ITR/SY: A Distributed Programming Infrastructure for Integrating Smart Sensors

NSF Program CCR-0121638

PIs: Umakishore Ramachandran
Kenneth Mackenzie
Steve DeWeerth
Irfan Essa
Thad Starner

College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
Phone: (404) 894-5136
FAX: (404) 385-2295
e-mail: rama@cc.gatech.edu
WWW URL: http://www.cc.gatech.edu/~rama

Annual Report via Fastlane June 14, 2003

1 Activities and Findings

1.1 Research and Education

The proposed research is integrating sensing hardware, embedded processing and distributed system support to build a seamless programming infrastructure for ubiquitous presence applications. Fundamental invention and integration of techniques spanning programming idioms and runtime systems for distributed sensors, and building blocks for embedded processing are expected as the primary intellectual contributions of the proposed research. Interfacing these technologies to emerging applications on the one end and novel off-the-shelf sensors at the other end are secondary goals of the proposed research.

This subsection details the research accomplishments this past year (2002-03) as well as plans for the coming year (2003-04).

1.1.1 Distributed Systems Technology

Media Broker Architecture There was a time when the world of sensors and sensor-based processing (mostly in the world of DSP) was distinct from mainstream high-performance computing. It is becoming increasingly clear that the two worlds are inevitably merging as sensors and embedded systems become more ubiquitous in our environment. There is an interesting philosophical conflict in this merger. The sensor-driven DSP world knows the value of "throwing" away data as much as using sensitive data. The mainstream computing world with its bag of tricks for consistency, check-pointing, etc., is very much concerned about keeping around data expecting everything to be used! Given the confluence of sensor-driven and mainstream computing, there is a need to merge the bag of tricks in the two worlds and perhaps discover unifying tricks for the merged sensor-based distributed computing fabric. In this merged world, interfacing and integrating with the physical environment will be a key property of high-performance distributed computing.

We expect more and more mainstream applications of the future to take on a "control loop" flavor in the following sense: inputs from a variety of distributed sensors may be gathered, filtered, processed, correlated, and fused to facilitate higher levels of hypotheses and inferences (implying heavy duty processing) which may then be used to drive distributed actuators to affect the physical world in some application-specified manner. Examples of such applications include mobile robots, smart vehicles, aware environments, disaster response, surveillance, and environmental monitoring.

We are building a *Media Broker* that is aimed at addressing adaptive naming, resource discovery and data fusion. The media broker (see Figure 1) acts as a clearinghouse for sensors and actuators, for data naming and efficient resource discovery, and for specifying application specific behaviors such as sensor fusion, and is being built on top of the D-Stampede [1] system. We feel that naming and resource discovery should be elevated to the level of programming. Data fusion is a natural way to realize virtual sensors. For e.g., one can think of a virtual sensor that fuses corresponding inputs from two or more real sensors (such as audio and video). In this sense, data fusion is closely related to the naming problem: naming and resource discovery are common issues to be dealt with for both real and virtual sensors. Identification of the problems in constructing complex control loop applications and Initial ideas on addressing them have been reported in [11]. Two undergraduates Martin Modahl and Ilya Bagrak have been involved in the implementation of the media broker architecture.

Data Fusion. Following the technology curve, the sensor "motes" of tomorrow are expected to have the computation and communication capabilities of current handhelds. Thus, advanced multimedia applications that use sophisticated computing infrastructures today may become viable for deployment in future wireless ad hoc sensor networks. Simple in-network data aggregation techniques for sensor networks have been the focus of several recent research efforts. In this research, we extend these techniques to future sensor networks and ask two related questions: (a) what is the appropriate set of data aggregation (or fusion) techniques, and (b) how do we optimally assign aggregation roles to the nodes of a sensor network. We have developed an architectural framework, *DFuse*, for answering these two questions. It consists of a data fusion API and a distributed algorithm for energy-aware role assignment. The fusion API enables an application to be specified as a coarse-grained dataflow graph. It also eases application development and deployment. The role assignment algorithm maps the graph onto the physical network based upon the network topology and environmental conditions, periodically adapting the mapping in case of changes using role migration. Extensive measurements on an iPAQ handheld farm show that DFuse API calls incur low overhead. Preliminary results from this research are reported in [6].

Migration of Computational Abstractions. A key technology that is needed to support the DFuse architecture is on-demand migration of computational abstractions to match the role assignment. We have designed and implemented facilities in Stampede for dynamically packaging and migrating Stampede channels without disrupting the application progress.

Dead Timestamp Identification in Stampede. This is a continuation of the activity that we started last year in collaboration with Dr. Kath Knobe of HP CRL. We have developed a theory for identifying and eliminating irrelevant items in a Stampede computation and reported it in the 2001-02 annual report. The measurement infrastructure coupled with the post-mortem analysis technique we have developed in Stampede allows us to ask several "what if" questions to validate or eliminate possible design directions for optimizing the performance of the Stampede runtime system.

Towards Aspect-Oriented Programming Support for Cluster Computing. This is continuation of an effort to automate certain aspects of distributed programming technologies, in particular the exploration of parallelization strategies and the plumbing among the computational entities. We are extending STAGES that provides these functionalities for a cluster computation to a fully distributed framework with dynamic join/leave capabilities. We are also using this as a starting point for providing higher level debugging facilities in Stampede.

Stampede.NET. An effort started recently extends the Stampede programming model to wide area internet using the Microsoft .NET infrastructure. The basic value that this efforts to the overall project is to enable access to the Stampede programming model via the Web Service paradigm. The Web Services paradigm is predominantly used in the B2B domain for XML based document exchange. Using this paradigm for Stampede-style distributed stream processing opens new opportunities for stress testing the transport layer and evolving new protocols. .NET has mechanisms for supporting language heterogeneity and garbage collection. We are exploring ways to exploit these features for supporting the runtime model of Stampede.

1.1.2 Mechanisms for Embedded Architectures

Software Caching for Embedded Sensor Processing. This is continuation of the work we reported in the 2001-02 annual report to reconcile programmability with cost for embedded processing associated with sensors in a distributed sensing infrastructure.

Using our initial implementations of a SoftCache system for both Sparc and ARM/XScale systems [5, 4], optimization efforts are underway to reduce the penalties incurred by the SoftCache. Part of this work is attempting to characterize and predict moving from steady state to steady state without emulation of block invalidates ([3] contains some aspects of this work in which we have involved a female undergraduate Naila Farooqui).

Due to the complexity of debugging binary rewriting and translation systems with no *a prior* source code knowledge, the ARM/XScale version which uses a client-server methodology has been incorporated into SimpleScalar 4/ARM and gdb. SimpleScalar has been modified to support a large body of missing kernel functions (sockets, mmap, etc.), while gdb has been modified such that it can directly read and manipulate the server data structures to facilitate debugging. In this manner, translated addresses and code can be directly mapped back to the original program, and gdb sessions can be synchronized to understand the point of failure. Two undergraduates (Christopher J. Parnin and David Raeman) have been at the center of getting this work done.

To address the energy-delay issues that the SoftCache model forces on target devices, work has begun on addressing the different affected dimensions. After a review of methodology, we observed that the Energy Delay metric proposed by Gonzalez and Horowitz has been applied inconsistently. After addressing this issue by publishing a study of the problems [7], we have performed a detailed analysis of the energy-delay impact when replacing local memory with remote network accesses. Our results suggest that using the network as a remote memory can be more energy efficient than local memory under a wide range of conditions [2]. Additional analysis has been applied to the reorganization of the on-die structures, considering application performance and energy consumption of the processor overall [3].

Work is also moving forward on fully instrumenting an XScale platform (the ADI BRH from ADI Engineering, Inc.) to perform live power measurements while allowing for dynamic voltage- and frequency-scaling. Once completed, this instrumentation will allow for hardware verification a new analytical model. This modeling will account for individual power contributions to the overall system of DRAM, network, CPU, etc. This will permit correlation of simulated SoftCache power savings and network energy efficiency to typical hardware.

1.1.3 Sensor Technologies

Sensor Lab. We have provisioned a Sensor Lab within the College of Computing to support our research into flexible software infrastructure for pervasive, heterogeneous, sensor networks. Our lab includes modest quantities of a wide variety of devices, both wired and wireless, from simple commercially available (COTS) components to complex research prototypes.

In last year's report, we described in detail five specific sensor technologies we planned to deploy including an IR tag-based location system, passive temperature sensors, ultrasonic sensors, Berkeley Motes, and HP SmartBadge systems. All have been acquired and all but the ultrasonic sensors and Berkeley Motes have been deployed. In addition we have made extensive use of cameras, microphones and speakers and will be acquiring some RFID equipment and proximity cards for use in the Future Library project.

We have continued outfitting the Sensor Lab within the College of Computing and had a group of students develop a demo application that exercised the deployed sensor infrastructure. The Systems Studio and Systems Lab, along with other portions of the second floor of the College of Computing, have been outfitted with a commercial IR/RF tag-based tracking system. This system provides room-granularity tracking information on participants caring a tracking badge. Low-level sensor data is processed by a proprietary server which then emits tracking data in XML format to our Stampede infrastructure where it is published for applications. The Sensor Lab also contains a variety of microphones and cameras, with varying capabilities, including a pan-tilt-zoom camera. We have contact sensors on doors and passive temperature sensors. We have also been working with iPAQs and prototypes of the HP SmartBadge, which is a portable, wireless single-board computer with attached sensors that runs Linux. Finally we have started working with Berkeley Motes which provide a rich range of capabilities.

Our demo application involves tracking a target individual carrying an IR badge. This provides coarse location information (e.g. in the hallway, in the lab). we have sensors inside and outside doorways to detect entry/exit events. This data is further reinforced (via sensor fusion using the Fusion Channel API in Stampede) by door contact sensor readings. Once a tracked target is confirmed as entering the lab, a pan-tilt-zoom camera orients and takes a snapshot of the individual entering the room. This is stored for future reference. To track location within the lab, we have deployed simple login monitors on the various machines. Keystroke signatures are used to confirm that an individual logged into an account is the actual target. Using a simple spatial model of the sensored labs and spaces, appropriate cameras are identified to provide surveillance of the tracked target. This demo is mostly complete and exercises the infrastructure while providing a framework for further research. For example, part of the world model includes a sensor/actuator device registry. We have explored simple XML encodings of device characteristics for discovery, all within the Stampede framework. We look forward to augmenting this demo in novel ways with the Berkeley Motes. This demo used the HP SmartBadge as a computational node (although we didn't employ the onboard sensors) running Stampede.

The wireless temperature sensors and HP SmartBadge platform were both used by students in Professor Ramachandran's Pervasive Computing seminar (8803E) during Spring semester of 2003. We hope to see make the Sensor Lab more widely available to classes and students in the future.

We have done some preliminary evaluation of the Berkeley Motes but await delivery of several dozen new Motes and supporting infrastructure. Expected delivery is mid-summer 2003 and we hope to gear up quickly, training several undergraduate and graduate students in the use of the Motes and the TinyOS programming environment. Due to the flexibility of the Motes, many projects are possible. Early on we hope to develop a minimal Stampede protocol stack to run on the Motes, allowing them to integrate easily into existing applications and other research efforts.

Two other efforts are related to the Sensor Lab. First, to facilitate ongoing interaction with Professor Ron Arkin and the robotics group here at Georgia Tech, we plan on purchasing and deploying robots currently used by Ron Arkin's Mission Lab system. Robots are effectively autonomous, mobile sensors and provide a good application of the Stampede infrastructure. Having our own dedicated robotics hardware will allow us to progress more quickly with our efforts in this application domain. Second, in joint work with Professor Ramesh Jain, we will be deploying Stampede as part of a campus-wide Georgia Tech Event Web prototype. This system facilitates development and access to web-mediated events. In the Georgia Tech prototype, the system will be used to access talks, seminars, and conferences held on campus. This work will involve outfitting several spaces on campus with teleconferencing capabilities, effectively extending the Stampede-managed Sensor Lab to a campus-wide presence.

Software Infrastructure. Last year's report outlined a variety of infrastructure technologies that were planned or under development including basic sensor support, fusion channels, time synchronization, power management, and the Media Broker. We have made progress on several fronts and some new research directions have crystallized in the intervening year.

We have demonstrated our ability to sample and control sensors and actuators within the Stampede framework with a variety of applications and demos. We have integrated the Fusion Channel abstraction into Stampede. We have undertaken a power management study and developed algorithms and infrastructure for dynamic power monitoring and management by task placement and migration (DFuse). Transparent task migration in Stampede required the development of a channel migration mechanism. This mechanism is useful independently and can be used to implement load-balancing and traffic management.

Efforts on the Media Broker are ongoing. A preliminary implementation has been used to support and structure the Family Intercom application in the Aware Home. This application allows the description and discovery of microphones and speakers throughout the Aware Home. These are often accessed in a combined form as a virtual sensor-actuator pair (ICombo). Information about the placement, attributes and status of devices is maintained in an LDAP repository. The LDAP repository also functions as a name server, allowing abstract named access to Stampede internal entities (e.g. channel descriptors, etc.). This information is augmented by a location tracking system in the Aware Home, accessed via a Java interface through the Context Toolkit. The Family Intercom supports tracking of listeners and speakers and activation of nearby audio resources. Ultimately, users will be able to move freely throughout the home with automated "switch-over" of audio devices as the user enters and leaves various rooms.

Development on the Media Broker proceeds on several fronts. We are adding mechanisms for stream publishers (sensors, producers) to specify the various data formats they can produce using a lattice-based type system. Data types are related by possible stream transformations in these lattices (for example, down-sampling, rate reduction, cropping, clipping, compression, etc.). Stream publishers also export a command set for changing from one format to another. Internally, the Media Broker is able to implement further transformations that are part of a library of well-known transformation, or via transforms that are provided by clients and dynamically linked on demand. The lattice structure helps the Media Broker determine a "greatest common format" when conflicting requests are presented. We are also exploring additional description and discovery techniques and are considering the use of DSML (Directory Service Markup Language) for representing LDAP data.

We have also undertaken a refactoring and redesign of Stampede to provide flexibility and performance enhancements and to add new features. This work involves careful object-oriented design and modeling and is proceeding with attention to current best software development practices. This effort will yield an improved and well-documented code base to support future enhancements. This work proceeds synergistically with an ongoing Stampede/.NET implementation in which a C# version of Stampede is being implemented within the .NET framework and exposed via web services. We have begun focusing initially on a redesign of the CLF communication substrate to modularly support a variety of transport mechanisms (UDP, reliable UDP, TCP, HTTP, etc.). We are redesigning this layer to provide a secure and flexible deployment/startup mechanism using SSH. In addition we are designing a dynamic, flexible group membership manager with basic fault tolerance. This will allow an improved, integrated implementation of the D-Stampede functionality (currently implemented via TCP proxying) as well as cluster-to-cluster interactions. The redesign will also support heterogeneous address spaces, eliminating the current requirement that all Stampede cluster participants host copies of a single statically-linked address space.

1.1.4 Application Driven Studies

We continue to drive our research efforts with concurrent application development to ground and confirm our ideas. Last year's report described the TVWatcher and Aware Environments. The TVWatcher project continues as previously described and our work in Aware Environments is currently focused on what we call the Future Library project.

TVWatcher. We have made good progress with the TVWatcher this year and are moving into several exciting new developments this summer. TVWatcher is a GUI client for managing, viewing and correlating complex media streams. It has many applications and relies on a distributed infrastructure including capture hosts, a registry and correlation engines. We have high quality text-based correlation now working and have stabilized and optimized the GUI client. We are currently adding infrastructure to support audio-based and video-based correlation. This summer we will be adding simple OCR recognition of scrolling text on video streams and add this information to the text correlation engine. We hope to add rudimentary audio correlation and simple video frame correlation using principal component analysis. Separately we are developing a means to find and display related websites for the current target stream. We are in the process of writing a paper describing this effort.

Future Library. The Future Library project aims to produce an application framework for pervasive computing using the Stampede infrastructure. Specifically the Future Library is intended to assist a user entering a library to find a desired set of resources. We assume the user is supported by a personal computing environment which could be a handheld or wearable or some combination of devices. We assume bountiful computing resources to project the scenario into the future a few years. We are particularly interested in how a pervasive, computationally rich physical environment can assist in such a resource location scenario. We imagine "smart" or computationally enhanced walls, desks, bookshelves, books, etc. Finding physical resources in a library is an analog to a large variety of important and interesting resource tracking, management, and discovery problems including inventory tracking and control. There are obviously a variety of security and surveillance related applications as well. We are currently exploring the use of RFID and proximity technologies. We are using this project as an opportunity to stretch our imaginations and discover interesting new applications and challenges of our technologies. We plan on having a working prototype by the end of summer 2003.

1.2 Training and Development

1.2.1 Undergraduate Research Participation

We continue to attract bright and interested undergraduates to research projects in our group. Undergraduate participation in research within the College is facilitated by the excellent UROC program (`www.cc.gatech.edu/program/uroc`), coordinated by Amy Bruckman. A variety of institute-wide programs are also available (`www.undergraduateresearch.gatech.edu`) including a special fund sponsored by the president of Georgia Tech (PURA) and several NSF-sponsored projects. We were pleased to support three undergraduate on ITR-related projects during the Spring semester of 2003. They were: Ahmed El-Helw (TVWatcher), Martin Modahl (MediaBroker) and Ilya Bagrak (MediaBroker). An additional student, David Hilley, will be supported in the summer of 2003 by this award for his work on the TVWatcher project. For details of the PURA program, along with a list of recipients, see the website.

Many of the ongoing ITR-related projects are partially staffed by students working in the context of the Systems Hackfest. This is a group of undergraduates who participate in various research projects for pay, course credit, or just for fun. Hackfest is supervised by Research Scientist Phil Hutto and runs throughout the year. Summer sessions are most productive and have recently involved more than 10 students. Students meet briefly in a weekly session to report progress and plan milestones for the coming week. The group meeting allows cross-fertilization of project ideas and helps to educate the students. In addition, it provides an opportunity for group brainstorming on design and debugging issues. Weekly project meetings are focused on specific research tasks and often involve relevant faculty, grad students and staff.

During the last year undergraduates have participated in the following projects: TVWatcher, Family Intercom, MediaBroker, Future Library, Sensor-based Location Tracking, and HP SmartBadge.

1.3 Outreach

In the Fall of 2003, we plan to organize a workshop that brings together researchers from several different areas of sensor-based distributed computing spanning applications to hardware design. The intent is to familiarize everyone with the complementary research efforts in this space so that the overall impact of the ITR program can be far greater than the individual research efforts. The idea for such a workshop originated from a discussion with Dr. Helen Gill in the Fall of 2002.

1.3.1 Advisory Board

Prof. Ramachandran is also a PI on an NSF research infrastructure (RI) award. As part of managing the NSF RI award, we hold an annual research advisory board meeting with external board members. The external board members for our Fall 2002 meeting included: Roger Haskin (IBM Almaden), Yousef Khalidi (Microsoft), Jim Rowson (HP labs), Rick Schlichting (AT & T), Kath Knobe (HP), Willy Zwaenepoel (Rice), Margaret Martonosi (Princeton), and Mary Vernon (UW-Madison).

The internal members included: Jim Foley (COC), Nikil Jayant (GTBI), Krishna Palem (ECE), and Rich DeMillo (COC).

Much of the research work being conducted for the ITR project uses the RI equipment. Thus the advisory board meeting features the research work of the ITR project quite prominently and thus obtains valuable feedback and establishes connections with industries.

The next board meeting has been tentatively scheduled for Spring of 2004.

2 Publications and Products

2.1 Publications

See the references at the end of this document.

2.2 Web Site

Please visit the project web site at www.cc.gatech.edu/~rama/ubiq-presence/

3 Contributions

The activities we are currently undertaking have resulted in a significant number of publications and software artifacts. These are listed in the references at the end of this report.

3.1 Human Resources

We have roughly 15 graduate students, 10 undergraduate students, and 4 research scientists working in areas related to this project.

3.2 Research and Education

The research artifacts from the project are finding their way into graduate courses. Professor Ramachandran taught a special topics course entitled, "Pervasive computing with distributed sensors," (URL: `www.cc.gatech.edu/classes/AY2002/cs8803e_spring`). Some of the students in the course used D-Stampede as an implementation vehicle for the course project.

Further we are in the process of establishing a *sensor lab* via funding from an NSF RI award. This lab will serve both the research agenda of this ITR award as well as curricular development both in the College of Computing and School of Electrical and Computer Engineering at Georgia Tech.

4 Special Requirements

The total budget for this 5-year proposal is $1.35M. However, due to fiscal constraints NSF chose to front-load the total budget by awarding $750,000 in the first year. The second year increment was $200,000. The understanding with the program manager (Dr. Helen Gill) is that our spending plan for the award will be more balanced over the 5-year period despite the front-loaded nature of the allocation. Hence, we have over $500,000 still remaining from the first two years of allocation.

References

[1] Sameer Adhikari, Arnab Paul, Bikash Agarwalla, and U. Ramachandran. D-stampede: Distributed programming system for ubiquitous computing. Technical report, CERCS, Gerogia Inst. of Technology, 2003. Expanded version of a paper that appeared in the 22nd International Conference on Distributed Computing Systems (ICDCS), to be submitted to IEEE TPDS.

[2] J.B. Fryman, C.M. Huneycutt, H.S. Lee, K.M. Mackenzie, and D.E. Schimmel. Energy efficient network memory for ubiquitous devices. Technical Report GIT-CERCS-03-05, CERCS, Gerogia Inst. of Technology, 2003.

[3] J.B. Fryman, H.S. Lee, C.M. Huneycutt, N.F. Farooqui, K.M. Mackenzie, and D.E. Schimmel. Softcache: A technique for power and area reduction in embedded systems. Technical Report GIT-CERCS-03-06, CERCS, Gerogia Inst. of Technology, 2003.

[4] Joshua Fryman, Chad Huneycutt, and Kenneth Mackenzie. Investigating a softcache via dynamic rewriting. In *Fourth Workshop on Feedback-Directed and Dynamic Optimization (FDDO)*, 2001.

[5] Chad Huneycutt, Joshua B. Fryman, , and Kenneth Mackenzie. Software caching using dynamic binary rewriting for embedded devices. In *International Conference on Parallel Processing 2002*, August 2002.

[6] Rajnish Kumar, Matthew Wolenetz, Bikash Agarwalla, Jun-Suk Shin, Phillip Hutto, Arnab Paul, and Umakishore Ramachandran. Dfuse: A framework for distributed data fusion. Technical report, CERCS, Gerogia Inst. of Technology, 2003. Submitted to ACM SenSys ' 03 , Los Angeles,CA, Nov 5-7,2003.

[7] H.S. Lee, J.B. Fryman, A.U. Diril, and Y.S. Dhillon. The elusive metric for low-power architecture research. In *Proceedings of the Workshop on Complexity-Effective Design*, 2003.

[8] Hasnain A. Mandviwala, Nissim Harel, Kathleen Knobe, and Umakishore Ramachandran. Distributed garbage collection for timestamped data. Technical report, CERCS, Gerogia Inst. of Technology, 2003. This is an expanded version of two papers that appeared in earlier conferences, ICPP 2002 and LCPC 2002, to be submitted to IEEE TPDS.

[9] Arnab Paul, Sameer Adhikari, and U. Ramachandran. An implementation of a cluster runtime system for streaming and multimedia applications. Technical report, CERCS, Gerogia Inst. of Technology, 2003. To be submitted to Software Practice & Experience.

[10] Arnab Paul, Nissim Harel, Sameer Adhikari, Bikash Agarwalla, Umakishore Ramachandran, and Ken Mackenzie. Interaction between stampede runtime and operating systems. In *2003 IEEE International Symposium on Performance Analysis of Systems and Software*, March 2003.

[11] Umakishore Ramachandran, Phillip Hutto, Bikash Agarwalla, and Matthew Wolenetz. A system architecture for distributed control loop applications. In *Workshop on Future Trends in Distributed Computing Systems*, May 2003.

[12] Umakishore Ramachandran, Rishiyur S. Nikhil, James Matthew Rehg, Yavor Angelov, Arnab Paul, Sameer Adhikari, Kenneth Mackenzie, Nissim Harel, and Kathleen Knobe. Stampede: A cluster programming middleware for interactive stream-oriented applications. Technical report, CERCS, Gerogia Inst. of Technology, 2003. Submitted to IEEE TPDS special issue on middleware.

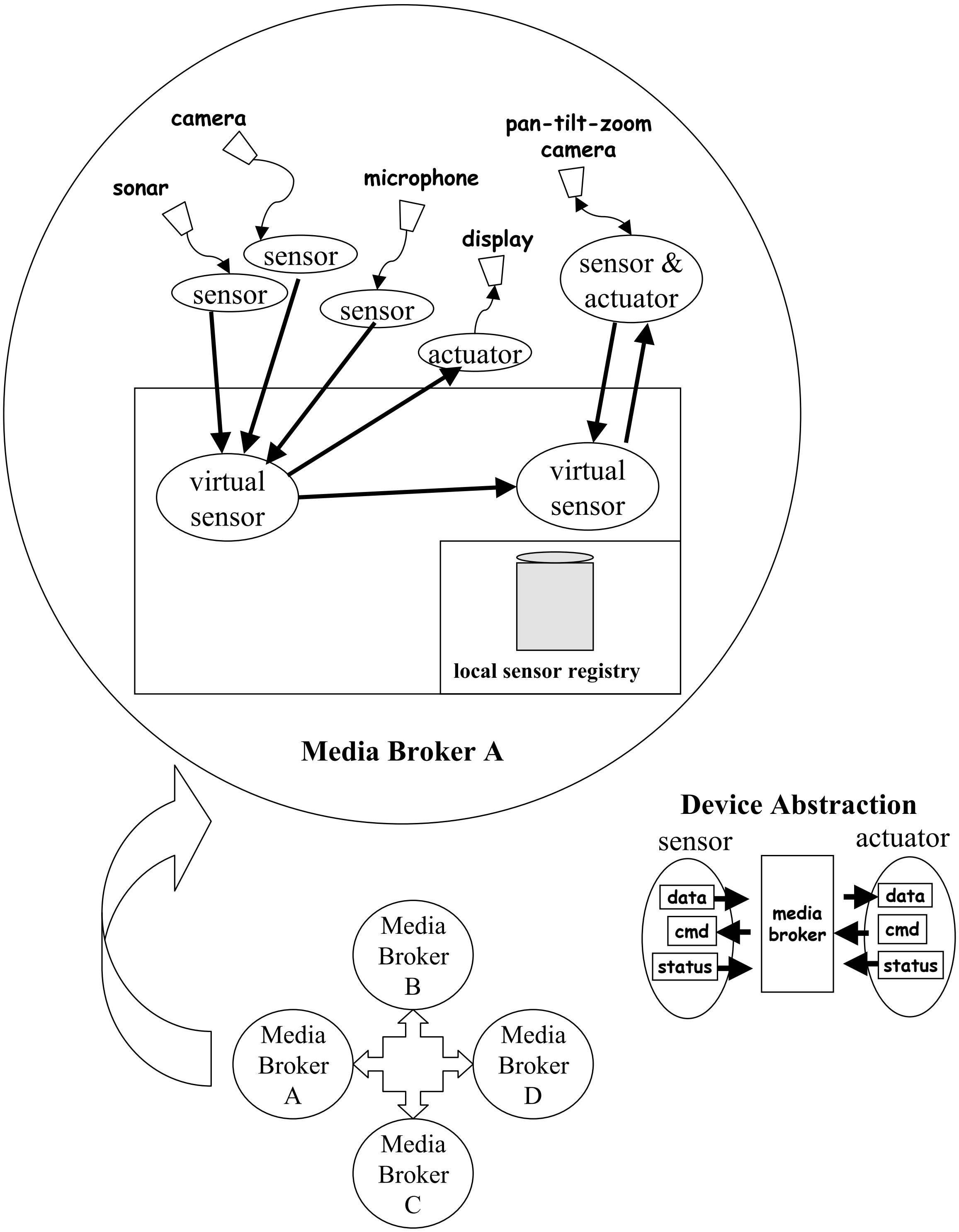

Figure 1: Media Broker Architecture