Towards a Theory of Events[*]

K. Mani Chandy
California Inst. of Technology
mani@cs.caltech.edu

Michel. Charpentier[†]
University of New Hampshire
charpov@cs.unh.edu

Agostino Capponi
California Inst. of Technology
acapponi@cs.caltech.edu

ABSTRACT

Event-driven systems are used in a wide range of applications such as responding to missile attacks, interdicting potential terrorists, exploiting arbitrage opportunities and responding to congestion in supply chains. The designs of event-driven systems vary widely because the costs and benefits to users of different applications are markedly different. This talk proposes a framework for unifying designs of different types of applications by representing the design problem as a constrained optimization and by defining interaction between components in distributed event-based systems in terms of a concept called "shared models." This talk is intended to suggest that there are concepts that unify analyses of a range of event-driven systems in nature, human social organizations and information technology systems.

Categories and Subject Descriptors

H.1.1 [**Models and Principles**]: Systems and Information Theory

General Terms

Design, Theory

Keywords

Events, Event-Driven Architectures, Sense-and-Respond Systems, Shared Models, Cost/Benefit Analysis, Optimization

1. SENSE-AND-RESPOND SYSTEMS

This paper proposes ideas that we believe help in developing a theory of events. Systems that are event-based fall within the class of "sense-and-respond systems." We begin by discussing sense-and-respond systems and then introduce ideas that help in developing a unifying framework for designing and analyzing such systems.

Basic Concepts. A *system* operates within an *environment*. The environment for a company includes its customers, suppliers, government, and markets. In this paper, we use the term *universe* for the system coupled with its environment.

An *event* is a significant change in the state of the universe. A *significant state change* is one for which an optimal response by the system is to take an action. An insignificant state change is one for which the system need take no action. An action may be registering information about the event in the enterprise's memory. Insignificant state changes are not registered in memory; they are never "remembered."

A *sense-and-respond* system is one that responds to events. Responses may be proactive. A proactive response is an action taken by the system in anticipation of a probable critical change in state of the environment. An example in a hurricane warning system is sending messages to schools warning them of a possible incoming storm. All social organizations and biological systems are sense-and-respond systems to varying degrees.

Ubiquity of Sense-and-Respond Systems. A stork pounces on a frog when it senses movement in the reeds, and the frog jumps away when it senses a predator. A frog's senses tell the frog's nervous system when it detects threats or opportunities and the frog responds. Animals that respond inappropriately die. A living being that does not detect threats gets killed. Likewise, one that frequently takes unnecessary action wastes energy and perishes.

Social organizations are sense-and-respond systems. Prides of lions, packs of hyenas, and herds of zebras on the savannah are interacting social organizations, each of which is a sense-and-respond system. Lions in a pride signal others about their actions. Likewise, zebras respond collectively when they detect state changes that merit action.

All human societies are sense-and-respond systems. Therefore, it is no surprise that sense-and-respond systems have been developed since the dawn of civilization. Abstract concepts about events have been proposed by several communities of researchers including social scientists and engineers. Information technologies to support sense-and-respond organizations have been developed in several fields including automatic control, distributed systems, messaging systems,

[*]This work is supported in part by AFOSR MURI award FA9550-06-1-0303.

[†]On sabbatical leave at the California Inst. of Technology.

DEBS '07, June 20–22, 2007 Toronto, Ontario, Canada

databases, business intelligence and sensor networks.

DEBS—distributed event-based systems—are information technologies that support sense-and-respond enterprises. Likewise, EDA—event-driven architecture—is a software architecture pattern for sense-and-respond systems. Literature on event-driven architectures and sense-and-respond systems can be found in several web sites including [16] and [12]. The proceedings of workshops on EDA and DEBS identify a part of the literature.

Steve Haeckel [12] is a pioneer in the study of managerial sense-and-respond systems. He studied companies that were successful in the past and companies that continue to be successful now. Companies that were successful in the past developed long-range plans carefully, and then executed these plans efficiently but somewhat rigidly. Companies that continue to be successful today are flexible organizations that adapt rapidly to changing situations; such companies are sense-and-respond systems.

A Unifying Framework. We attempt to develop a common framework for analyzing sense-and-respond systems ranging from prides of lions to human societies to DEBS. The attempt is not preposterous because it merely seeks to apply concepts and theories developed over many decades in areas including feedback control systems [20, 2, 19] and operations research—particularly Bayesian decision theory, optimization, classical statistics and game theory [18]—and organizational structure [12].

We seek to understand answers to questions such as: What information should one component of a distributed sense-and-respond system send to another? When should the information be sent? What are protocols by which agents determine when to interact? How do birds in flight coordinate their actions so that they fly in formation? And is that related to the design of baggage-handling systems at airports?

The key question is: What action, if any, should an agent in a sense-and-respond system take at a given instant? The action depends on the estimates that the agent has about the state of the environment and the states of other agents. High-value information improve the agent's actions. Low-value information doesn't change the agent's actions or results in poorer actions than no information at all.

A fundamental principle in designing sense-and-respond systems is to get high-value information to agents subject to the constraints such as limitations on bandwidth, computational capacity and energy. We propose two ideas to help implement this principle: 1) shared models for specifying interaction between agents, and 2) representing sense-and-respond systems as constrained optimizations.

Constrained Optimization: When there are no limits on resources such as communication bandwidth, computation, and energy the design of sense-and-respond systems is straightforward: send all the information all the time to a central processor that does all the necessary computation and communicates continuously with all responders. When there are constraints, both the problems of designing and running sense-and-respond systems are constrained optimization problems. The problem of design is that of acquiring resources such as bandwidth and computational power. The problem of running a system is that of sharing fixed resources among competing requirements. Constraints in the design process include the overall budget and types of components, such as types of sensors, that are available. Constraints in the execution process include the given network, computers, sensors and responders.

Shared Models: The value of a piece of information to an agent (a component) is the difference in benefits between actions the agent makes *with* the information and actions made *without* the information. How does one agent tell another what is high-value information? How does the value of information change with time?

Consider a problem faced by a mayor of a city in dealing with natural disasters such as hurricanes. The decisions that the mayor has to make, when a hurricane is detected several miles out to sea, include whether to ask state or federal government for additional resources and whether to order an evacuation of different parts of the city. A piece of information is valuable to the mayor if and only if the mayor makes better decisions as a consequence of receiving the information. What information should be sent by agencies to the mayor, and when?

Consider two streams of information to the mayor: one from the engineer in charge of the pumping system and the other from the police chief. The mayor expects to be informed if the pumping system seems to be failing or if the security situation is not normal (such as signs of rioting or unusual numbers of traffic accidents). These expectations can be described as models shared between the mayor and the engineer, and between the mayor and the police chief.

The mayor expects that when reality—as determined by the engineer—matches the model that indicates a probable failure in the pumping system then the engineer will proactively inform the mayor. The message may be as small as a single bit indicating that reality matches the model: the pump system is failing. More likely, the message will include additional information such as which pump is failing.

The mayor, likewise, shares a model with the police chief. This model defines what is normal in the city. The mayor expects to be informed proactively when reality—as determined by the police chief—deviates from the model representing normality. The absence of messages from the police chief conveys information that the security situation in the city is not abnormal.

In the former case communication takes place when reality matches the model. In the latter case communication takes place when reality deviates from the model.

The shared model may not be formal. Generally the model is not in terms of a predicate or a SQL query. Often the model is learned and tuned over time. The shared model is a helpful concept in understanding sense-and-respond systems.

2. THE PROBLEM DOMAIN

This section describes the problem domain we study. We propose a unifying framework for this domain. The domain is described in terms of the following characteristics: 1) components of distributed sense-and-respond systems; 2) modes of interaction between components; and 3) constraints and objectives of sense-and-respond systems.

Distributed Nature of Sense-and-Respond Systems. Sense-and-respond systems consist of four types of components:

- *Sensors* that obtain estimates of parameters of the state of the environment.
- *Processing agents* that carry out computations that change the state of the system itself.
- *Responders* that carry out actions that result in changes to the state of the environment.
- Information *dissemination networks* that transmit information between sensors, processing agents and responders.

The sense-and-respond system in an animal is distributed: eyes and ears provide estimates of the state of the animal's environment; neurons in the brain and spinal chord integrate data from sensors with memory; muscles are responders that react to nervous signals; and the neural and chemical pathways form the dissemination network.

Modes of Interaction between Components. There are three ways in which components of sense-and-respond systems interact with each other:

- **Schedule**: Groups of components interact at scheduled times.
- **Pull**: A component requests information from other components, which then reply to the requests.
- **Push**: A component sends information to other components when it discovers state changes relevant to its listeners.

A unifying framework for designing sense-and-respond systems must deal with all three types of interaction.

Objectives and Constraints. The objective, as in optimal control systems, is to maximize the benefits of responses to events over lifetimes of systems. The benefits of responses depend on the application. We begin with a simple model from classical statistics and then consider more complex models. We use systems that respond to natural disasters as illustrations.

A *false positive* is execution of an action by a system in response to an anticipated or perceived state change in the environment that does not occur. Clearing a beach front in anticipation of a tsunami that does not arrive is an example of a false positive. A *false negative* is the absence of responses to state changes that require responses. A tsunami hitting an area without warning is an example. The costs of false positives and negatives depend on the states of the system and the environment. An objective function is to minimize a combination (such as the amortized sum) of the costs of false positives and false negatives over a system's lifetime.

Better objective functions specify benefits of responses as functions of timeliness. Tsunami warnings to people on beachfronts 10 minutes before tsunamis strike are more valuable than 10-second warnings. Even better objective functions specify accuracy in terms of gradations and not merely in terms of binary false/true characterizations. A category-5 storm warning before a category-4 storm hits is a false positive; however, a better characterization is in terms of the costs of inaccuracy. Costs increase with the degree of inaccuracy: a category-5 warning when no hurricane hits is more expensive than the same warning when a category-1 storm hits; and this, in turn, is more expensive, than the same warning prior to a category-2 hurricane.

A design task is to build, maintain or modify sense-and-respond systems to maximize their benefits subject to constraints. A critical constraint is the overall cost. Other constraints deal with the kinds of available sensors, responders and processing agents.

3. INTERACTION AND SHARED MODELS

In this section we focus on what information should be communicated, which protocols to use for communication and how the information should be analyzed to determine responses. A graphical illustration of these three different communication pattern is depicted in fig. 1. The thick-line curve represents a parameter (or function of the state) that is being transmitted from one component to another. The circular dots are communication events. In the first model (schedule), these are spaced regularly on the time axis as they are triggered by time delays. In the case of pull-based communication, communication events are triggered by external requests (queries) represented here as pointing bolts. In the last model—push-based communication—they occur when actual measurements differ enough from the expected behavior of the shared model, depicted as a thin-plain line. In this example, the shared model is a model of normality.

3.1 Modes of Interaction between Components

The three modes of interaction—synchronized communication, pull and push—can be evaluated within the framework of constrained optimization.

Schedule. Most organizations employ some form of scheduled synchronized communication. CEOs meet with their vice-presidents every Monday morning or at prespecified intervals. Research advisors meet with their students at specified times. There are several advantages to scheduled communication including the following:

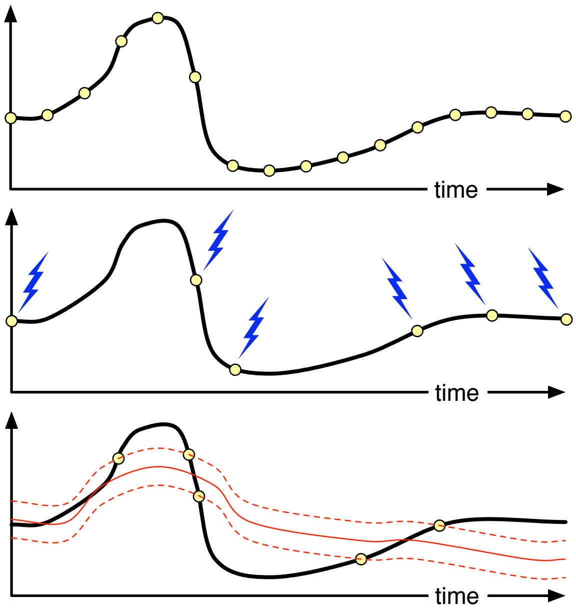

Figure 1: The three different communication patterns: scheduled-based, pull-based, and push-based

- **Heartbeat**: A component may fail. A synchronized communication serves as a heart-beat mechanism. If a component does not participate in an interaction, other components may infer that it has failed.
- **Cost of listening**: Some wireless devices require energy to listen continuously even when there is no information being transmitted. Synchronous interaction allows components to go dormant between planned interactions thus saving energy. This problem is less acute with devices with two levels of listening—low-energy passive listening for signals that trigger a mode change to higher-energy active listening for information. Low-energy listening is limited to getting the single signal that switches the component to active listening.
- **Group meeting**: Scheduled meetings allow groups to be formed, which can result in more effective communication. The same piece of information can be shared by all the components that participate in a scheduled meeting.

Pull. Consider the following example. In the event of a crisis, such as a toxic plume emanating from a factory accident, the state's emergency controller organizes all governmental and private ambulances to rescue the most people. The data required to mount an effective response includes information about the accident, the direction of the plume, and the locations of several hundred ambulances. This data could be acquired by a combination of scheduled interactions, push, and pull.

Assume that the average rate at which such crises occur is once every five years and that the crisis lasts for a day. Each ambulance could send information about its location every T minutes to the crisis center. Alternately, the crisis center could request information about ambulance locations in the rare event that there was a crisis. Provided most ambulances respond to these requests within time T, the request-reply protocol is more efficient than the scheduled communication protocol.

Consider the communication pattern between the crisis center and the subsystem responsible for monitoring the environment. The crisis center could send a request to the environmental subsystem asking for its current status. But when should it do so? The crisis center doesn't "know" the status of the environment—accidents can happen at any instant—and the costs of tardy responses—thousands suffering from health ailments—are high. So, to poll the environmental subsystem is a possible policy for the crisis center as long as it does so frequently enough.

An alternate pattern is for the environmental subsystem to first push information to the crisis center, which can then poll for more information if needed. This pattern is discussed in the next paragraph.

Push. The environmental subsystem may include thousands of sensors organized in a hierarchical graph such as a tree. These sensors can be organized as a push network. Each sensor sends messages when its measurements exceed a threshold. Sensors may also combine this pushed information with heartbeat messages to inform others that they are alive. Events flow from the leaves towards the root. Nodes higher in the tree aggregate information and send events up the tree, towards the root, if the aggregated information indicates a critical situation.

Sensors may make readings every minute and generate events once a day on the average as their readings cross thresholds. Event rates decrease rapidly higher up the tree, and the environmental subsystem may only send events to the crisis center once a year. When the crisis center receives an alert event from the environmental subsystem, the crisis center requests location information from the ambulance subsystems.

Push technology requires precise shared models to determine what information to push. The provider and consumer of information have a shared model; when reality, as measured or inferred by the provider fits the model then the provider sends a message to the consumer. As long as reality doesn't fit the shared model, the provider sends no information.

The consumer gains knowledge from the *absence of information*. If the consumer hasn't received messages then the consumer infers that reality, as determined by the provider, doesn't match the shared model.

A mathematical analysis of push networks is nontrivial when messages may be delayed between providers and consumers of information, when providers may die, or when measurements or analysis carried out be providers may be erroneous [7]. The work on the design of push networks in [11] proposes to use model-driven techniques for querying sensor networks. Such models provide a framework for optimizing the acquisition of sensor readings; sensors are used to acquire data only when the model is not rich enough to answer the query with acceptable confidence. Performance metrics are resource consumption, delay when answering queries and communication costs [21]. For instance, if a sensor is far from the query source, then the query source cannot acquire the reading from it without forwarding the request to other nodes.

3.2 Shared Models

What information should a producer give to a consumer? Traditional ways of describing the information transferred is in terms of responses to queries. In the "pull" communication pattern, a client (the consumer) executes a query on the server's (producer's) state, just as a database user may execute a SQL query against a database. In the "push" communication pattern, a client subscribes for information with a server (publisher) by posting a subscription or a persistent continuous query. In the schedule pattern, participants execute transactions on their combined state at pre-specified instants; these transactions can also be represented as queries and updates. We suggest that a "model" is a better representation than "query" for describing the information that is exchanged.

Shared models help in evaluating alternative designs of dissemination networks. The specificity of the shared model determines, in part, the amount of information that needs to be shared across multiple agents. Consider, for example, an intrusion detection system in which sensors of different types are distributed on fences and walls, open outdoor areas, and in buildings. Designers may choose among different shared models by which sensors (such as accelerometers) on fences report information.

The simplest shared model is no model at all: the accelerometer continuously sends its analog readings to other agents that analyze the information. A sophisticated shared model is one in which the sensor evaluates accelerometer readings to determine if there is any activity, and then categorizes the activity as normal (wind, rabbits, snowfall) or abnormal (possible intruder) or assigns probabilities to different possibilities.

The communication pattern may be by schedule, push, or pull. A simple shared model communicating by schedule sends accelerometer readings every T seconds. A sophisticated model using the push pattern sends signals only when a probable intruder is detected with probability greater than some threshold (say 1%). When an intruder is detected, the information that the sensor sends may be minimal—merely the identity of the sensor that located the intruder—or it may include details such as the accelerometer readings over the last 5 minutes. The communication bandwidth and energy requirements for different designs are very different.

We discuss a few examples and then return to the issue of models versus queries. Over time, models of patterns of signals in normal operation are constructed for different subsystems. For example, the fence subsystem may have models of patterns signaling deer or cattle moving up to the fence. The intra-fence subsystem may have models of signals generated during heavy snow or when foxes run in the area. The subsystems generate events when reality as measured and inferred by a subsystem deviates from the model of what is "normal." In this example, the model that identifies the anomalous condition is the complement of the model that defines normality.

Models that define possible tsunamis deal with time patterns of wave heights. Models that specify possible hurricane landings are complex simulations that evaluate likely trajectories. These models, as opposed to models in the intruder interdiction system, specify critical conditions rather than complements of critical conditions.

In the "push"' communication pattern, a subscription is a model shared by the producer (publisher) and consumer. Likewise, in the "pull" and "schedule" patterns, requests are for items of information that support or invalidate models shared by producers and consumers.

For example, consider a scheduled communication in a hospital, such as doctors meeting with nurses, residents and interns in morning rounds. The information that the doctor expects to get from night nurses is based on shared models of medicine and health.

The information shared by producer and consumer can be defined in terms of queries provided the concept of a query is extended to include requests for information that validate or invalidate models. The conventional view of queries is, however, primarily in terms of SQL-like operations on tables or on windows in data streams. This is too limiting in our view.

The shared-model description of the information that is exchanged is helpful in analyzing sense-and-respond systems. An agent that is responsible for initiating a response takes action based on its estimates of the state of the universe. Its estimates are based on a model of the universe which, in turn, is based on models that it shares with agents that provide it with information. All sensing, communication, computation and responses can be defined in terms of shared models.

4. CONSTRAINED OPTIMIZATION

The tasks of designing and running sense-and-respond systems can be represented as constrained optimization problems. In this section, we discuss various constraints and costs to be taken into account.

4.1 Communication and Computation Costs

Continuous communication and computation are optimal when communication and computation carry no cost. Consider a flight of birds where the goal is for all birds in the flight to head in the same direction in a formation. Communication between birds is achieved by sight and sound. A bird can see its neighbors, and the cost of seeing is negligible; therefore birds keep their eyes open while in flight. Each bird determines its heading based on the headings that it observes of neighboring birds in the flight. "Computing" its direction carries no cost; therefore a bird's nervous system continuously adapts its heading to a weighted average of its neighbors' headings. Bird calls do take energy; so these aural forms of communication are not used continuously.

Evaluating possible architectures of information dissemination networks is one of the first steps in the design of sense-and-respond systems. A simple solution is for all sensors to send measurements periodically to a central site that aggregates all the information and determines optimal responses, and then actuates responders. The flow of information follows a star pattern: from many sensors to a central point and then from the central point to multiple responders. The star network is appropriate for systems in which volumes of communication are low or in which the costs of long-range communication are negligible; however, most systems in nature and society are not star networks.

Organizations, such as hospitals, are organized in complex and fluid networks of sensors, databases, computers, actuators, and humans in different roles. Information doesn't flow along fixed networks using a single communication pattern (such as schedule, push, or pull). Though information generally flows in a hierarchical fashion—from sensors on patients in hospital wards to nurses in charge of wards, to residents, to doctors—any agent may request information from any other when necessary. Many systems including airplane baggage handling, supply chain and military systems have underlying networks organized in roughly hierarchical fashion, but allow modifications to the flow when necessary.

A constrained optimization framework, using shared models as the mechanism for communication, helps in evaluating alternatives. The constraints include bounds on: rates of power consumption in computing and communicating for sensors that use batteries, interference between multiple agents communicating by wireless to a remote central site and computing capacities at each node.

The specificity of models determines the amount of computation and communication. A model that identifies a fence-crossing intruder in heavy snow in the arctic circle is different from one deployed in the desert. Models are tuned over time as increasing experience helps distinguish normal from abnormal patterns. Non-specific models—such as report any shaking of the fence—result in more communication and more computation at central sites.

The goal of this section is merely to suggest that the shared-model and constrained optimization ideas can help provide a framework for systematic analyses of design alternatives.

4.2 Costs and Benefits of Responses

This subsection discussed examples of sense-and-respond systems with different costs of false positives, false negatives, inaccurate responses, and tardy responses. The subsection suggested how the different costs impact designs. The design of sense-and-respond systems is determined in large part by the benefits of timely appropriate responses in comparison with the costs of inappropriate or tardy responses. We use three illustrative examples—tsunami warning systems, hospital emergency care, and accounting—with very different cost/benefit profiles.

Tsunami Warning Systems. The cost of a non-response to a tsunami can be horrendous. The benefit of an appropriate warning decreases with the gap between the instant at which the warning is issued and the instant at which the tsunami strikes. The cost of an inappropriate warning—clearing beach areas when no tsunami hits—is also high.

The public accepts warnings of say 4-meter waves as reasonable even if wave height is only 2 meters. The cost of a warning of 4-meter waves when waves of height h strike increase as h decreases.

Tsunami-warning systems have high costs of erroneous responses. The design is also influenced by the shape of the cost of a warning as a function of the interval between the warning and the tsunami strike. Earthquakes off Indonesia or the US Pacific Northwest can result in tsunamis hitting coastlines in Indonesia or Washington and Oregon states in tens of minutes, whereas the same faults may result in tsunamis in Sri Lanka or Hawaii only hours later. These costs dictate the optimal locations of sensors and communication patterns.

Hospitals. The cost of a non-response (false negative) in a hospital can be the cost of a patient's unnecessary death. Costs of appropriate tardy responses are also high. An example of a false positive is an alarm turned on by a sensor on a patient when the patient is not in danger. The costs of false positives depends, in part, on the responses.

The cost of a false positive for a cardiologist who takes care of large numbers of patients is the cost of being woken up at home and driving to hospital to deal with a normal, non-critical situation. The cost for a resident at the hospital who oversees several wards is the cost of (perhaps being woken up and) going to the ward to evaluate a patient who is not in trouble. The cost for a nurse in charge of a ward is the cost of walking down to the patient's bed and checking on the patient.

Hospital information flow is organized to maximize the benefits of optimal responses and minimize the costs of false positives, false negatives and tardy responses. A largely hierarchical flow of basic information using the "push" information pattern is coupled with more detailed information "pull" by doctors from nurses and residents who provide continuous monitoring.

Providers and consumers of information share models about when events should be generated and what information should be shared. Residents and nurses share models (expectations) of what situations merit alerts to residents; likewise, residents and attending doctors share models. Problems arise when the models of providers and consumers differ.

A discussion of hospital design, representing information flow in terms of shared models, and treating the overall problem as constrained optimization, is beyond the scope of this paper. Here, we merely want to show relationships between different costs of responses and structures of information flow.

Accounting. A mutual fund company has many funds with different investment strategies and different benchmarks managed by several fund managers. The company monitors compliance of each of its funds with respect to reporting requirements and company goals. When a fund deviates from its model, an alert is raised to a person responsible for evaluating performance reporting. The alert includes information about deviation from the model and points to locations in a business-intelligence datacube that help with analyzing the deviation. A person who gets the alert can deal with the alert, or if necessary escalate the alert up the management hierarchy.

A false negative occurs when a deviation isn't detected. The cost of a false negative includes penalties for violating compliance regulations and possible loss of customers. The cost of a false positive is similar to the case of hospitals: the cost depends on the role of the person who handles the false positive. The cost of an appropriate but tardy response decreases only slowly with delay. Unlike tsunami warnings and hospitals, delays of minutes or even hours don't carry huge penalties.

An optimal design of this sense-and-respond system is as a star network. Information about all activity is sent to a central site that executes complex models to determine if anomalies exist. The cost-delay function permits centralized solutions with significant computing to avoid false negatives. The cost-delay function also permits schedule-based communication and computation: execution of models periodically, say once an hour, is satisfactory.

5. ILLUSTRATION OF DESIGN DECISIONS

Before discussing design patterns, we summarize the key points of this paper. We represent interactions between agents in terms of shared models. Whether the communication pattern is by schedule, pull or push, producers of information determine what information to share based on models that they share with consumers of the information. In the ideal situation a producer and consumer of information share exactly the same model, and the model is tuned for the specific problem. In some cases, producers and consumers may use different models; producers communicate and determine communication content based on one model, while consumers expect communication based on a different model. This results in non-optimal communication.

The costs of non-optimal responses, the frequencies of responses and the costs of computation and communication determine, in large part, the design of sense-and-respond systems.

The network of information flow can be represented by dynamic directed labeled graphs in which the vertices represent agents and the edges represent information flow. The graph is dynamic because patterns of information flow may change. Each edge is labeled with the models used by the producer of information and the consumer of information along that edge.

A simple star network in which information from sensors is sent to a central site where decisions are made and then sent to responders is appropriate for certain problems. More often, complex networks organized based on the roles of participants, using different communication patterns, are employed.

Computation and communication are continuous in systems in which their costs are negligible. Examples from nature, such as movements of flights of birds and schools of fish, illustrate such systems. The dynamics of such systems can be defined in terms of differential equations. Each agent (e.g., bird, fish) continuously solves a local constrained optimization problem based on the local information it has. Continuous solutions of local optimizations results in the global system following optimal trajectories.

Star networks and periodic schedule-based communication and computation are appropriate for systems in which the cost of tardy responses do not increase rapidly with delay. The accounting example illustrates this situation in which data from all information sources is aggregated at a single point which then computes optimal responses and sends the computed values to responders.

The "push" pattern is optimal when events are relatively rare and where the costs of tardiness are significant. Hospitals and tsunami-warning systems are examples. Since events are rare, pushing event information is more economical than having responders poll multiple sensors and agents frequently. Many systems combine an initial warning based on information pushed to it with acquisition of relevant information based on information pull. Doctors are alerted about a patient in trouble by means of information push from sensors through nurses and residents; then they pull relevant information directly from agents and databases. In some cases, the optimal amount of information that is pushed is small compared with the information that is pulled later by decision makers.

Hierarchical patterns of information flow are efficient for systems in which costs of false positives vary significantly across different roles. Doctors oversee several residents each of whom oversees several nurses each of whom oversees several patients each of whom may have several sensors and emergency-call buttons. The optimal strategy is to permit relatively large numbers of false positives near the information sources and to whittle them down by intelligent filtering towards the top of the hierarchy where agents have larger responsibility.

Geographical patterns of information flow are optimal for systems in which sensors are distributed across wide geographical areas and in which responses are also distributed. Geographical patterns of information flow may also be organized hierarchically with city, county, state and federal offices. An intruder interdiction system is an example of a system with geographical grid-based or hierarchical patterns of information flow.

6. CONCLUSION

Most systems in nature, human society and technology are sense-and-respond systems. We believe that a common theory of events will be developed to understand the entire range of sense-and-respond systems. Central to the theory are the value of a piece of information and the cost of communicating that information. Information theory deals with the optimum way of encoding information so that it can be sent along a channel with bounded capacity. Event theory deals with what information to send, when to send it, and how (schedule, push, pull) to communicate.

The problems of designing and running sense-and-respond systems are constrained optimization problems. The optimizations tradeoff the benefit of supplying valuable information with the costs of computing and communicating that information. The value of a piece of information to an agent is the incremental benefit obtained as a consequence of better actions taken as a result of the information. Thus, the value of information depends on the role of the recipient and the recipient's state. Models shared by producers and consumers of information identify what information to exchange.

We believe there is an opportunity to develop a theory of events. Such a theory will be based on ideas studied over decades from control theory, decision theory, game theory, distributed computing, and the social sciences. This paper suggests possibilities for developing such a theory.

The literature on these subjects is vast. The references merely point to a few papers, books or web sites. Control theory for distributed systems is described in [4, 2, 19, 5, 20, 14]. The theory deals with sense-and-respond systems with continuous communication and computation. Sense-and-respond systems in biology are discussed in several papers including [6, 10]. Biological systems at all levels from bacteria to flights of birds are fascinating sense-and-respond systems. Recent papers [18] discuss the relationship between cooperative distributed control and game theory. A great deal of work has been carried out on sensor networks [21, 11] that are directly relevant to the design of sense-and-respond systems. Research that is directly relevant to sense-and-respond systems has been carried out in the area of data streams [3, 17, 1]. Commercial tools for event processing are described in the sequence of event-directed architecture workshops; links to these workshops can be found in [16]. An excellent source of pointers to work in management of sense and respond enterprises is [13]. Work in distributed computing on sense and respond systems include those pointed to from [8] and [15, 9, 7].

7. REFERENCES

[1] M. Abadi et al. The design of the borealis stream processing engine. *Second Biennial Conference on Innovative Data Systems Research*, 2005.

[2] K.J. Astrom and R.M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Manuscript, 2007.

[3] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data streams. *Invited paper in Proc. of the 2002 ACM Symp. on Principles of Database Systems*, 2002.

[4] D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and distributed computation: Numerical methods*. Prentice Hall, 1989.

[5] V.D. Blondel, J.M. Hendrickx, A. Olshevsky, and J.N. Tsitsiklis. Convergence in multiagent coordination consensus and flocking. In *Proceedings of the Joint forty-fourth IEEE Conference on Decision and Control and European Control Conference*, pages 2996–3000, 2005.

[6] J. Buhl, D. J. T. Sumpter, I. D. Couzin, J. Hale, E. Despland, E. Miller, and S. J. Simpson. From disorder to order in marching locusts. *Science*, 312:1402–1406, 2006.

[7] Agostino Capponi. Estimation problems in sense and respond systems. Master's thesis, California Institute of Technology, 2006.

[8] K. Mani Chandy. Infospheres. *http://www.infospheres.caltech.edu*, 2007.

[9] K. Mani Chandy and Michel Charpentier. Self-similar algorithms for dynamic distributed systems. In *27th International Conference on Distributed Computing Systems (ICDCS'2007)*, June 2007.

[10] Iain D. Couzin, Jens Krause, Nigel R. Franks, and Simon A. Levin. Effective leadership and decision-making in animal groups on the move. *Nature*, 433:513–516, February 2005.

[11] A. Deshpande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and Wei Hong. Model-driven data acquisition in sensor networks. *Proceedings of the 30th International Conference on Very Large Data Bases*, 2004.

[12] S. Haeckel. *Adaptive Enterprise: Creating and Leading Sense-And-Respond Organizations*. President and Fellows of Harvard College, 1999.

[13] S. Haeckel. Sense and respond. *http://www.senseandrespond.com*, 2007.

[14] A. Jadbabaie, J. Lin, and A.S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003.

[15] D. Luckham. *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Addison-Wesley Professional, first edition, 2002.

[16] D. Luckham. Complex event processing. *http://www.complexevents.com*, 2007.

[17] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. The design of an acquisitional query processor for sensor networks. In *SIGMOD*. ACM, 2003.

[18] J. Marden, G. Arslan, and J. Shamma. Connection between cooperative control and potential games illustrated on the consensus problem. In *Proceedings of European Control Conference*, 2007.

[19] R. Olfati-Saber and R. M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, September 2004.

[20] J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stocastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, September 1986.

[21] F. Zhao and L. Guibas. *Wireless sensor networks: an information processing approach*. Morgan Kaufmann, first edition, 2004.