, December 2003.

[6] A. Kansal, E. Yuen, W.J. Kaiser, G.J. Pottie, M.B. Srivastava, "Sensing uncertainty reduction using low complexity actuation," Proceedings of IPSN 2004, 2004.

[7] M. Rahimi, R. Pon, W.J. Kaiser, G.S. Sukhatme, D. Estrin. M. Srivastava, "Adaptive Sampling for Environmental Robotics," IEEE International Conference on Robotics and Automation, 2004.

[8] M. Batalin, G. S. Sukhatme, Y. Yu, M. H. Rahimi, G. Pottie, W. Kaiser, and D. Estrin, "Call and Response: Experiments in Sampling the Environment," Proceedings of SenSys 2004, pp. 25-38, 2004.

[9] L. Girod and J. Elson and A. Cerpa and T. Stathopoulos and N. Ramanathan and D. Estrin, "EmStar: a Software Environment for Developing and Deploying Wireless Sensor Networks," USENIX, 2004.

[10] http://platformx.sourceforge.net

[11] http://www.r-project.org/

[12] J. M. Chambers, "Computing with Data: Concepts and Challenges" The American Statistician, pp. 73-84, 1999.

[13] http://www.omegahat.org/

[14] http://www.locfit.info/

[15] http://depts.washington.edu/wrccrf/

[16] http://www.jamesreserve.edu/

[17] http://www.cens.ucla.edu/portal/nims