Situation Awareness

Umakishore Ramachandran
College of Computing
Georgia Institute of Technology

1. Application Scenario

A person is running up the escalator the wrong way in a busy airport. There is excessive commotion in a public place. A traveler says a "forbidden" word in a high security area. An automobile is making some suspicious moves in a public parking lot. An automobile runs a red light in a busy intersection during rush hour.

How do we know which of the above statements are isolated events and which of them are connected in some fashion (to one another and perhaps to other such events that happened in the past) and could lead to an "incident" that law enforcement officials should care about beyond just a misdemeanor? These are questions that become more and more relevant for societal safety in our ever changing complex life. There is an enormous amount of data that currently gets generated today, and there are enormous amounts of archival information on *incidents* and *events* leading up to them. This is only going to get worse with the proliferation of sensors (such as cameras) becoming part of the environment. Needless to say that these events need not necessarily be happening at the same time or at the same location. The problem is one of deriving accurate higher level knowledge out of the myriad of data and information sources (some of which may be faulty, error prone, and/or compromised) both live and archived leading to *situation awareness*.

There are three communities -- *sensor networks*, *pervasive computing*, and *high performance computing* – that carry out research that are relevant to achieving situation awareness in a scalable and timely manner. The main limitations of today's systems to achieving situation awareness are:

- Lack of a coherent vision for bringing together the above three communities to solving the problem
- Centralized (and hence non-scalable) solutions to information acquisition, processing, and alert generation and dissemination
- Lack of resource-conscious solutions for data fusion from several information sources and meeting real time constraints

2. A Concrete Example – Airport Surveillance System

Just to make the discussion concrete, let us assume an airport as the public place, and a *distributed surveillance system* as a canonical example of a cyber-physical system. Imagine a bank of cameras, microphones, and other sensors deployed in the entire airport. Figure 1(a) shows today's centralized way of dealing with such a bank of cameras wherein humans watch a bank of monitors corresponding to the various areas.

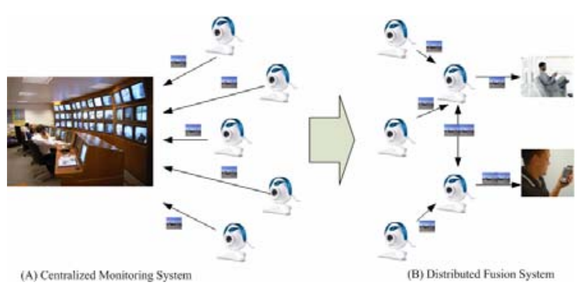

Figure 1: Moving from a centralized to a distributed solution.

Figure 1(b) shows the vision for a distributed solution. Data from the sensors (e.g., video from the cameras) are filtered, processed in the network for suspicious activities and *selected knowledge* streams (e.g., events of interest, video footage, audio clips, etc.) are delivered simultaneously to multiple destinations in a manner best suited to the specific sink (e.g., a database for later forensic analysis, an automatic alarm system, the PDA of a roving security agent, or the desktop of an officer). The knowledge streams result in actuation (e.g., an alarm goes off) that could lead to requiring changes in the information flow through the network. Thus there is a *sense-process-actuate control loop* that characterizes this cyber-physical system. At a cognitive level, this control loop represents a transformation of *data* to *events* to *actionable knowledge*.

To realize this vision, there is a camera network deployed throughout the airport. The cameras are part of a sensor network that is comprised of sensors, compute nodes, and actuators (such as alarms and PDAs) distributed throughout the airport. The nodes may be connected either through a wired or wireless network. Figure 2 shows a deployment with just two cameras. In this figure, there are both low bandwidth sensors (e.g., light and sound) and high bandwidth sensors (e.g., cameras). Logically, there are two networks: *control* and *data*. The control network acts as a vehicle for triggering the data network. For example, the low bandwidth sensors may form the control network, while the cameras form the data network in the figure. Of course, it is completely reasonable for the control network to be realized using the cameras themselves with local processing on the attached processors to discern activity. Based on the intelligence gathered from the control network, cameras are selectively turned on, the video delivery rate is throttled (time-lapse vs real time), video streams are processed en route in the network (if necessary, corresponding to the scenarios identified earlier), and either the raw video or the digest of the processing delivered to the various sinks that we mentioned earlier.

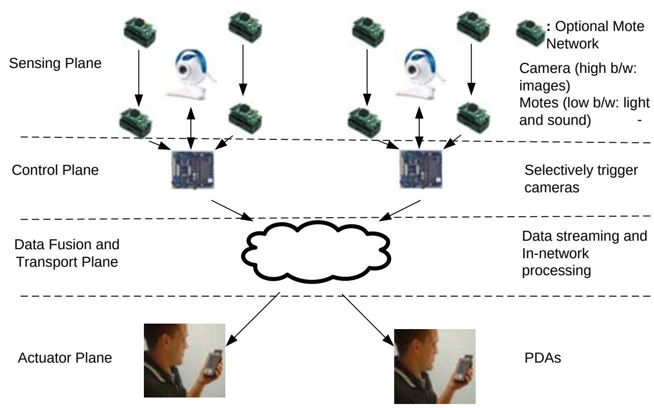

Figure 2: A heterogeneous sensor network

Figure 2 breaks down the deployment into several logical layers. The sensing plane is self-explanatory. There is a need to have redundancy at this level (for example multiple cameras and microphones covering a given area) since sensor failures (permanent or temporary) could result in loss of situation awareness. Further, such redundancy also helps in triangulation and pin-pointing the location of any untoward activities. The control plane consists of the embedded compute nodes that implement the control network. The data fusion and transport plane serves to fuse the data streams from different cameras to create an information digest for dissemination to actuators and transport the streams through the network. The actuator plane consists of the sinks (e.g., end devices such as PDAs as well as actuators such as alarm systems) that are connected to the data network. The actuator plane may also serve as a query plane in that an agent (human or software) may be able to pose specific queries that may trigger some action in the data network (e.g., "show me a collage of images from two cameras X and Y"). In this sense, the system adapts to function in either *PUSH* mode to send timely alerts, and/or in *PULL* mode to satisfy targeted queries.

Two things stand out from the above discussion. First, this example is a microcosm of similar needs for situation awareness in a battlefield, in urban warfare, disaster recovery, and emergency response. Second, while it seems like all of the above should be doable today, there are a number of research challenges that impede realizing the vision. The main challenges as we see in achieving situation awareness are:

- Reducing the load on the infrastructure (both computation and communication) to deal with the massive amount of real time data that is continually generated

- Reducing the cognitive load on any human in the loop (e.g., security personnel in an airport) by providing them *selective attention* to highest priority events
- Prioritizing the information flow in the system continually to adapt to dynamic changes in the observed events and the queries posed by humans and/or software agents

3. Directions for Realizing the Vision with an Expected Timeline

1. Realizing a baseline distributed system for situation awareness (3 years)
   - *Selective Attention*: The control network allows selective triggering/throttling of high bandwidth streams based on activity in the region. Managing the control and data network and overlaying them on the available physical network (both wired and wireless) is one possible research direction. Inherent in this pursuit are capabilities for detecting and eliminating *false positives* (e.g., a false alarm) and *false negatives* (e.g., missed hits) from the control network, as well as triangulation to pin-point location of commotion.
   - *In-network processing and data fusion*: As data flows through the network, the streams are analyzed to derive *situation awareness*. Managing the deployment of application-specific code to analyze the streams, migration of the code among the compute nodes to optimize quality metrics (such as energy), and fusion of multiple streams to derive higher level situation awareness (and thus reduce cognitive load) constitute another possible research direction.
   - *Information Prioritization*: Another interesting direction of research is designing a *Priority-aware Overlay Network (PON)* that takes into account the dynamics of the application and the information becoming available from the data sources to dynamically adjust the priority of the information flows in the network that will lead to better situation awareness.
2. Improving robustness, performance and real-timeliness (5 years)
   - *Ubiquitous High Performance Computing*: The above vision assumes that high performance computing resources are available ubiquitously for stream processing to aid in situation awareness. Integrating Pervasive Computing, Sensor Networks, and High Performance Computing is another important avenue for future research.
   - *Component Failure*: Failure (either malicious or natural) has to be dealt with graciously without bringing the system down. Dealing with component failure (hardware or software) as a fundamental design goal is another important research direction.
3. Achieving a trustworthy system with 99% accuracy of situation awareness (7 – 10 years)
   - *Data Integrity, Infrastructure Security, and Information Privacy*: There needs to be two-way authentication of the information sources and the infrastructure and this is another important research direction.
   - *Advanced Media Processing Algorithms*: Deriving meaningful *events* from data and *actionable knowledge* from events are two important avenues of future research as well. While significant progress has been made in areas such as computer vision, there is still a considerable amount of work to be done to analyze multi-modal stream data for higher level inference.

4. Bio

Umakishore Ramachandran received his Ph. D. in Computer Science from the University of Wisconsin, Madison in 1986, and is currently a Professor in the College of Computing at the Georgia Institute of Technology. For two years (July 2003 to August 2005) he served as the Chair of the Core Computing Division within the College of Computing. His fields of interest include parallel and distributed systems, computer architecture, and operating systems. Currently, he is leading an NSF-ITR funded project investigating the programming idioms and runtime systems for a distributed sensing infrastructure. He is the recipient of an NSF PYI Award in 1990, the Georgia Tech doctoral thesis advisor award in 1993, the College of Computing Outstanding Senior Research Faculty award in 1996, the College of Computing Dean's Award in 2003, and the College of Computing William “Gus” Baird Teaching Award in 2004. (E-mail: rama@cc.gatech.edu; Phone: (404) 894-5136; URL: www.cc.gatech.edu/~rama).