A Spatiotemporal Model for Ephemeral Data in Pervasive Computing Networks

[1]Jonas Michel, [1]Christine Julien, [2]Jamie Payton, and [3]Gruia-Catalin Roman

[1]University of Texas at Austin, [2]University of North Carolina at Charlotte, [3]University of New Mexico
Email: {jonasrmichel c.julien}@mail.utexas.edu, payton@uncc.edu, gcroman@unm.edu

TR-ARiSE-2011-012

ARiSE
Advanced Research in Software Engineering

A Spatiotemporal Model for Ephemeral Data in Pervasive Computing Networks

[1]Jonas Michel, [1]Christine Julien, [2]Jamie Payton, and [3]Gruia-Catalin Roman
[1]University of Texas at Austin, [2]University of North Carolina at Charlotte, [3]University of New Mexico
Email: {jonasrmichel c.julien}@mail.utexas.edu, payton@uncc.edu, gcroman@unm.edu

***Abstract*—Pervasive computing evokes a vision of digitally-accessible environments with which applications and users interact in localized ways. In this vision, information is *ephemeral*: it is created, moved, stored, and deleted on-demand at rapid rates. Without a formal data model that enables the data itself to speak about its spatial and temporal bearings, it is difficult to build support for accessing an information-rich digital world in a general-purpose way. In this paper, we demonstrate the need for an expressive data model of the inherently ephemeral data in pervasive computing and propose the beginnings of such a model that explicitly tags information with spatial and temporal semantics. Our model is founded on *spatiotemporal trajectories*, which capture the spatial and temporal semantics of data and the phenomenon it represents. We further demonstrate both the need for and potential impact of a general-purpose expressive spatiotemporal data model using several use cases.**

I. Introduction

Pervasive computing entails a future of digitally-accessible environments, populated with digital resources and services, with which applications and users interact directly. Such environments are as dynamic as the real-world entities that occupy them; time passes, devices and their users are constantly in motion, social patterns evolve, data is moved, information is shared and expires. Technologies like sensor networks, RFID tags, smart objects [1], and smart phones capture real-world phenomena and generate enormous amounts of real-time sensory, contextual, and user data. This data is *ephemeral*: it is created, stored, and deleted on-demand at a rapid rate.

A key challenge in pervasive computing is capturing and managing information subject to the high levels of dynamics present in reality. Much work has simplified how applications access data (e.g., through query abstractions and publish/subscribe mechanisms). However, all of these implicitly assume the existence and availability of an information-rich digital environment, supported by some coalescence of mobile opportunistic and peer-to-peer networks. Even after years of research on middleware, several questions remain, limiting pervasive computing capabilities: *Where does data come from? How is it stored? How does it move? And how does it die?* Without a formal data model that enables the data itself to speak about its spatial and temporal bearings, it is difficult to build support for populating, accessing, and interpreting an information-rich digital world in a general purpose way.

This material is based upon work supported by the National Science Foundation under Grant No. 844850. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation (NSF).

Fully realizing the vision of pervasive computing requires a general-purpose data model that (*i*) acts as the common glue between resources and services; (*ii*) enables data to exploit contextual dependencies; and (*iii*) reduces the responsibilities of developers that create applications that produce or consume data. Space and time and their inherent dynamics must be first-class citizens of such a model. In this paper, we demonstrate the need for a general-purpose data model capable of capturing the ephemeral data in pervasive computing networks and put forth a proposal for the foundation of such a model that explicitly tags information with spatial and temporal semantics.

Our data model is founded on *spatiotemporal trajectories*, which enable applications to reason about relationships between spatiotemporal phenomena and the digital data that represents those phenomena. Consider an example in which a boat collects observations of oil in coastal waters. These observations are represented as digital data carried by a device on the boat. Each of these *datums* moves in physical space as it is carried by the boat and is therefore subject to the boat's movement patterns; any one of these datums could also move by being passed from one device to another. The physical phenomenon of interest also often exhibits spatiotemporal dynamics after its observation; in this example, the observations are of a physical phenomenon subject to dynamics in both space and time as coastal currents cause the oil to move and dissipate. A datum's spatiotemporal trajectory records these dynamics of the observation and the physical phenomenon of interest, allowing applications within the pervasive computing network to create rules governing how datums capturing the phenomenon should move, change, live, and die. Spatiotemporal trajectories also provide a foundation for applications to reason about the impact of spatiotemporal dynamics on the use of a datum. In this paper, after expositing necessary components of a spatiotemporal-aware data model for pervasive computing, we further demonstrate both the need for and potential impact of a such a general-purpose expressive spatiotemporal data model through a set of use cases.

II. The Need

Much attention has been paid to capturing context, enabling resource discovery, delivering relevant information, and handling dynamics in pervasive computing environments; far less has been devoted to composing and organizing such

systems [2]. Ultimately, resources and services will need to interoperate in real-world deployments. Applications may need to seamlessly move tasks among environments [3], users may wish to aggregate information across resources [4], a service might need to collaborate with others [5], [6]. A common glue, or language, is needed to facilitate interoperability, enable composition, and maximize resource use in pervasive computing networks. This underlying uniformity will be best satisfied by embedding meta-data within the data already communicated among applications. A shared data model would greatly simplify application development and support capture of richer contexts, enabling new classes of applications.

Context-awareness is a major theme in pervasive computing. From defining it to leveraging it, context determines the *modi operandi* of most pervasive computing applications. A common approach is to explicitly define an application-specific notion of context; determining what data is "relevant" rests solely with the application. This is sufficient when data exists only instantaneously and is not shared among applications. This is not the case in many pervasive computing environments, which significantly increases the amount of data available. A far better approach is to enable the data itself to articulate the context in which it was acquired to provide clues as to contexts in which it might be relevant. A data model designed in this way would enable the data to exploit its contextual dependencies, providing a separation of concerns and greatly easing development burdens.

Traditional software models offer little support for the challenges that arise due to the inherent dynamics of pervasive computing environments [7], and developers are often forced to address these facets at the application level. A general-purpose model capable of independently exploiting contextual dependencies that could be shared across services would enormously simplify and facilitate pervasive computing application development. Rather than impose a totalitarian framework or middleware, we believe that these facilities are best addressed by and within the data already used by applications.

User and application needs in pervasive computing are driven by real-world phenomena; they are inevitably a product of the environment of the users' interactions [8]. This is evident in the popularity of location-based services. A general-purpose data model should promote the separation of concerns called for in pervasive computing and emphasize attributes present in all real-world phenomena, namely, space and time.

Every action and event that captures one's existence has *both* spatial and temporal attributes, not merely one or the other [9], [10]. Users, devices, and data "move" through time. As time passes, they may move through space. Space and time may be treated independently, but are inseparable as correlated attributes of real-world phenomena. This is intuitive; the passage of time is naturally understood in terms of perceived changes to objects in space [11]. We argue that space and time are necessary (but not sufficient) to express context. Thus, a general-purpose data model for pervasive computing must account for the inherent spatiotemporal characteristics of real-world phenomena. More to the point, space and time must be first-class citizens of such a model.

III. Background

Modeling spatiotemporal data, both in theory and in practice, is not new; spatiotemporal models have received prolific attention in the database and Geographic Information System (GIS) communities. Within pervasive computing, specifications pertaining to the creation, storage, and death of data are either embedded in the application or neglected altogether; an information-rich environment is typically presupposed.

The advent of Database Management Systems (DBMS) and positioning and tracking technology (e.g., GPS) led to a boon of conceptual and practical spatiotemporal modeling and a plethora of spatiotemporal data models, databases, query languages, and ranking and indexing techniques. A detailed survey of this work is in [11]. Recent work has focused on continued development of spatiotemporal query languages [12], on-line analytical processing [13], and extensions for mobile devices [14]. While these approaches have produced new standards in GIS and DBMS [15], they rely on centralized resources to catalog and intelligently index information. Valuable lessons can be learned from these mechanisms, but unprecedented challenges arise when an equivalent degree of spatiotemporal analysis must be performed in a distributed fashion on highly dynamic data with limited resources.

Pervasive computing is characterized by a desire to explicitly associate physical space with virtually accessible data and resources. DataSpace [16] envisions a spatially-addressed network in which physical space is modeled as a collection of *datacubes*. Querying and monitoring objects is spatially driven and constrained. In EnviroTrack [17], events in the environment are addressable entities. Context-specific computation and actuation (e.g., recording, signaling other devices, etc.) can be "attached" to these event signatures regardless of where they are in physical space. Similarly, CASAMAS [18] makes cohesive cooperative groups called *communities* first-class entities. Software agents within a community may remotely interact when they are "sensitive" to the *fields* emitted by other agents. Field intensity is modulated by space according to a *diffusion function*, and each agent type is characterized by a *sensitivity function*. In addition to spatial locality, other approaches leverage logical [5], [19], temporal [20], [21], and even social [22] locality to facilitate resource access. These approaches are indeed driven by a common need for resource access parameterized by a notion of locality, be it space, time, or some combination. However, they each address that need by embedding a model into the application itself. This is largely the case in existing pervasive computing applications.

Discovery of embedded resources is paramount in pervasive computing, and countless mechanisms facilitate efficient resource discovery in dynamic networks requiring decentralized control (e.g., peer-to-peer (P2P) networks, mobile ad hoc networks (MANETs), etc.). Distributed hash tables [23] provide distributed storage for structured spatiotemporal data in P2P networks. Similarly, query-access mechanisms for spatiotemporal data in dynamic networks [4], [24], [25] maintain virtual

overlays to handle dynamics. Rule-based approaches [26]–[28] enable autonomous opportunistic data propagation following a provided set of application-specific policies. The publish/subscribe paradigm has enjoyed copious attention in pervasive computing (e.g., [7], [29]). Publish/subscribe provides a high degree of decoupling, flexibility, and scalability, while enabling efficient event distribution. This wide variety of techniques heavily reflects a fundamental need to discover and share pertinent information in dynamic networks. These and many similar approaches focus almost exclusively on the retrieval, or querying, of information, presupposing a data-rich environment and neglecting (or simply ignoring) questions related to how data is created, replicated, stored, and destroyed. These concerns necessitate a flexible general-purpose data model that can be shared across applications and at the same time tailored to meet individual application requirements.

IV. A Trajectory-Driven Data Model

Existing work in pervasive computing focuses almost exclusively on how to *query* data, presupposing an existing data-rich environment and neglecting questions related to how data is created, replicated, moved around, stored, and destroyed.

Space and time must be first class citizens of our data model since pervasive computing intrinsically entails interaction with the physical world, and phenomena in this real world are inherently spatiotemporal. To bridge from the physical world into the digital one, we distinguish *reality* from our *perception* of it. Reality is defined by physical *phenomena* that can be sensed by the digital world (e.g., environmental conditions, availability of resources, presence of humans). An *observation* is made when a phenomenon is sensed. The observation is a digital representation of the phenomenon as it is instantaneously captured through the sensing process and is associated with an immutable spatiotemporal stamp. A *datum* is a digital representation of some *quantum* of knowledge about an observation. We want to expose the relation between a datum and an associated phenomenon (especially as data moves and time passes, i.e., as the datum experiences *spatiotemporal dynamics*). To facilitate this, we augment each datum with spatiotemporal meta-data that represents the dynamics of the datum and the associated phenomenon in space and time. This *spatiotemporal trajectory* can capture both the (expected) dynamics of the phenomenon and the (actual) dynamics of the datum. Applications that create and use the data can define *rules* that use the trajectories to determine how data moves, changes, lives, and dies. These rules are not part of our data model, but the next section gives examples of rules that are obviously useful for spatiotemporal data in pervasive computing. In a simple sense, one can relate a trajectory to *spatiotemporal decay*, or the notion that the further a datum gets in space and time from its "genesis" the "weaker" it is.

A. From Phenomenon to Datum

The first challenge is the jump from the physical to the digital. A datum represents an observation of a phenomenon digitally. We store datums as semi-structured data [30], [31] that is self-descriptive in the sense that it consists of a set of name-value pairs. We assume an underlying vocabulary for data that is shared among all participants in the digital world.

Every phenomenon is associated with a "place" and "time." The place is a description of the spatial *influence* of the phenomenon; depending on a phenomenon's type, the shape and scope of its place may differ dramatically. Space can be physical or logical; for simplicity in this paper, we assume traditional two-dimensional physical spaces. A phenomenon's time is a (potentially open) range with a discrete beginning.

An observation logs a phenomenon and generates a datum. In the simplest sense, a datum exists at exactly the phenomenon's place for exactly the phenomenon's lifetime. This may not be possible or desirable for several reasons: (*i*) the observation may not happen at the exact location of the phenomenon; (*ii*) there may not be a digital device at the phenomenon's place; (*iii*) the phenomenon's place may be larger than a single device; (*iv*) devices that store data may be dynamic or unreliable; or (*v*) we may want to disseminate the datum more widely than the phenomenon's associated place.

In our model, each datum has a spatiotemporal trajectory that captures the initial relationship between the datum and the phenomenon. Over time, the trajectory should also capture the evolution of this relationship by capturing the evolution of the datum and the phenomenon in both space and time.

B. Spatiotemporal Trajectories

A datum's spatiotemporal trajectory may evolve for many reasons: the phenomenon may be expected to be dynamic, the device carrying the datum may move, or the datum may be passed through the network. A datum's spatiotemporal trajectory should react to these dynamics and update itself. We separate *how* data items are communicated, stored, or moved from the fact that, by existing and moving in pervasive computing networks, they generate "fields of influence" given by their spatiotemporal availability. We begin with a simple yet expressive model of spatiotemporal trajectories and give insights into how these trajectories can be used to support a variety of expressive pervasive computing applications.

A datum is associated with a space-time stamp that marks where and when the observation was made. Each datum's *trajectory* is a sequence of vectors indicating the movement of the datum in space and time relative to its phenomenon. If the datum is carried by a mobile device, the vectors are defined by the path the device takes. If the datum is communicated from one device to another, then the trajectory contains a vector that traverses the distance between the devices at a velocity defined by the time the exchange requires.

Consider a data item about an exhibit in a museum collected and carried by the mobile device of a visitor:

Phenomenon. the Mona Lisa is in room 6
Observation. (at location $[x, y]$ and time t) the Mona Lisa is observed
Datum. $\langle painting = ML, (loc = [x, y], time = t)\rangle$
Trajectory. vectors of the visitor's path in the museum

An application relying on this information to share information with visitors about objects nearby them in the museum may use this trajectory to implement a form of data decay in space. This datum may *decay* with space as the distance from the painting grows. This notion of decay is similar to that captured by some existing middleware for pervasive computing [26]. The information may not decay in time since the painting is not expected to move or change. One could do something similar with a datum that decays only in time but not in space (i.e., anything that is true in all places but not at all times).

As another example, consider observations of the air quality (AQI) at a particular place and time collected by users' smartphones and shared via peer-to-peer interactions:

Phenomenon. the particulate concentration is $40.5\mu g/m^3$
Observation. (at location $[x, y]$ and time t) the AQI is 101
Datum. $\langle AQI = 101, (loc = [x, y], time = t)\rangle$
Trajectory. vectors of the movement of device(s) carrying the datum and passing of the datum among devices

An application that uses these observations to provide the user a dynamic and localized view of the air quality may degrade datums that are further in both space and time from their observations. Note that these first two simple examples only employ the the observation's space-time stamp and a representation of the "here and now." Our model is not limited to these situations, and our later examples will show how the trajectory itself may be essential to the application.

A phenomenon itself may have some space-time behavior that must also be associated with the datum. Consider the following, in which the presence of oil in water is detected by a mobile collection point:

Phenomenon. oil in water moving at $32cm/s$ to the northwest
Observation. (at location $[x, y]$ and time t) there is oil in water whose current is $32cm/s$ to the northwest
Datum. $\langle oil = \text{true}, speed = .32m/s, direction = 135°, (loc = [x, y], time = t)\rangle$
Trajectory. vectors of the movement of the mobile collection device and the (expected) movement of the phenomenon

An application can use this information to, for example, make predictions about the current location of the oil. The complete trajectory can also give information about where the oil has been, enabling direction of cleanup efforts.

In this final example, the spatiotemporal dynamics of the phenomenon (the oil in the water) are captured (as the current's speed and direction) in a very application-specific way. We have limited ourselves for now to a simple model of two dimensional space; we also simplify our representation of a phenomenon's spatiotemporal dynamics as a single vector, whose starting point is given by loc and $time$, and whose (expected) speed and direction are represented as part of the datum. In general, we can associate each phenomenon with its own trajectory that starts from the observation (as measured by loc and $time$). This trajectory could be simply a single vector as in the example, a series of vectors, or even some function of time and context whose value is a series of vectors.

C. Computing with Trajectories

Associating a spatiotemporal trajectory with an inherently spatiotemporal object is intuitive, but it is also extremely powerful. In the next section, we describe ways that this information can be used to enable applications to reason about and interact with the ephemeral spatiotemporal data that characterizes their environments. First, we give a taste of the computations that can be done on our spatiotemporal trajectories to enable additional application semantics.

Smoothing Trajectories. Spatiotemporal trajectories associated with datums that live and move for long periods of time or with high degrees of dynamics may grow very long. Using vector addition, we can "smooth" trajectories, reducing the resolution of the information about the datum's spatiotemporal dynamics, at the benefit of decreased size of the datum.

Computations on Trajectories. Applications can also do relatively simple calculations using a datum's spatiotemporal trajectory. For example, an application could compute a *decay* value that numerically represents how "far" a datum is in space and time from its phenomenon. Applications can also perform computations over multiple trajectories. For example, if a datum encounters another datum that represents either the same or a different phenomenon, their trajectories can both positively and negatively reinforce each other.

Consider our example of oil movement and the trajectory that captures the spatiotemporal dynamics of the phenomenon (i.e., the speed and direction of the current). If a datum d_1 capturing an observation at location $[x_1, y_1]$ and time t_1 encounters a second datum d_2 of the same type (i.e., oil in the water) whose location $[x_2, y_2]$ and time t_2 lie on the trajectory defined by d_1's phenomenon's trajectory, then the trajectory in d_1 can be updated to reflect new observations of the current. Such a situation is shown in Fig. 1, in which the trajectories (arrows) have been simplified to depict only the movement of the phenomenon (omitting the potential movement of the datums).

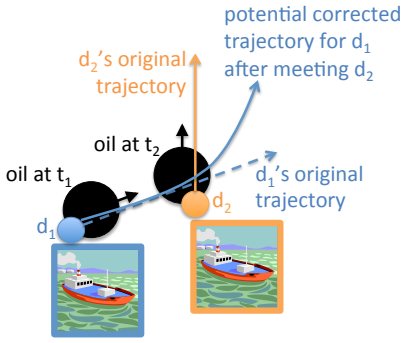

Fig. 1. Sample trajectory computation

Aggregations of Observations. We can also extend our base model to allow a datum to be an aggregation of one or more observations. Consider the following example, where observations are made by vehicles moving in an urban area:

Phenomenon. there is an available parking space on the south side of 3^{rd} Street between Pine Street and Oak Street
Observation. (at location $[x, y]$ and time t) there is an available parking space
Datum. $\langle parking = 1, (loc = [x, y], time = t)\rangle$
Trajectory. vectors of the vehicle's movement

As the vehicle carrying the datum moves, it may observe additional available parking. It may aggregate these observations

into the original datum, incrementing the counter *parking* and expanding the location and time. An application can use the associated trajectory to discover a path along which available parking spaces lie. Alternatively, the datum carried by one vehicle may encounter a datum from a different vehicle measuring parking availability along a different trajectory. These datums can be aggregated together in a more complex way to give a sense of overall parking availability in a general area (e.g., by computing the bounding box of the combination of the trajectories) or by representing the aggregate trajectory as a set of trajectories, giving a web of spatiotemporal information.

V. Use Cases

Our spatiotemporal model for ephemeral data enables pervasive computing applications to reason about the data they rely on, providing various ways to judge the (spatiotemporal) *quality* of data. In this section, we give a handful of examples of uses of this spatiotemporal model. Our model is not limited to these few uses; instead we intend to give a flavor of the variety of possibilities that exist. Effectively, each example is a way in which a pervasive computing application deployment provides a set of *rules* that dictate how the spatiotemporal trajectories associated with datums can be used to determine how data in is used, moved, stored, changed, and destroyed.

A. Data Death

The most obvious (and likely common) thing to do is to define a very thin layer on the data model to control when datums are deleted. Such a layer should be parameterized by both space and time; i.e., when a datum gets a certain distance in space and time from its observation, it should be deleted. Consider the following simple rule that a data death layer could use to periodically delete any data item d that originated more than $threshold$ units of distance away:

$$delete(d) \text{ if } dist(myLoc, d.loc) > threshold$$

Similar rules could account for time or for location and time jointly. Rules can also be defined over the entire trajectory; for example, if the sum of the trajectory's vectors indicates that the datum has *traveled* a certain distance, it could be deleted:

$$delete(d) \text{ if } \Big(\sum_{v \in d.traj} v.length \Big) > threshold$$

These definitions are not themselves part of the data model but instead are enabled by the availability of the spatiotemporal information the data model provides. In fact, the definitions are highly application-dependent. Both thresholds on data death and the definitions' use of locations and trajectories will depend highly on the application and its operating conditions.

B. Data Persistence

Many applications generate data that is relevant in a particular physical space with the desire that the data stays in that space, even if the digital devices that inhabit that space move. A carrier of a datum may desire to *transfer custody* of the datum to another device if that device is closer to a target location or has a higher degree of location stability (e.g., as computed based on local context). For example, the rule:

$$transfer(d, h) \text{ if } dist(h.loc, d.loc) < dist(myLoc, d.loc)$$

would transfer the custody of datum d to the device h if h is closer to d's initial location than the current custodian. More complicated rules could also use the trajectory to attempt to make a datum's trajectory approximate the expected trajectory of a phenomenon (e.g., in the oil example described above).

C. Supporting Fidelity Estimation

In addition to using the spatiotemporal trajectories to move, replicate, and delete data, we can also use them to post-process data and potentially reason about the quality of queries or applications it supports. We can define *quality metrics* that associate values with a datum, where the quality is determined based on space and time. For example, data that moves quickly may be determined to have a high fidelity (and a high positive potential impact on query resolution). The following rule uses the trajectory to compute an average velocity of a datum based on the velocities of the component vectors:

$$velocity(d) = \frac{\sum_{i=1}^{|d.traj|} \frac{d.traj[i].loc - d.traj[i-1].loc}{d.traj[i].time - d.traj[i-1].time}}{|d.traj| - 1}$$

Measures of fidelity could be based on how fast (or slow) data moves, how much it moves, or even how widely it moves; the appropriate fidelity metric is clearly application-dependent.

D. Directing Data and Queries

Using datums that have associated notions of spatiotemporal decay, we can think of the digital world as having *gradients* defined by the movement of data through space and time. We can use these gradients to direct data, queries searching for relevant data, and physical entities that traverse the space. For example, given the (aggregate) datum that represents parking availability in an urban area, an application on an automobile could send a reservation for the parking space in the reverse direction of the datum's trajectory.

Considering a datum that indicates the potential propagation of a plume of oil in a large body of water, subsequent queries about water temperature or the concentration of important ocean flora can follow the gradients to the potentially highly impacted areas. Fig. 2 shows a simple such situation in which a single datum generates a reverse query path; clearly multiple datums representing observations of the same or similar phenomena may generate more complex resulting query paths.

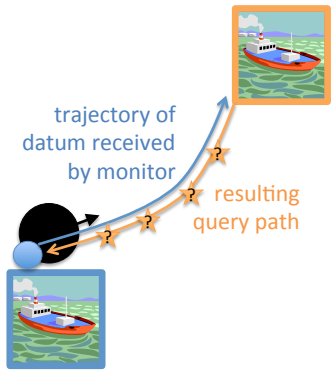

Fig. 2. Query path defined by gradient

E. Trust and Security

Space and time also have significant potential impacts on trust, privacy, and security. For example, given that we can often control access to physical spaces, even if only for brief periods of time, we can compute a spatiotemporal bounding box to determine whether a particular datum has been *compromised* by exiting some controlled space and time. The following rule defines a data time d as "*safe*" if its entire trajectory is contained within *locThresh* distance of *targetL* and occurred within *timeThresh* time of *targetT*:

$$safe(d) \text{ if } \forall \tau \in d.traj : dist(\tau.loc, targetL) < locThresh \\ \wedge |\tau.time - targetT| < timeThresh$$

VI. Looking Forward

Space and time must be first-class entities in a data model for pervasive computing as any formulation of context must be expressed in terms of one, the other, or a product of the two. The crux of our model is the *spatiotemporal trajectory*, which captures the initial relationship of a datum, in both space and time, to an underlying real-world phenomenon and the evolution of that relationship over space and time. Trajectories enable data to articulate information about the context in which it was observed and its spatial and temporal history. We envision the expressive notion of trajectory to be defined by individual applications to meet particular specifications.

Our strawman model is by no means perfect; many interesting conceptual and practical questions remain. For example, an entirely new model could be envisioned if we assume that observing phenomena is a continuous process instead of a discrete one. Other entirely new views open up if one considers a richer notion of space than just physical space. For example, consider the application of the last use case to logical spaces that represent administrative domains. Further, our strawman spatiotemporal data model and sample uses cases motivate the desire for an easy to use set of operators that allow computations over trajectories and datums. In addition to allowing applications direct access to the datums and their trajectories, these operators could enable common operations such as finding the shortest trajectory among a specified set, choosing the most recent observation from a set of datums and trajectories, or merging trajectories in ways sensitive to the semantics of the underlying observations.

As digital devices and embedded virtual resources continue to pervade our surroundings, there is an ever-growing need to effectively and efficiently capture, store, discover, and retrieve highly volatile information in environments subject to real-world dynamics. We argue that the data generated and disseminated in pervasive computing networks must be able to speak to its spatial and temporal influences to most aptly meet these needs on a large scale. To that end, this paper proposes the creation of a general-purpose spatiotemporal data model for pervasive computing environments. A foundational data model will facilitate interoperability, composition, and ultimately new classes of systems and applications.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Net.*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] W. Griswold, R. Boyer, S. Brown, and T. Truong, "A component architecture for an extensible, highly integrated context-aware computing infrastructure," in *ICSE*, 2003, pp. 363–372.

[3] J. Sousa and D. Garlan, "Aura: an architectural framework for user mobility in ubiquitous computing environments," in *WICSA*, 2002.

[4] V. Rajamani, S. Kabadayi, and C. Julien, "An interrelational grouping abstraction for heterogeneous sensors," *ACM Trans. Sen. Netw.*, vol. 5, pp. 27:1–27:31, 2009.

[5] L. Mottola and G. Picco, "Logical neighborhoods: A programming abstraction for wireless sensor networks," in *Distributed Computing in Sensor Systems*, 2006, pp. 150–168.

[6] Q. Cao and T. Abdelzaher, "Scalable logical coordinates framework for routing in wireless sensor networks," *ACM Trans. Sen. Netw.*, vol. 2, pp. 557–593, 2006.

[7] D. Frey and G. Roman, "Context-aware publish subscribe in mobile ad hoc networks," in *Coordination*, 2007, pp. 37–55.

[8] D. Mountain and A. MacFarlane, "Geographic information retrieval in a mobile environment: evaluating the needs of mobile individuals," *Journal of Info. Science*, vol. 33, pp. 515–530, 2007.

[9] A. Pred, "The choreography of existence: comments on hagerstrand's time-geography and its usefulness," *Planning-Related Swedish Geographic Research*, vol. 53, no. 2, pp. 207–221, 1977.

[10] T. Hägerstrand, "Innovation diffusion as a spatial process," 1967.

[11] D. Peuquet, "Making space for time: Issues in space-time data representation," *GeoInform.*, vol. 5, pp. 11–32, 2001.

[12] M. Lyell, D. Voyadgis, M. Song, P. Ketha, and P. Dibner, "An ontology-based spatio-temporal data model and query language for use in gis-type applications," in *GEO*. ACM, 2011, pp. 15:1–15:9.

[13] L. Gomez, B. Kuijpers, and A. Vaisman, "A data model and query language for spatio-temporal decision support," *GeoInform.*, vol. 15, pp. 455–496, 2011.

[14] T. Reichenbacher, "Geographic relevance in mobile services," in *LocWeb*, 2009.

[15] Open Geospatial Consortium, "OGC," http://www.opengeospatial.org/.

[16] T. Imieliński and S. Goel, "Dataspace–querying and monitoring deeply networked collections in physical space," in *MobiDE*, 1999, pp. 44–51.

[17] T. Abdelzaher, B. Blum, Q. Cao, Y. Chen, D. Evans, J. George, S. George, L. Gu, T. He, S. Krishnamurthy, L. L. Luo, S. Son, J. Stankovic, R. Stoleru, and A. Wood, "Envirotrack: towards an environmental computing paradigm for distributed sensor networks," in *ICDCS*, 2004, pp. 582–589.

[18] F. Cabitza, M. Locatelli, and C. Simone, "Cooperation and ubiquitous computing: an architecture towards their integration," in *COOP*, 2006.

[19] R. Newton, G. Morrisett, and M. Welsh, "The regiment macroprogramming system," in *IPSN*, 2007, pp. 489–498.

[20] R. Menezes and A. Wood, "The fading concept in tuple-space systems," in *SAC*, 2006, pp. 440–444.

[21] K. Schelfthout and T. Holvoet, "A pheromone-based coordination mechanism applied in peer-to-peer," in *Agents and Peer-to-Peer Computing*, 2005, pp. 109–132.

[22] P. Costa, C. Mascolo, M. Musolesi, and G. Picco, "Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks," *IEEE J. on Selected Areas in Comm.*, vol. 26, no. 5, 2008.

[23] E. Kuhn, R. Mordinyi, H. Goiss, T. Moser, S. Bessler, and S. Tomic, "Integration of shareable containers with distributed hash tables for storage of structured and dynamic data," in *CISIS*, 2009, pp. 866–871.

[24] A. Ziotopoulos and G. de Veciana, "P2P network for storage and query of a spatio-temporal flow of events," in *Percom Workshops*, 2011.

[25] M. Motani, V. Srinivasan, and P. S. Nuggehalli, "PeopleNet: engineering a wireless virtual social network," in *MobiCom*, 2005, pp. 243–257.

[26] M. Mamei, F. Zambonelli, and L. Leonardi, "Tuples on the air: a middleware for context-aware computing in dynamic networks," in *ICDCS Workshops*, 2003, pp. 342–347.

[27] G. Russello, E. Scalavino, N. Dulay, and E. Lupu, "Coordinating data usage control in loosely-connected networks," in *POLICY*, 2010.

[28] C. Scholliers, E. Boix, and W. D. Meuter, "Totam: Scoped tuples for the ambient," *Electronic Communications of the EASST*, vol. 19, 2009.

[29] G. Cugola, A. A. Margara, and M. Migliavacca, "Context-aware publish-subscribe: Model, implementation, and evaluation," in *ISCC*, 2009.

[30] S. Abiteboul, "Querying semi-structured data," in *ICDT*, 1997.

[31] D. Gelernter, "Generative communication in Linda," *ACM Trans. Program. Lang. Syst.*, vol. 7, no. 1, 1985.