

Ubicomplexity: Applying Useful Concepts from Complex Adaptive Systems to Ubiquitous Computing

Jason Alderman and Matt McKeon
Georgia Institute of Technology

1 Introduction

The realization of Mark Weiser's vision of ubiquitous computing, of "machines that fit the human environment" (1991), is predicated on large numbers of massively interconnected devices that can move fluidly between the edges and the focus of our perception. The number of potential interactions between these devices, their environment, and the humans within it makes it difficult to design both applications and systems-level frameworks for their development.

These sorts of multi-agent, massively interconnected systems are the subject of an evolving area of interdisciplinary study, known as *complex adaptive systems theory*. Drawing researchers from mathematics, biology, physics, sociology, and computer science, CAS represents a sprawling and disjointed body of knowledge, ranging from rigorous thermodynamic models to management textbooks on organizational theory. While the stated aims of CAS and ubicomp appear to be congruent, it is difficult at first glance to identify elements of CAS research that might be applicable to the latter.

In this paper, we outline our findings from a special topics course in ubiquitous computing that was conducted in the spring of 2005. Our goals for the course were threefold. First, we set out to survey a broad spectrum of literature in ubicomp, discuss its theory and application, and develop a competency in current areas of ubicomp research. Second, we sought to develop a bibliography of study in complex adaptive systems, and relate concepts from our readings to ubiquitous computing. Third, we wished to create a model for ubicomp applications informed by the paradigms of complexity, and create a final project based on this model.

Unfortunately, our goal of a final project within a one-semester timeframe proved to be too ambitious, given our uncertain theoretical grounding. Instead, we are recording our findings as a foundation for future work. While our model for an "ubicomplex" system remains rough-shod, we believe our work shows that concepts learned from complex systems theory can suggest valuable design lessons and directions of study for ubiquitous computing researchers.

2 Complex Adaptive Systems

2.1 What is complexity?

The study of complex adaptive systems is, for lack of a better word, complicated. So many disparate fields of study have been drawn under its title that sometimes they are like forced puzzle pieces that do not truly fit.

Complexity theory rose to prominence in the scientific community in the late 1960s, when Evelyn Keller and Lee Segler first published their studies of the formation of slime molds (1969). They found that there were not, in fact, a breed of “pacemaker” cells in the slime mold—no leaders among the unicellular elements that merged together to form a larger organism. Instead, each cell had the potential to become a pacemaker, given certain initial conditions. This breakthrough led to a paradigmatic shift in science. Nowadays, scientists increasingly think of things in *decentralized* terms—that larger patterns in large systems of individual elements are actually the result not of a few hegemonic entities directing the actions of the masses, but rather behaviors that arise from the combined actions of autonomous agents, all following the same set of rules, adapting to changing environmental conditions, and self-organizing. This school of thought, of trying to deconstruct large systems to the smaller rules that determine its larger behaviors has been applied to many fields of science and social science, from biology to psychology to economics to military strategy, with varying success. Frank Miele, senior editor of Skeptic magazine, warns that these are “important concepts in mathematics and physics, but we should look skeptically at their introduction into other fields.” (2000)

Complexity theory falls under the rubric of nonlinear dynamics, since the actions of the constituent agents in a complex system do not always have direct linear effects on the behavior of the larger system on the whole. (Usually their actions are multiplied, rather than added, making the system’s behavior difficult to model by simply applying the same rates of change as the constituents to the system as a whole.) Complexity is often confused with chaos theory, another nonlinear science, which originates from the mathematics and physics communities.

Chaos theory (in an extremely general definition of the field) explains systems that have a high sensitivity to initial conditions, yet do not settle into an equilibrium state. There are two kinds of chaotic behavior: stochastic chaos, or true randomness, and deterministic chaos, which appears random to the naked eye, but has underlying patterns. Stochastic chaos cannot be produced by mathematical formulas. Conversely, deterministic chaos is generated by equations which describe a mathematical object, known as a *strange attractor*, which governs the behavior. One can see the effects of the attractor when comparing the phase plot (a graph which plots the value of the data at any time t against the value of the data at time $t+1$) of stochastic and deterministic data sets. The stochastic chaos plot will look like a field of static on a television screen; the deterministic chaos plot will look like the outline of a Salvador Dali clock—oblong curves that somewhat resemble the motion of a pendulum. This periodicity of the phase plot—looping back and forth but not reaching complete equilibrium—shows that the data set has an order to it, but it does not settle into any kind of stability.

2.2 Lam's Paradigms

Lui Lam (2000), a professor of physics at San Jose State University, takes the field of nonlinear science and tries to outline his own history of complex adaptive systems. He claims the field proposes four primary paradigms that can influence thinking in other fields (the fourth being his own research).

2.2.1 Fractal recursion

First is the visually-recursive fractal, or the “self-similar characteristic that a small part of [the complex system], when enlarged in scale, resembles the whole.” Not all complex systems involve fractals, but fractals are a common quality of complex systems.

2.2.2 Chaos

Chaos, described above, is the second paradigm listed by Lam, and the strange attractors of chaos are fractals. Chaos theory brings to the table the consideration of sensitivity to initial conditions, and while chaos describes certain complex systems, Lam re-emphasizes that all complex systems are not chaotic.

2.2.3 Self-organized Criticality

Lam's third paradigm, self-organized criticality, is the characteristic of complex systems manifested when a system gains enough elements that they self-organize into a system that behaves as a larger entity rather than as a collection of smaller constituents. This phenomenon has been the focus of the work of many scientists, each trying to describe the same effects of self-organizing complex systems, but two perspectives, in particular stand out. The work of John Holland and the writings of Francis Heylighen (of the so-called “Belgian school”) will be discussed shortly.

2.2.4 Active Walks

Lam's fourth paradigm is the concept of active walks, in which constituent elements of a complex system leave trails that change their environment (making their walks *active* rather than *passive*) and shape the interactions of other constituent elements. Lam's primary example of this phenomenon is ant pheromones: as ants wander back and forth looking for food, they leave trails of pheromones for other ants to follow. When an ant finds food, other ants that follow its trail of pheromones will leave pheromones of their own, increasing the gradient of attractiveness of that path, and increasing the number of ants that follow that path. In this way, through simple rules of behavior, the individual ants create loops of positive feedback that encourage travel down the most efficient paths to food.

2.3 Holland's Seven Basics

By far one of the most-cited researchers in the field of complexity science is John Holland, a professor of psychology and computer science at the University of Michigan (and father of “genetic algorithms,” which determine their own fitness through iterations of “breeding”). He describes seven “basics” of complex adaptive systems: four properties and three mechanisms that help their formation (1995).

2.3.1 Aggregation

The first property, *aggregation*, refers to both the way that we model complex systems, by lumping together similar elements and treating them as equivalent, and also the way that complex systems behave. Much in the manner of *self-organized criticality*, aggregation refers to the way that “complex large scale behaviors” can arise from “the aggregate interactions of less-complex agents.” Holland goes on to say that aggregates of agents can, in turn, be meta-agents themselves in larger aggregates.

2.3.2 Tagging

The first mechanism, *tagging*, refers to the action of giving the agents in a system tags that define their possible functions and interactions. Tags “facilitate selective interaction” by differentiating the components of the system from each other and providing context.

2.3.3 Nonlinearity and flows

The second property, *nonlinearity*, which has already been discussed previously, is the way that agents’ rate of change is by no means the rate of change of the aggregate system. *Flows*, the third property, focus on the actual changes occurring between the agents. Flows are a term borrowed from economics to describe the interactions between agents, generally the changes in stocks of resources or capital as they pass between nodes in a company. MIT’s Sloan School of Business likes to describe things in terms of The Beer Game, in which stocks of bottled beer are shipped from the factory to distributor warehouses to the actual stores, and the player tries to meet both supply and demand by regulating the flows—the rate at which the factory produces beer, the rate at which they ship it to the distributor, the rate at which the distributor gets the beer to the stores to meet fluctuating demand. The game teaches Holland’s two principles of flows, *the multiplier effect*, that small changes in the stocks at any one node will cause multiplied (nonlinear) effects on other nodes in the system, and *the recycling effect*, which points out the effects of cycles in a complex system by showing that recycling resources diminishes the need to acquire them and increases the output of the system (e.g., recycling bottles with a flow from the store back to the factory means less outlay of capital to a glassblowing factory to make new bottles for the beer brewed).

2.3.4 Diversity

The fourth property, *diversity*, is simple, but carries wider ramifications. Holland points out that complex systems are filled with a “panoply” of different agents, each with their own unique capabilities. “Roughly, each kind of agent in the system fills a niche

defined by the interactions centering on that agent.” When an agent is removed, other agents with similar qualities can *adapt* to fill the hole. Thus, if the interactions of diverse agents form patterns, Holland emphasizes that these patterns evolve as agents fall out, or new niche roles are created, and existing agents adapt themselves to gap-fill.

2.3.5 Internal Models and Building Blocks

Finally, Holland’s second and third mechanisms, *internal models* and *building blocks* draw upon the other properties and mechanisms to attempt to deconstruct the under-the-hood workings of complex adaptive systems. Internal models, in Holland’s definition, are essentially the rules or algorithms, in the actual construction of the agent itself, that govern how the individual agents interact with the world. There are two types of internal models, *tacit* (simple models prescribing an action with an implicit prediction of a future gain, such as following the smell of cooking food will lead you to food) and *overt* (models that serve for explicit explorations of possible actions, such as enumerating the possible ways of how exactly to sweet-talk the chef for a bowlful of that amazing-smelling stew).

Holland argues that internal models can be inferred from looking at the morphology of the agents themselves, which are generally made of common building blocks that can be endlessly reshuffled (like eyes, ears, nose, and mouth on a potato-head face), following certain rules, to make different internal configurations. These building blocks are recombined to make internal models that compete in an evolutionary sense to determine which better fit will vet out. Approaching from the opposite direction, by studying the internal models of agents, scientists can determine and identify their building blocks. Building blocks can bring with them certain functionalities, too, which through the internal model they comprise, inform the scientist about the type of environment in which the agent operates.

2.4 Heylighen and Gershenson

Francis Heylighen and Carlos Gershenson, of the Free University of Brussels, continue thinking along these lines (2004). While they do not outline a taxonomy of basics as explicitly as Holland, they focus on the self-organization qualities of complex systems. Heylighen and Gershenson describe them as dynamic systems of large interacting components, whose elements are mutually dependent and thus reinforce certain states of the system. Global order, they explain, emerges from local interactions, which work out a *best fit* for the system that propagates exponentially to other interacting components, as the system literally gains mass, until there are no unaffiliated components left to grow the system. At this point, the feedback within the system becomes negative and attempts to prevent component loss from the system. Heylighen and Gershenson describe constructing shared ontologies in the same manner as Lam describes the active walks (using the ant metaphor, yet again) by “clustering similar concepts into categories, and linking the categories that are most strongly associated” so that algorithms reinforce their own construction of best-fit ontologies. They even propose that self-organization should be introduced into the field of ubiquitous computing: “various devices such as fridges, thermostats, or phones connected to a network can learn to mutually coordinate their activities, thus minimizing the burden on the user.”

3 An Ubicomplex System

Having outlined some relevant concepts from complex adaptive systems theory, we will now attempt to illustrate how they might be used in the design of a ubiquitous computing system. We call such a system an *ubicomplex system*, in that it uses concepts from the study of complex adaptive systems and emergent behavior to guide the development of ubiquitous computing applications. In this section, we outline the properties of a hypothetical framework that aids user configuration of mobile devices by building dynamic, task-oriented patterns of control and data flow based on simple rules. We then relate characteristics of the solution to current areas of research in ubiquitous computing. Finally, we discuss the technical and social prerequisites for a deployed ubicomplex system, and their impact on the development of real-world applications.

3.1 Properties of an Ubicomplex System

3.1.1 Tagging and Folksonomies

Tagging is the assignment of freely-chosen keywords by system users to elements of an ubicomplex system. In aggregate, these tags form a *folksonomy*, an emergent classification of devices and capabilities that is used by the system to support the identification and suggestion of relationships between devices.

Folksonomies have become a prominent feature of social software systems in recent years, with the growing popularity of social bookmarking sites such as del.icio.us and media-sharing sites such as Flickr and freesound. “Folksonomy” is something of a misnomer (it is more appropriately described as “folk classification” (Merholz 2004)), yet it shares its basic premise with a well-studied field. Emile Durkheim first examined the social and cultural values embedded in the “folk taxonomies” of traditional societies (1912), while Claude Lévi-Strauss further explored them in his major structuralist works. As with folk taxonomies, it is the lack of objectivity in folksonomies that makes them valuable sources of knowledge – they provide an ubicomplex system with “visibility” into characteristics of a user’s task context that transcends devices or media and cannot be anticipated by the designer.

According to Holland (1995), the mechanism of tagging is one of the basic elements of a complex adaptive system. Tags, when applied to entities within the system, facilitate three key operations. First is *aggregation*, the ability to group and manipulate entities with similar tags. Second is *boundary formation*, the ability to delineate groups of entities from each other. Third, and most important for our purposes, is the manipulation of *symmetries*. Tags permit a system to observe and act upon properties of an entity that might otherwise be hidden amongst the overall symmetry of the system.

For example, a user may have a camera phone, a PDA and two computers. The user may frequently synchronize data between all four devices. A system that doesn’t use tags may notice that images are usually synchronized between two devices: the camera phone and one computer. From then on, the system might suggest that action whenever the devices are connected. However, if the user introduces a fifth device (e.g. an iPod Photo) and wishes to synchronize images between all three devices, the user must

configure that relationship manually. A system that uses tags is able to detect that not only are photos only synchronized between two devices, but that those devices are the only ones with the “photos” tag. Introducing another device with the “photos” tag enables the system to suggest adding that relationship to the device.

More generally, tags enable agents to identify unanticipated sub-symmetries within the complex system. Instead of “the entire subnet”, “an individual device” or “every device that can take pictures”, agents are provided with information on task context contributed by the user, in the form of emergent symmetries between tags and the tagging of new devices. It is this mechanism that enables the system to suggest relationships based on past usage.

It is not envisioned that users will be required to manually tag every device in their possession. A device may be imbued with tags from several sources with little or no intervention by the user: an automated association based on the device’s type and capabilities, tags derived from other users, or even tags that have been acquired based on the device’s relationship to other devices in the subnet.

Several researchers have investigated metadata-rich structures for service composition in ubiquitous computing. For example, the Intentional Naming System (Adjie-Winoto et al, 1999) enabled service lookup based on hierarchies of attribute-value pairs.

3.1.2 Channels and Flows

Flows describe the relationships between devices in an ubicomplex application. A *channel* is a unidirectional channel of data or control from one device to another device. Flows are logical relationships between devices that spread across one or more channels. Ubicomplex flows are derived from Holland’s description of flows in complex adaptive systems – a flow is a changing pattern of interaction that spreads across multiple nodes in a complex system.

Flows have two properties that are critical to complex adaptive systems. First, there is the *multiplier effect*, wherein data or messages are altered by the nodes they traverse. Second is the *recycling effect*, wherein cycles across one or more flows create feedback in the system. Feedback may take the form of positive feedback (in which the entire system is tipped into a new state based on a small initial stimulus) or negative feedback (which enables the system to maintain equilibrium in the face of fluctuating environmental conditions).

One example of a channel would be the connection from a wireless still camera to a base station. Whenever the camera is signaled to take a picture, the data is then sent to the station over the channel. An example of a flow incorporating this channel might be a security application. The base station signals the camera to take a picture every 5 seconds. The picture is then forwarded by the base station on to a server. The server processes the picture and looks for an exceptional condition (such as a change in over 60% of the pixels). If it finds that condition, the server MMS-messages a user with the picture and signals the base station to increase the picture-taking frequency to 1 second. If there are no more significant changes after 1 minute, the server signals the base station to decrease the frequency to 5 seconds again.

In and of themselves, flows do not distinguish an ubicomplex system; explicitly representing patterns of behavior composed of individual relationships is an obvious

design choice. The key insight provided by Holland is that the critical flows within a complex system are delineated by tags, and may oftentimes be replicated between similar elements. By defining channels and flows in terms of the tags they connect, flows decompose relationships between devices into a network-centric, rather than device-centric, view of the system. In the wireless camera example from the previous paragraph, one might apply the same logical flow element to any combination of devices with the “images” (the camera), “storage” (the server), and “messages” (the cell phone) tags.

3.1.3 Flocks and Formations

The model for an ubicomplex system is build upon a shifting pattern of task relationships between an ever-changing set of devices. This is in keeping with Kindberg & Fox’s “Volatility Principle”, which mandates that the set of participating users, hardware, and software will be highly unpredictable in a ubiquitous computing application. However, the ubicomplex concept of *flocks* takes a step back from this general principle, and claims that from moment to moment the set of devices with which a user will interact will be only *minimally* predictable. To be more specific, a user’s personal devices may be thought of as the core constituency of a “flock” of interoperating devices. From moment to moment, devices may join or leave the flock; a user may even replace his or her personal devices with others. However, at all times, the user is at the center of a conceptual flock of devices whose relationships are patterned after a particular task.

The flows that exist within a flock are referred to as the flock’s *formation*. Changing and editing flock formations is a first-class operation within the ubicomplex user interface – the user is always in the loop within a ubicomplex system, yet may take advantage of its properties to ensure that the interaction is brief and minimal. Ideally, the system is able to suggest possible formations based on available devices and past usage. The system may even be able to “learn” formations from manual configuration of devices, based on their tags and the relationships created by users (a programming by demonstration system similar to aCAPpella (Dey 2004)). Formations are shareable and transmissible – a user may keep a “library” of formations, and design or acquire new ones to apply to their flocks.

At the lowest level, a formation consists of both channels and *slots*. Slots are tagged placeholders for devices – a device may occupy any slot with matching tags. The channels that connect slots contain implicit tags for their input and output terminals, as well as simple scripts that transform and map data between slots. Slots inherit any tags from their incoming and outgoing channels, as well as whatever tags are owned by the slot.

To continue our security example, our formation for this application would have three slots: one slot named “Image Source” with the tags “images” and “periodic”, a second slot named “Image Host” with the tags “storage” and “cpu”, and a third slot named “User Notify” with the tags “messages” and “images”. The channels would include the data channels from image source to host and image host to user, as well as control channels from the host to the source and (possibly) the user to the host.

3.1.4 Aggregation, Building Blocks, and Hierarchies

Aggregation of devices and services is a common theme in ubiquitous computing. Holland claims aggregation is one of the properties of a complex system – it is simply a means by which humans conveniently organize and simplify our understanding of a complicated set of related components. However, aggregation may also refer to the emergence of complex behaviors from simple components in a system. For our purposes, aggregation in an ubicomplex system is used as a means of organizing multiple flocks into a hierarchy of functional groups. This enables us to treat an entire flock of devices as a component of another flock. The longer-term vision of ubiquitous computing, which may include dozens or even hundreds of sensors and displays in a single room, requires a conceptual model that provides this level of scalability.

The flock is not the only unit of aggregation in an ubicomplex system. Holland observed that in complex systems, subsystems may often be identified and conceptually separated into *building blocks* within the larger system. Heylighen (2003) compared these to Benard rolls – when one heats a liquid from the bottom, local interactions result in multiple cyclical flows within the liquid. In a similar manner, a single formation may be made up of a combination of flows that bear strong resemblance to flows in other formations. A formation designer may have a palette of flows upon which to draw when constructing a new formation, or may simply copy them from a formation in her library.

In the security example from the previous section, notice that we grouped the camera and its base station into a single slot. In this hypothetical system, the camera and its base station (along with other wireless sensing devices) might form a single flock, whose tags are aggregated from its devices and may be treated as a single entity. Thus, it is immaterial to the system whether the occupant of a slot in a flock is a single device or multiple devices – one may seamlessly replace a group of devices with a single device that has multiple capabilities. In this mode, one may have a “formation within a formation,” with the ability to alter or change lower-order formations without affecting higher-order formations.

3.2 Where does this system fit?

While we have touched upon some prior research in discussing the properties of an ubicomplex system, we have yet to address the question of “Why is an ubicomplex system needed at all?” We intend to define the notion of ubicomplex system design in relation to current investigations in ubiquitous computing, and make the case for further research.

3.2.1 End-user configuration

Mark Weiser (1991) outlined a vision of “hundreds of computers in every room, all capable of sensing people near them and linked by high-speed networks.” Clearly, there is the potential for users to be quickly overwhelmed by service composition frameworks that require meticulous configuration of devices.

“Calm technology” (Weiser 1997) moves easily from the periphery to the center of a user’s attention, and back again. It provides awareness in the periphery, and control at the center. Fundamentally, we *want* the user to be involved in configuration of their

devices, in order to establish trust and maximize their utility. The key is to establish appropriate and relevant channels of input from the user.

Our system shares this approach with several other efforts. The Jigsaw system (Humble et al 2003) used a puzzle-piece metaphor to enable end-users to configure devices and develop multi-device applications. The CAMP interface (Truong 2004) abandons device-orientation altogether; users assemble magnetic poetry pieces to express intent, and the system selects appropriate devices and configures them to capture data. As a last example, aCAPpella accumulates contextual data using basic multimodal sensing capabilities, requires the user to select relevant sections from that data that characterize intent, and then utilizes machine learning to identify that intent in the future based on this data.

Of these solutions, the ideal of an ubicomplex system most resembles the “programming by demonstration” model represented by aCAPpella. aCAPpella differs from other “smart” systems in that it brings the user into the loop in order to assign meaning to particular patterns of context. Similarly, in an ubicomplex system it is the user’s task to assign meaning to devices and configurations, which the system may then organize and marshal for action. Unlike aCAPpella, an ubicomplex system explicitly “avoids intelligence”, and therefore trades some degree of task-centricity for scalability and mobility between contexts. Ubicomplex systems rely on lightweight classifications of devices and interactions, derived from both local users and the global user population, to ease the burden of end-user configuration and provide non-programmers with a means of tailoring the system to their liking. An ubicomplex system is not incompatible with intelligent systems like aCAPpella, however the two solutions represent different approaches to end-user configuration.

3.2.2 Context Awareness

Context awareness is not explicitly modeled in an ubicomplex system, although it is not precluded by the approach. In their development of context-aware applications, Dey et al. (2001) separated context into identity, location, status, and time for people, places and things, and showed that additional pieces of context might be derived from these basic categories. While this may capture the lion’s share of context necessary for most ubiquitous computing applications, an ubicomplex system also attempts to infer context from a functional perspective without reference to domain. That is, context for a user’s action may be derived from the resources available (what I can use) and the tags given to these resources (how I can use it). Because tags are domain-less, the user may define the relevant contexts for action based upon whatever criteria they choose, such as “work” vs. “personal”, “photos” vs. “music”, or “blue” vs. “green”. Almost certainly, automated means of tagging formations and devices with data from Dey’s categories would be useful – however, within the limited domain of devices and relationships modeled by an ubicomplex system, the system is not limited to these categories when selecting appropriate choices for the user. In this respect, the ubicomplex system gives up a great deal of representational fidelity for flexibility and extensibility.

3.2.3 Robustness

Robustness is a principal design goal for service composition frameworks – the requirement for an application to survive the inevitable device and network failures that accompany a heterogeneous system integrated with the physical environment. Most frameworks that provide service lookup and/or composition (such as the Intentional Naming System, uPnP, Jini and Bluetooth) permit some degree of transparency for services and devices based on their properties and capabilities. An ubicomplex system configures its elements based on their tags, and thus permits hot-swapping of devices with identical tags. Of course, depending on the implementation, this may have the effect of simply pushing the problem of robustness and graceful failure down to the level of middleware. However, from the user’s perspective, this emphasis on device capabilities rather than hardware or identity permits her to take advantage of whatever robustness is offered by the fundamental layers of the system.

3.2.4 Application centricity

An ubicomplex system occupies a middle ground between application and device centricity. As was mentioned in a previous section, many systems that permit end-user configuration simply wall-off significant amounts of complexity from their users. They are dependent upon having a programmer or other expert user re-configure the system whenever new devices or contexts are introduced.

The ubicomplex system takes a somewhat more graduated approach. All users must think about devices or collections of devices, to a certain degree, when interacting with the system. They must define what is or is not part of their flock of devices, and then apply a formation to that flock. However, less-expert users may take advantage of more sophisticated users’ flock and flow designs; one can imagine a simple interface that enables users to combine flows from different formations into a formation that suits them. Rather than plugging individual devices together, the user may assemble a formation from building blocks and then apply the formation to their devices. Thus, while an ubicomplex system must take a somewhat more device-centric view of the world than is strictly desirable, complexity in the system is more effectively distributed among the designer, the expert user, and the end user and provides for a more flexible and extensible task environment.

3.3 *What are the prerequisites for such a system?*

3.3.1 Middleware, middleware, middleware

While the properties of an ubicomplex system that we have identified may place requirements on a middleware system, they do not address the general problem of service discovery, integration, and translation of data representations. Instead, ubicomplex is intended as a conceptual model for a user’s interaction with an underlying middleware system. There are significant technical challenges involved – such a middleware system must work across a wide range of devices and protocols, and have sufficient expressive power to describe their functions. Furthermore, the middleware must have an

architectural model that permits ubicomplex features (such as tagging and end-user configuration) to be available on both traditional and impoverished devices.

That said, there are several promising frameworks on which to build. For a simple and limited version of an ubicomplex system, Bluetooth and ZigBee both provide a physical transport layer, service discovery, and device profiles that may be used to translate data representations. Jin Nakazawa's uMiddle provides a more exciting prospect, as it bridges multiple protocols and has a more generalized scheme for classifying flows of data between devices.

3.3.2 Critical Mass

A weakness of all folksonomy applications is that they require a significant user base to enter and use metadata within the system. The true power of folksonomies comes from sharing and propagating them – without a significant base to share, their utility is severely curtailed. Furthermore, there must be a critical mass of devices that are supported by the ubicomplex system's middleware and that are tagged and in use by users.

There will always be pioneering power users that can provide initial momentum, however it is only when one reaches a threshold of users who are drawn to the system by its existing capabilities AND contribute to it that one has built the necessary critical mass to tap into the full potential of an ubicomplex system.

3.3.3 Ease of Use, Suitability to Task

Would people really tag their devices, and update tags when appropriate? One can build good user interfaces to assist in the tagging process, both to autogenerate and suggest tags based on context and device. However, one must have a UI that spans multiple devices with some degree of consistency. Some other potential avenues for helping users to select tags include tag rotators (that can change a device's tags based on date/time, location, or other contextual clues), multi-tagging (altering the tags of several devices at once), and smart tag suggestion (suggesting tags that the user frequently uses on other devices but that are infrequent for that type of device, or suggesting tags that other people frequently use for that device).

Furthermore, an ubicomplex system needs a good base of available formations that support activities users actually want to accomplish. System designers may create an initial batch of formations for users to tweak, however they must rely on users to design new formations in order to broaden the scope of the system. This requires that the user interface for the development of formations be easy and available to the end user.

In general, an ubicomplex approach is most suitable for applications involving larger and more diverse numbers of devices than one typically encounters in today's world. One might conceivably create a "walled-off" system such as CAMP in which system designers can anticipate and support most sensing devices and platforms on today's market, and continue to add support for new devices as they are introduced. This would be easier to use than an ubicomplex system and appropriate within its application domain. An ubicomplex approach may not become relevant until we find ourselves in a world with far more devices available in the environment and higher user expectations for their interoperation. Or, an ubicomplex system may provide an intermediate "expert

user layer” on top of which one may build a more task-focused interface that is tailored to a specific set of capabilities, devices and contexts.

4 Application Scenarios

To better illustrate how the features of an ubicomplex system might impact everyday usage in real-world contexts, we present two application scenarios that provide examples of both traditional and non-traditional ubicomp applications.

4.1 A Day at School

Andy is a junior at LaSalle High. Compared to kids from previous generations, he's saturated with technology. Devices and tools that would have been unthinkably expensive for the average student just a few years ago are now as common and necessary as chemistry books. This scenario follows Andy through a day at school.

Andy's second class of the day, after a particularly dull homeroom, is Physics Lab. Today, each group in the class is working with a voltmeter, a breadboard and some assorted electronics parts, recording various resistances and calculating the capacitance of a circuit. To demonstrate, Andy's teacher asks each student in the room to add their notebooks to the teacher's flock. The current formation of the teacher's flock links the Bluetooth voltmeter in a demonstration setup to the teacher's notebook, which then transmits the text data from the voltmeter to other devices in the flock with the tag "display." Once the demonstration is finished, Andy removes his notebook from the flock, sets up his experiment with his partner Billy and begins to work. In order to record the data to his notebook computer, Andy browses the subnet and adds the voltmeter to his flock, then chooses a formation available from the voltmeter that streams data to any device with the tag "storage." Andy's notebook automatically positions itself in the available slot and begins recording data from the voltmeter. Unfortunately, Billy forgot his notebook that day. So, he uses his iPod Wireless (which also has the tag "storage") to join Andy's flock. Billy's iPod automatically positions itself in the "storage" multi-slot, and records the data from the voltmeter for his later perusal.

The Physics Lab over, Andy and Billy go to the student lounge to eat their lunch and relax. Billy mentions that he bought the new Anteataz album, and that it is "slammin'." He wants to play a track or two, so he joins his iPod to the music player flock in the lounge. The music player flock consists of a stereo digital music player, the small information device that drives the lounge's projection display, and whatever other audio sources with the tag "music player" join the flock. The music player is visible as a small WinAmp-like interface on the lounge's projection display. Two other kids have their portable audio devices linked to the music player flock: their devices are visible in the player interface. Only one song is currently in the queue, so Billy queues up several Anteataz tracks and turns the volume up to "high" from his iPod. An angst-filled growl rumbles through the lounge, and Andy realizes that, indeed, the Anteataz are slammin'. He attempts to join his notebook to the music player flock, and frowns when he is told he cannot. Andy notices that he must give his notebook the "music player" tag in order to join the flock. He opens iTunes, checks the "share my music" box, and then tags his notebook with the "music player" tag before joining the flock. Andy may now stream

audio from his notebook to the digital stereo in the lounge. He offers to trade Billy two other albums if he can listen to the Anteataz for a few days. They agree, and trade files by dragging and dropping them on each other's devices in the player.

The bell rings, Andy tosses his lunchbag in the garbage, and walks across the building to the library. He has a meeting with his project group for English class – they have a presentation to give on American poets. Andy, Cindy, and Danny all cluster around a table in a meeting room with their notebooks and a large pile of dusty volumes. Cindy's running PowerPoint on her notebook, and she activates a formation for her personal flock that allows devices with the "editor" tag to collaboratively edit the presentation. Andy joins the flock with his notebook and PowerPoint, and can edit it just as easily as Cindy. Danny doesn't have a notebook; however he does have a palmtop he can use to edit his part of the presentation. He tags his palmtop with the "editor" tag, and specifies that that tag should be associated with the Remote UI Client on his palmtop. This client, when activated by a remote service, requests a user interface to interact with that service that is appropriate for the palmtop. When Danny joins the flock, Cindy's PowerPoint application activates the Remote UI Client over a channel and subsequently provides it with a significantly degraded interface for editing the presentation. While he cannot see all of the nice formatting and images, Danny can edit text which is subsequently formatted appropriately by the presentation template. Bored, Andy opens up the Flocks Control Panel on his notebook screen, tags each slot in the current formation with the "music player" tag, and then connects the slots with an "audio stream" channel. He queues up the Anteataz album and commits the formation changes. Cindy and Danny both raise their eyebrows when presented with a confirmation dialog for the formation change. They look at Andy, who grins and motions for them to plug in their headphones.

It's time for their presentation in English class. Cindy joins her notebook to the classroom flock, and tries to change its formation from "Lecture" to "Presentation", so that the Smartboard will become a display and not an input device. The flock is locked by the teacher, so the change fails. He apologizes, unlocks the flock, and permits Cindy to change its formation. The Smartboard flickers, and is replaced with the display from the teacher's notebook. Cindy bumps her teacher's notebook from the "closest, image source" slot by bringing her notebook closer to the Smartboard than his. Cindy hands the Bluetooth mouse occupying the "handheld, controller" slot to Andy, who will be leading the presentation. Andy is playing the part of Walt Whitman, and has constructed a remarkable beard for the occasion. He clicks through the presentation with the Bluetooth mouse and narrates the story of American Transcendentalist poets. Unfortunately, halfway through his scandalous account of Bronson Alcott's "Fruitlands" experiment, Andy notices that the Bluetooth mouse has stopped working. Still talking, Andy fishes out his mobile phone and joins the class flock. The phone prompts him to "kick out" the mouse and join the flock in the "handheld, controller" slot. The mouse's slot was selected because it was the best match for the phone's "handheld" tag; forcibly occupying the slot has the side effect of imbuing the phone with the "controller" tag. Andy selects "OK", and his phone's display now shows how the keys are mapped to the inputs of the channel ("left button", "right button", "scroll up", etc). Andy finishes the presentation with a funny joke, and only the really nerdy guy in the class laughs.

4.2 Social Gaming

Wendy and Xavier are in San Jose, California, for the latest bleeding-edge technology conference. It's been a long day of tutorials and paper presentations; their arms are full of schwag, their feet are tired, but they're meeting their friends at a hip and trendy afterparty sponsored by some megalithic software corporation. As if the demo of the new NVidia XXXtreme Quatro Diablo AR Graphics card earlier that day wasn't enough, Xavier's been literally jibbering nonstop about this afterparty—driving Wendy crazy by talking about how much fun this corporation's afterparty was at the previous year's conference. (Wendy couldn't go because she had prior obligations.) So, tired though she is, Wendy isn't about to back out of this one and let Xavier taunt her about it for yet another year.

After dumping all their gear at the local hostel where they're staying (making sure to take their electronics with them...you can never be too cautious, after all), Wendy and Xavier hike up the boulevard to the party location. Wendy's got her PDA stashed in a jacket pocket, while Xavier, the consummate geek is lugging his gamer Tablet PC in a messenger-bag, with a panoply of other peripherals and devices in his surplus cargo pants. (Wendy just rolls her eyes.)

When they get to the party, Yvette and Zach are waiting outside to meet them and try to hurry them inside. "The poker game is starting in just a few minutes!" Poker game? Wendy raises a please-don't-do-this-to-me-Xavier eyebrow, for she's never been much for cards. Zach laughs and explains: this game is a little bit different; they have to make the "cards" first.

The game works in two parts. When the first countdown starts, it'll be their job to meet as many people as possible and tag the devices of the people that they meet with descriptors that describe their professional specialty. The range of possible professions is synonym-controlled by a safe-list from the game arbiters, but especially in the world of interdisciplinary professions, there are quite a few job titles that you could use to describe the work that any of the people at the party do on a daily basis. Therefore, card suits of people and their devices are determined by the general category of profession they're tagged with, and the value (rank) of their card is determined by the relative number of similar profession tags they've received. Wendy is suddenly struck by not-so-nostalgic flashbacks to high school, trying to collect signatures to run for student body vice president (and failing miserably).

For the second phase of the game, Zach and Xavier have agreed that they'll all meet back together to form a "hand" with their devices, and test their luck against other flocked hands. At this point, the losers of these head-to-head competitions, will be "added to the pot," or reshuffled into the flock of the winner, but in a place in the formation that is drawn upon as needed by the internal model of the flock to try to form the best hand to compete with the other flocks encountered.

Wendy's still a bit confused, but the game is already afoot, and the four players part ways to mingle in the crowd. Wendy sets her PDA permissions to allow tagging by other devices, but only for the set of tags established by the game arbiters and distributed wirelessly to the party's players. She runs into Victor, an old acquaintance from undergrad, who she hasn't seen for quite some time. While they both agree that they'd

like to catch up, this isn't really the time to have a long talk over drinks at the bar. They form a temporary flock to exchange contact information. Wendy discovers that Victor is now a marketing executive, and she tags him with

marketing advertising PR flack

Surprisingly, Victor already has two other taggings for *advertising* and *flack* (since when, Wendy wonders, did flack become an acceptable tag?), and so these become his leading tags. By the rules of the game, he isn't allowed to tell Wendy what his tags are, and Wendy has disabled her ability to see his tags automatically upon connecting to Victor's smartphone.

After four more meetings in the next twelve minutes, Wendy's made some great contacts at other companies in the northern California area, and has acquired quite a few tags for *graphic designer* (a rarity at these tech conferences, so hopefully a good card for her team's hand), but the time limit is almost up. She makes her way back to the designated rejoin point to meet up with Yvette, Zach and Xavier. Seriously (she thinks to herself as she winds her way back and sips a bottle of complimentary cerveza), this game needs a longer first phase.

Luckily for them, the team has developed quite a strong first hand. Xavier's job as a production assistant has meant that his jumble of jobs (*coffeeboy?!?*) led to a rather schizophrenic tagging and his Tablet PC isn't as strong a card as the rest of the group, but it still might be enough to help along a straight, since his rank is just below that of Updike, the animation intern that he's brought into the fold to make card number five.

Unfortunately, Wendy, Xavier, Zach, Yvette, and Updike's straight hand isn't enough to beat the full house that they find themselves up against, and while they get to know the team that's bested them, their devices are rotated into the pot of the winning flock. Wendy finds that her device has been reappropriated into the team's new hand as it finds that it can make a four-of-a-kind with the new cards. Since several of the poker hands (e.g., the flushes) encourage different ranked cards from the same suit (profession) to join together, this second phase proves to be very helpful for Wendy, as she meets some of the few other graphic designers in attendance at the conference's afterparty, and they make plans to get together the following day for a business lunch.

When all is said and done, and there are only two gigantic flocks remaining, the four friends' final flock loses to the other flock. The game's arbiters, sponsored by the megalithic corporation, give out HDTVs to all of the members of the winning flock for no apparent reason. Wendy is crushed. Exhausted, Wendy and Xavier part ways with Yvette and Zach, and plod off heartbroken to their hostel. It's a good thing that they didn't win an HDTV, Xavier placates, because they wouldn't have any place to put it in their small apartment. Wendy hits him.

5 Conclusions

As complexity and nonlinear sciences have been applied to a wide variety of fields outside of their native mathematics and physics, the fit has not always been appropriate. Jeffrey Goldstein of Adelphi University, a proponent of the use of chaos theory in psychoanalysis, has been quoted as stating that chaos “provide[s] therapists with new metaphors and analogies rather than with ways to make their mental models more mathematically rigorous.” The principles of complex systems are alluring even to people who understand that mathematics behind them, and in both cases, they should not be used simply as an attractive, hip metaphor to mask a lack of understanding of the system.

This said, we believe that there *are* several direct analogues between ubiquitous computing and complex adaptive systems study, and that many of the mechanisms and properties of Holland, drawn from biology, can be translated to similar effects in pervasive computing.

However there is a long way to go before this state of technology is reached. Firstly, a standardized middleware must be established. While uMiddle appears to be a good candidate, it is still a tool of academic researchers, and not, as they say, ready for prime time. Any such middleware that allowed tagging would have to reach a critical mass of use before it would be of much use in forming complex adaptive systems of devices.

There would also likely be issues inherent to folksonomies that would need to be addressed. Tags are only as good as the meaning that they provide, and this context can be very ambiguous. First consider that words may mean different things within the same language (“blocks” could refer to block printing, building blocks, roadblocks, barriers to a certain project, martial arts defenses, parts of a pulley system, actions in a football game, or even LEGO-building activities), and this ambiguity is multiplied when you consider folksonomies on an international scale. Many languages have words that are similar to each other not just within their language but with other languages, and the meaning can be lost in this ocean of ambiguity. Conversely, many words may be tag synonyms, used to tag a device for the same reasons, but the words may be slightly different, or even misspelled—or in a different language. If tags do not properly label the possible interactions of an agent in a system, or label the interactions in a way that is dissimilar from the rest of the agents in system, it goes without saying that the system may not work as expected.

Furthermore, how does one choose the appropriate labels for a tagged item? In current folksonomies, participants often have no tagging guidance or suggestions, but simply tag as they see fit, aligning items with their own personal ontologies. In a few cases, folksonomies will suggest tags based on the tags of other users for the same type of item (like the currently suspended *nutr.itio.us* at <http://supergreg.hopto.org/nutritious/>, to suggest tags for social bookmarks site *del.icio.us*), but this tagging-by-democracy, like any group decision-making, may miss some of the possible functionality of the item.

Overall, we feel that an ubicomplex system scales much better than a traditional ubicomp system, allowing for devices to be added automatically and find their niches a larger complex adaptive system of devices. If devices were removed or suffered technical difficulties, the system as a whole would not fail, but “deteriorate gracefully.”

Unfortunately, ubicomplex systems are a possible solution for a problem that does not yet exist. When the day arrives that pervasive computing is truly as pervasive as it has ambition to be, and ubicomp devices number in the thousands and millions, perhaps ubicomplex will cease being simply a helpful metaphor for thinking of ubicomp systems and become a practical tool for aggregating them into functional systems that accomplish more than the individual elements could on their own. Regardless, we would urge ubiquitous computing researchers to look to complex adaptive systems as a source of inspiration for the design of their own frameworks and applications.

Bibliography

- Adjie-Winoto, W., Schwartz E., Balakrishnan H., and Lilley, J. (1999). The design and implementation of an intentional naming system. *Proc. of ACM SOSP*, 186-201.
- Dey, A. K., Hamid R., Beckmann C., Li I., and Hsu D. (2004). a CAPpella: Programming by Demonstration of Context-Aware Applications. *Proc. Of ACM SIGCHI*, 6(1), 33-40.
- Dey, A. K., Salber, D., and Abowd G. D. (2001). "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications." *Human-Computer Interaction Journal*, 16(2-4), 97-166.
- Durkheim, E., (1912). *Elementary Forms of Religious Life*. London: Allen and Ulwin.
- Heylighen, F. (2003). The Science of Self-Organization and Adaptivity. Retrieved Apr 25, 2005, from <http://pespmc1.vub.ac.be/Papers/EOLSS-Self-Organiz.pdf>.
- Heylighen, F., Gershenson, C. (2004). The Meaning of Self-organization in Computing. Retrieved Apr 25, 2005, from <http://pespmc1.vub.ac.be/Papers/IEEE.Self-organization.pdf>
- Holland, J. H., (1995). *Hidden Order: How Adaptation Builds Complexity*. Reading: Addison-Wesley.
- Humble, J., Crabtree, A., Hemmings, T., Akesson, K., Koleva, B., Rodden, T., and Hansson, P. (2003) "'Playing with the Bits': User-configuration of Ubiquitous Domestic Environments." In *Proceedings of the Sixth International Conference on Ubiquitous Computing* (UbiComp 2003; Seattle, WA).
- Keller, E. and Segel, L. (1969). "On the Aggregation of Acrasiales." In *Biophysical Society Abstracts of the Annual Meeting* (1969), 13:A-69.
- Lam, L. (2000). How Nature Self-Organizes: Active Walks in Complex Systems. *Skeptic*, 8(3), 71-77. Also retrieved April 25, 2005 from <http://online.itp.ucsb.edu/online/utheory03/lam/>
- Merholz, P. (2004). Ethnoclassification and vernacular vocabularies, *peterme.com*. Retrieved April 25, 2005 from <http://www.peterme.com/archives/000387.html>.
- Miele, F. (2000). Three Race Horses & Four Hobby Horses. *Skeptic*, 8(3), 54-60.
- Pigliucci, M. (2000). Chaos and Complexity: Should We Be Skeptical? *Skeptic*, 8(3), 62-70.
- Truong, K. N., Huang, E. M., and Abowd, G. D. (2004). "CAMP: A magnetic poetry interface for end-user programming of capture applications for the home." In

Proceedings of the Sixth International Conference on Ubiquitous Computing (UbiComp 2004; Nottingham, UK).

Weiser, M. (1991). The Computer for the 21st Century. *Scientific American*, 94-104.

Weiser, M., and Brown, J. S. (1997). The Coming Age of Calm Technology. In P. J. Denning, and Metcalfe, R. M. (Eds.), *Beyond calculation: the next fifty years of computing*. New York, NY: Copernicus/Springer-Verlag.