This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

The Swarm at the Edge of the Cloud

Edward A. Lee, Jan Rabaey, David Blaauw, Prabal Dutta,
Kevin Fu, Carlos Guestrin, Björn Hartmann, Roozbeh Jafari,
Doug Jones, John Kubiatowicz, Vijay Kumar, Rahul Mangharam,
Richard M. Murray, George Pappas, Kris Pister, Anthony Rowe,
Alberto Sangiovanni-Vincentelli, Sanjit A. Seshia,
Tajana Simunic Rosing, Ben Taskar, John Wawrzynek, David Wessel

March 27, 2014

Author Institutions: California Institute of Technology, Carnegie Mellon University, University of California at San Diego, University of California at Berkeley, University of Illinois at Urbana-Champaign, University of Michigan, University of Pennsylvania, University of Texas at Dallas, University of Washington.

Acknowledgements: This work was supported in part by the TerraSwarm Research Center, one of six centers administered by the STARnet phase of the Focus Center Research Program (FCRP), a Semiconductor Research Corporation program sponsored by MARCO and DARPA, and by the Berkeley Ubiquitous SwarmLab.

Dedication: This paper is dedicated to the memory of Ben Taskar.

Keywords: Internet of things, cloud computing, swarm technologies, sensor networks, distributed computing, security and privacy, big data, model-based design.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

Abstract: Today, large numbers of sensors and actuators embedded into innovative devices are being introduced into our connected world at an accelerating rate. This *sensory swarm*, or *the swarm* for short, presents an extension of the infosphere (today embodied in *the cloud*) into the physical world. The swarm gives the cloud eyes, ears, hands, and feet, enabling services that are directly embedded in the physical world rather than just in the cyber world. There is no question that the pervasive integration of smart, networked sensors and actuators into the physical world offers huge potential to address societal problems, to improve quality of life, and to smooth the boundaries between the human and the cyber worlds. But it comes with enormous challenges and risks — both technical and non-technical. To mitigate these concerns, this paper proposes the adoption of open and universal platform to enable the simple, reliable, and secure deployment and operation of a multiplicity of distributed sense and control applications (which we call *swarmlets*). Providing access control and resource guarantees is essential to quality of experience and safety. Making the platform open and universal will unleash millions of swarm device and swarmlet developers, just as smart-phone platforms opened the door to millions of app developers.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

A Introduction

Over the past two decades, there has been a growing realization that large numbers of sensors dispersed into the environment can help to solve societal-scale problems. These sensory swarms (as they were called by Jan Rabaey in a keynote talk at the Asia and South Pacific Design Automation Conference in 2008) can be wirelessly interconnected and interact with the cyber-cloud, and offer an unprecedented ability to monitor and act on a range of evolving physical quantities. Such pervasive observations and measurements enable unprecedented learning and modeling of the physical world under dynamically changing conditions.

At the core of all this are advances in design and manufacturing technologies, which have enabled a dramatic reduction in cost, size, and power consumption of a variety of sensing and actuation devices, along with the familiar improvements in computation, storage, and wireless communication. Industry observers predict that by 2020 there will be thousands of smart sensing devices per person on the planet (yielding a "tera-swarm"); if so, we will be immersed in a sea of input and output devices that are embedded in the environment around us and on or in our bodies.

The concept of wireless sensor networks is not new. Sensor-based systems have been proposed and deployed for a broad range of monitoring (and even actuation) applications. But the vast majority of those are targeting a single application or function. The potential of swarms goes far beyond what has been accomplished so far. When realized in full, these technologies will seamlessly integrate the "cyber" world (centered today in "the cloud") with our physical/biological world, effectively blurring the gap between the two. We refer to such networked sensors and actuators as the "swarm at the edge of the cloud,"[1] and the emerging global cyber-physical network as the "TerraSwarm," encompassing trillions of sensors and actuators deployed across the earth.

TerraSwarm applications, which we call "swarmlets," are characterized by their ability to *dynamically recruit* resources such as sensors, communication networks, computation, and information from the cloud; to *aggregate and use that information* to make or aid decisions; and then to *dynamically recruit* actuation resources — mediating their response by policy, security, and privacy concerns.

Achieving this vision will require a three-level model. The **cloud backbone** will offer extraordinary computing and networking capability, along with global data analytics, access, and archiving. Mobile battery-powered **personal devices** with advanced capabilities will connect opportunistically to the cloud and to nearby **swarm devices**, which will sense and actuate in the physical world.

Ubiquitous connectivity between the cloud and mobile devices such as smartphones is almost a reality today. Through common and general programming and communication interfaces (e.g., "app" programming and TCP/IP) this connectivity has turned the cloud+mobile universe into a flexible platform enabling millions of applications that we could not have imagined a few short years ago. These parts of the system will continue

[1] This phrase was coined by Jan Rabaey in a keynote talk at the VLSI Circuits Symposium in Kyoto, June 15, 2011 [8].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

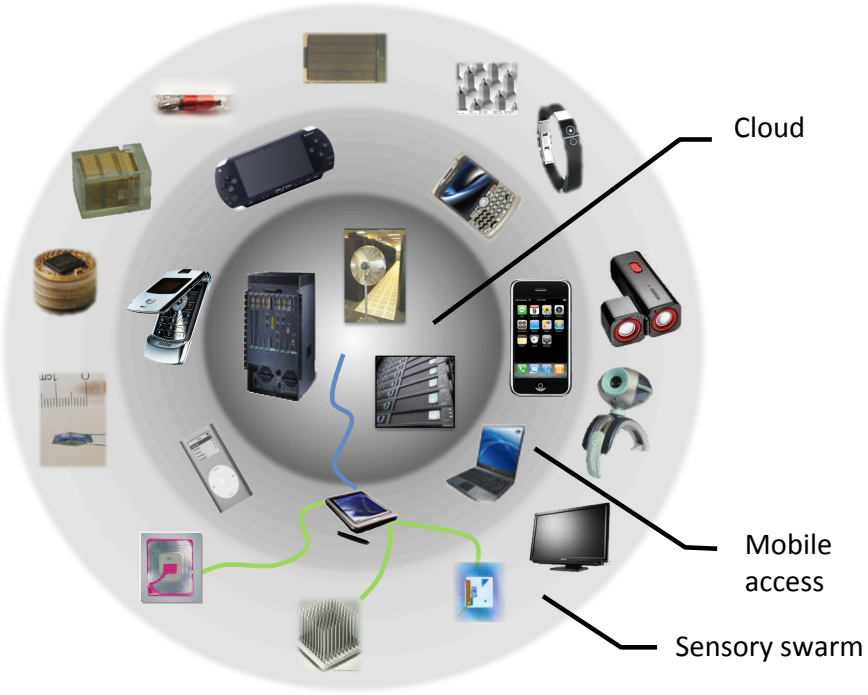

Figure 1: Three-tiered structure of the emerging information technology platform [9] .

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

to develop rapidly under large-scale commercial investment. The swarm level, however, because it directly interacts with the physical world, presents challenges that demand forward-looking research. The potential payoff of such research is a system that can fundamentally change and empower human interaction with the world.

Current "smart" applications, such as smart homes, smart grids, and battlefield management systems, typically address a single application on a dedicated set of resources. While this approach provides performance guarantees and reliability, it prevents economies of scale, and, more importantly, it prevents the explosion of possibilities that results from sharing data and devices across applications. The TerraSwarm vision cannot be achieved by a single vendor providing the components as an integrated system. What is needed instead is the swarm equivalent of the common, general, "app" framework that has recently enabled smartphones and similar devices to rapidly deploy and serve a vast range of often unanticipated applications by recruiting resources and composing services. The swarm will never achieve its potential without a "SwarmOS" on which such swarmlets can be built and composed by millions of creative inventors.

While open architectures with dynamically recruitable resources can open up significant security and privacy risks, they can also make systems more efficient (through sharing of resources), more resilient (through dynamic reconfiguration leveraging redundant resources), and more capable, enabling applications we have not yet invented or that cannot yet be realized. Adaptability, reliability, robustness, and security are essential ingredients to be considered from the start.

When the web was first launched, few people would have predicted the astounding range of applications that it would enable. It has profoundly changed the way people interact and behave, how businesses are run, and how information is exchanged. A similar revolution happened with the introduction of mobile platforms such as Android and iOS. We believe that swarm-based systems can have at least as much impact. Enabling this requires a collaborative environment in which to address the TerraSwarm's extraordinarily wide range of challenges and opportunities. By viewing key challenges through many different eyes, we expect to be able to generate a broad range innovative ideas and solutions.

B The TerraSwarm Challenge

While the TerraSwarm vision holds enormous promise, it also poses a number of daunting challenges. The technical challenges are defined by the following unique combination of characteristics of TerraSwarm systems:

- Large-scale: the swarm comprises a vast number of nodes generating corresponding "big data;"
- Distributed: components of the swarm are networked, separated physically and/or temporally;
- Cyber-physical: the swarm fuses computational processes with the physical world;
- Dynamic: the environment evolves continually;

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

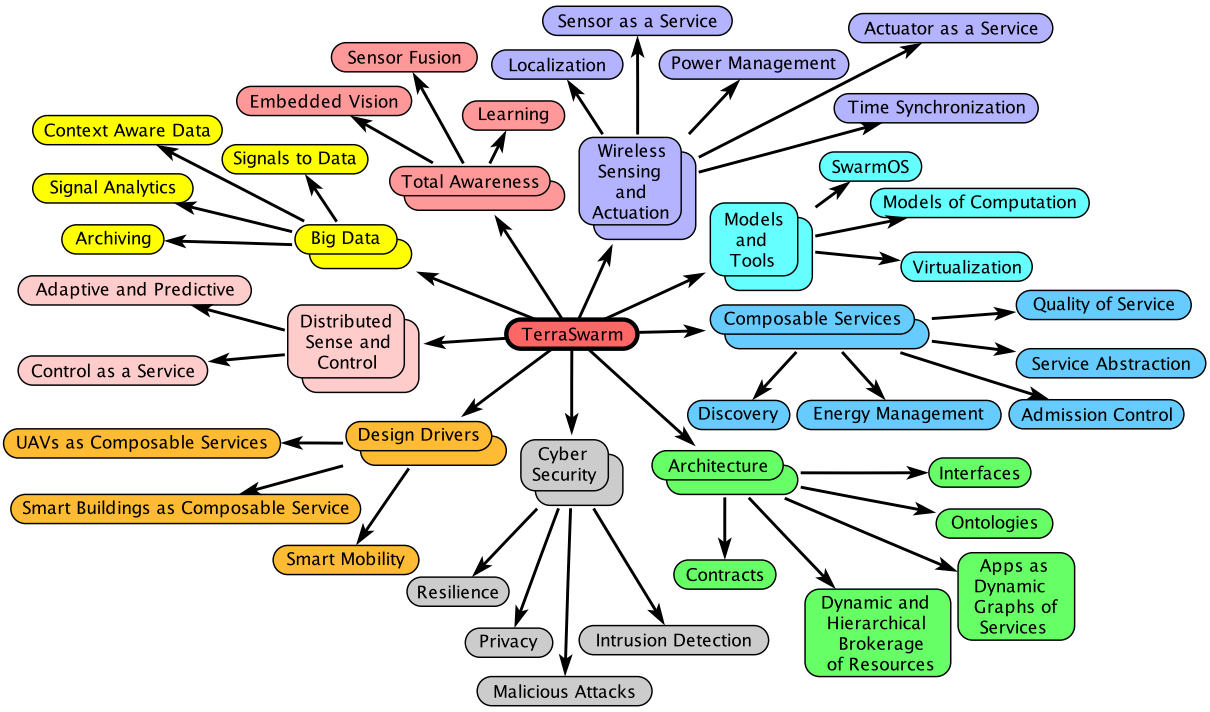

Figure 2: A map of the technical problem space of the TerraSwarm.

- Adaptive: the system must adapt to its dynamic environment, and thus the distinction between "design-time" and "run-time" is blurred; and
- Heterogeneous: swarm components are of various types, requiring interfacing and interoperability across multiple platforms and models of computation.

Given these characteristics, a sketch of the technical problem space facing a scalable and universal realization of the vision is shown in Figure 2. The challenges and opportunities include the following:

- Swarm systems rely on vast numbers of heterogeneous sensors that are generating massive amounts of data. How will these data be accessed, processed, stored, and interpreted? First, we observe that data are more valuable when aggregated than when isolated. The emergence of the social networking industry is a case in point. Second, we observe that data need not be communicated or stored if they can be predicted from models. If such models can be learned in an unsupervised way, then the TerraSwarm can be reflective, monitoring its own health, as well as the health of physical devices and humans that it interacts with.
- When data are used for security- or safety-critical systems, how can we verify that they are accurate (i.e., that the sensors are functioning properly), that they have not been compromised (i.e., secure from deliberate or inadvertent tampering), and that their source is known? We observe that today's mechanisms for identity and key management likely will not scale well to the TerraSwarm. The emergence of ubiquitous clock synchronization (with IEEE 1588 and 802.1AS for wired and wireless networks) offers

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

unique new opportunities for scalable security mechanisms, since stable local clocks provide a natural root for trust.

- TerraSwarm applications are generally cyber-physical systems that involve physical actuation and closed-loop control, and hence will have stringent testing and verification requirements. But they will also be highly dynamic, adapting their structure and recruiting resources on the fly. How can testing and verification extend to continuously evolving systems? How can we ensure that effects on the physical world are safe? We observe that on-line verification of adaptive and evolving systems will require lightweight formal methods, something that remains elusive today.
- Swarms and swarmlets that dynamically recruit resources will compete for those resources. How will costs (energy, opportunity cost, and capital investment) be managed? How can we ensure that new deployments do not disrupt established services? We observe that networking innovations such as AVB offer more control over quality of service than has been available in the past on open public networks, but how the control gets exercised in an open and competitive world remains an open question.
- How will we address data privacy and safety? We observe that, counterintuitively, privacy may be easier to preserve with more data than with less, using for example the notion of differential privacy [2].

It is worth observing that nearly every science and engineering university and a broad fraction of industry are engaged in activities that are either directly or peripherally related to swarm systems, often under the heading of the Internet of Things (IoT), the Internet of Everything, Industry 4.0, the Industrial Internet, Smarter Planet, Machine to Machine (M2M), TSensors (Trillion Sensors), or The Fog (like The Cloud, but closer to the ground). Relevant research areas include sensor technologies, actuators, semiconductors, communication systems, control systems, robotics, data analysis, data mining, modeling and simulation, operating systems, energy efficiency technologies, machine learning, data security and encoding, and cyber-physical systems, among others. Thus far, there has not been a coherent effort to bring together these disparate research efforts to serve swarm-based application development. Yet the swarm will only reach its full potential when it becomes a unified, standardized platform enabling the unencumbered development of many swarm applications.

C TerraSwarm Research

To address the challenges of the TerraSwarm, the authors launched the Berkeley Ubiquitous SwarmLab,[2] soon followed by the nine-university cooperative TerraSwarm Research Center.[3] The latter is organized along four themes. The first of these is focused on the realization of a "Smart City" scenario, developing applications that drive and test the technical developments of the other three themes. Technology development is structured around the three additional themes: Platform Architectures and Operating Systems; Ser-

[2] http://swarmlab.eecs.berkeley.edu.
[3] http://terraswarm.org.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

vices, Applications and Cloud Interaction; and Methodologies, Models, and Tools. In their totality, these four research themes cover the broad range of TerraSwarm technical problems identified in Figure 2.

C.1 Smart Cities

Cities are complex ecosystems, and their effective functioning has enormous impact on our quality of life and economic health. They stand to benefit from swarm technology, probably not through a utopian top-down authority-driven unified design,[4] but rather from the emergence of many of individual creative applications that leverage the swarm. We focus on two scenarios: a city during normal operation, and a city during natural or man-made disasters (such as accidents, infrastructure failures, earthquakes, or terrorist attacks). We call this the "tale of two cities."

In normal operation (the best of times), a swarm-enabled city not only helps run the infrastructure more effectively but empowers its occupants by providing more effective interfaces, better mobility, better information, and experiences in immersive realities in a way not possible before. For example, maintenance crews may recruit sensors from underground utilities, and combine that sensor data with data from pipe-crawling robots and from the cloud. They can use this information to guide maintenance operations using overlay displays in a manner similar to what televised sporting events use, based on contextual 3D information.

A key feature is the ability to aggregate information from multiple sources, using this information for example to reroute traffic, help citizens to find their way through the city or accomplish their chores, and identify health and safety threats (e.g. caused by air pollution). Recognizing the limitations of keyboards and screens as user interfaces, swarmlets might recruit local resources such as cell phones, nearby displays, or audio systems to interact with humans.

In the worst of times, in case of a disaster such as caused by utility failures, earthquakes, or terrorist attacks, the same systems may facilitate communication, find loved ones, mobilize response teams, and deploy robots to hazardous areas. Many swarm devices will be wireless, battery powered, and energy scavenging, offering the possibility of unprecedented robustness in the face of infrastructure collapse. The dynamic nature of the swarm, where resources come and go, implies that swarmlets must be, by design, adaptive, making them therefore, by design, more robust to failures of components.

A critical research challenge is how to recruit and compose heterogeneous resources, how to dynamically adapt applications to changing resources and contention for resources, and how to share resources without compromising safety, security, or privacy.

C.2 Platform Architectures and Operating Systems

In a TerraSwarm system, swarmlets compete for a variety of resources, including sensors, actuators, networks, computing resources, storage, energy, and wireless spectrum.

[4]For an excellent critique of such top-down utopian visions, see [4].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

To unleash the creativity of millions of swarmlet developers, we need to create a stable architecture that can dynamically balance the competing needs of distributed concurrent applications so that functionality, robustness, utility, and quality of service are guaranteed. We call the systems support for this adaptive, resource-aware architecture the "SwarmOS," a highly distributed infrastructure that touches every node in the system. Its purpose is to efficiently allocate resources based on complex optimization strategies, while maintaining appropriate security and privacy.

The SwarmOS must support continual reconfiguration of applications and of its own service definitions without ever having the luxury of a clean restart. It must also support richly heterogeneous components including sensors, actuators, networks, and computers, and it must tolerate appearance or disappearance of resources. It must be distributed and mobile, orchestrating actions across heterogeneous networks.

Swarmlets are interconnected graphs of services, likely constructed by leveraging ideas from service-oriented architecture (SOA) such as loose-coupling, service abstraction, discoverability, and composability. Service interfaces provide utility guarantees through service level agreements or contracts. So that swarmlets can be adaptive, health monitoring of the components, anomaly detection, and well-defined lifetimes for service-level agreements (SLAs) need to be provided by the SwarmOS.

A few of the services provided by a SwarmOS giving swarmlets access to resources are shown in Figure 3. One key insight that has emerged early in the project is that virtualization of computing resources is not necessarily in conflict with time-sensitive and real-time services, though more work is needed to manage quality of service in networks (leveraging protocols such as AVB and precision time protocols) and temporal isolation in processors. It has also become clear that simulation models will need to be integrated with deployed systems (a concept we call "swarm-in-the-loop simulation") in order to evaluate how services adapt in times of stress. In addition, the SwarmOS will need to embrace emerging and established mechanisms, including HTTP and REST, CoAP, XMPP, The Thing System (TTS), and many others.

The security of information, actuation, and brokerage is essential to the success of the TerraSwarm vision. For the leaf nodes of the TerraSwarm system (the sensors and actuators), it is important that security mechanisms be built-in without becoming an undue energy burden. This leads to a need to develop *energy-efficient* hardware support for encryption/decryption, authentication, and hardware-enforced key management.

C.3 Services, Applications, and Cloud Interaction

The TerraSwarm vision is one of composable services that can be dynamically recruited by applications. Formally, applications are defined as dynamic, distributed graphs of connected services. Both "dynamic" and "distributed" are important here; applications persist even as the individual components that comprise these applications change. This view elevates the concept of an *integrated modular architecture* (IMA), today's target for systems-of-systems design, from the system level to the enterprise level, and augments it with discovery, data aggregation, and run-time adaptation.

From the user perspective, the TerraSwarm provides (contextual) awareness, enabled

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

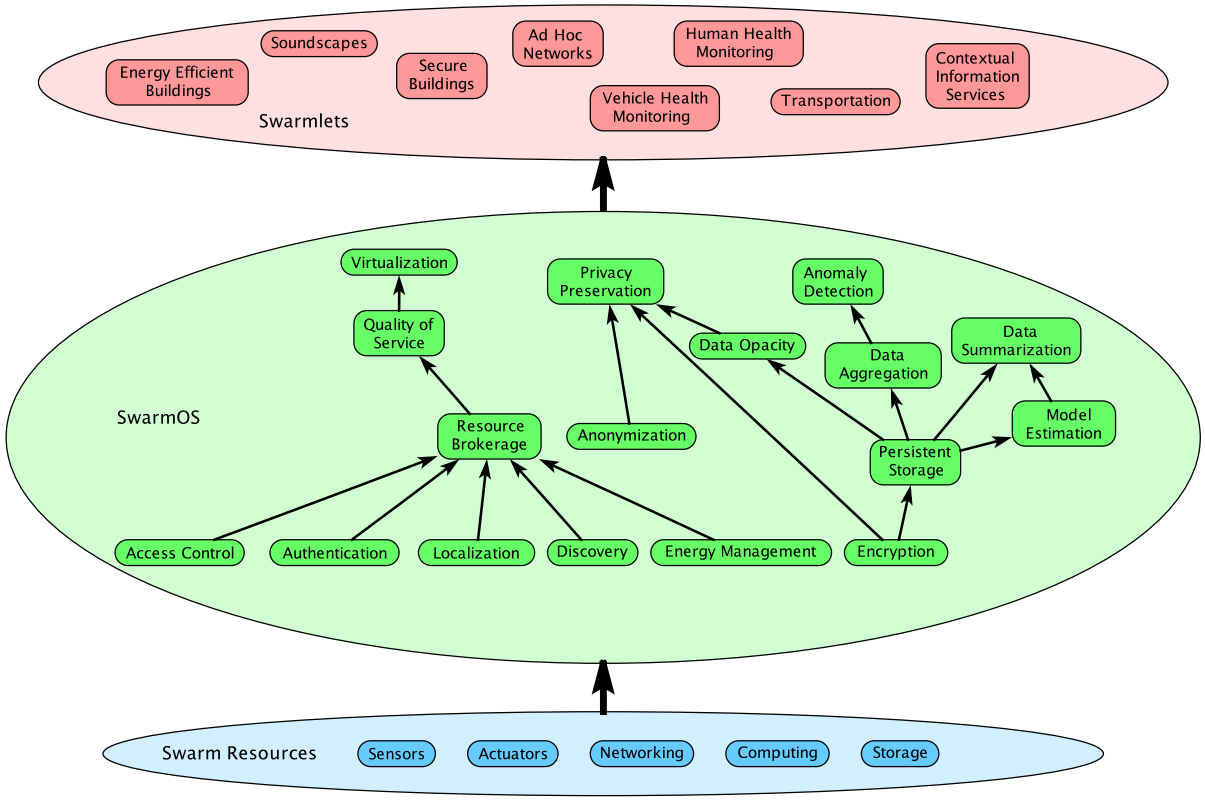

Figure 3: A few of the services provided by a SwarmOS giving Swarmlets access to resources.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

by a dynamically changing mixture of local and remote swarm sensors. Adaptive services will exploit these devices to improve accuracy and quality for the user. Actuators, such as mobile robots, can be used to place sensors and networking capability where needed. Ensuring that such adaptive services remain effective, efficient, and safe under dynamic restructuring is a challenging control problem. The TerraSwarm vision is to decentralize the design of such systems, improving their robustness and making them more adaptable and opportunistic. Control strategies will be synthesized on the fly from formal goal specifications and constraints (e.g. safety constraints), a vision we call *control as a service*. Formal methods will ensure that constraints are enforced as the application adapts.

A central challenge to be overcome is the imbalance between the massive amounts of information that could be collected and the time-sensitive interests and needs of the user(s). A naïve approach is to collect and store all data, and have cloud-based services distill the information for user consumption. But the most interesting services will need the right (contextual) data at the right time and the right place. Closed-loop cyber-physical interactions will not tolerate the latencies incurred by cloud-based archiving and indexing. Moreover, the vast data flood that will emerge from the TerraSwarm make this naïve approach far too costly, even with huge advances in storage technology. A smarter approach is use data to build and refine models of the data sources that, in effect, learn the normal behavior of the data source. Only the anomalous data, not predicted by the model, require a reaction. A useful and reasonable reaction might be, for example, data summarization, where only significant events are presented to human observers.

As with social networks and information search technologies, the cloud participates by aggregating data from a multiplicity of sources, something not possible on a single physical device, no matter how much computation and memory capability it has. The cloud is not just a computation and memory resource; it is an information aggregator and a service synthesizer. Data aggregation allows us to shift feedback control from the system level to the enterprise level. Imagine for example fleets of vehicles whose aggregated sensor data is used to improve the efficiency, reliability, and safety of each individual vehicle.

Aggregating data comes with risks to privacy. The web and social media have opened the floodgates of personal information available about us even to strangers. Even as our culture is only starting to learn to deal with the consequences of that information flood, that flood is about to be itself overwhelmed by data streams from physical sensors. The TerraSwarm vision is that security and privacy must be built into the very core of service definitions.

One approach uses a system theoretic formulation to address privacy concerns, defining filters that release useful information without compromising privacy. Such an approach can rely on the notion of differential privacy [2], which provides strong privacy guarantees against adversaries with arbitrary side information.

There is also potential data leakage introduced by composable services through side channels such as timing and power consumption. Fortunately, there are synergies. For example, temporal isolation may be introduced to guarantee resources to safety-critical services, but it can also be used to prevent side-channel attacks, where private information is deduced from temporal variations in software execution.

Security-related technologies and techniques such as static analysis, hazard analysis,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

and elliptic curve cryptography will also prove useful. So will existing research in the area of secure distributed storage [5, 7, 3, 1], which can inform the design of cloud-based swarmlets that need strong guarantees of security despite their reliance on physically insecure infrastructure.

In a connected world, the physical location of information does not matter much; the same is not true of sensors and actuators. Location is much more important at the cyber-physical boundary than in the cyber world of information technology. However, for many swarm services, location in three-dimensional space is not nearly as important as *semantic location*. A swarmlet needs to know what room a temperature sensor is in, for example, not where in 3-D space it is. In fact, small measurement errors could lead to significant mistakes, placing a temperature sensor on the other side of a wall, for example. The SwarmOS must provide infrastructure for location-based services, for semantic localization, and for learning the logical structure of semantic spaces by observing mobility.

C.4 Methodologies, Models, and Tools

A key challenge in designing TerraSwarm applications and infrastructure is that the distinction between "design time" and "run time" becomes blurred. Ensuring that different components and subsystems can be dynamically recombined yet still function properly will require new, highly advanced development methodologies, models, and tools. Functions to be realized must be separated from the components that will be used to realize them (the "separation of concerns" concept [10]). Programming models must be less centered on algorithms (step-by-step transformation of data) and more centered on dynamics (change of state over time), distribution, discovery, and adaptation. Optimizations that might be performed at design time in a conventional system-of-systems, such as mapping of functions to resources, will need to be performed at run time. Design-time testing and verification will not be adequate, because components and applications are dynamically composed and recomposed. Validation will need to be performed at a higher level, will need to cover families of possible run-time configurations rather than just one, and will need to include run-time validation strategies that are lightweight and energy efficient. Research will be needed in advanced modeling, verification, and adaptation approaches.

We have already observed the central role of models for managing the data flood. Models also play a central role in swarm system design and adaptation. Developing TerraSwarm systems will require the ability to effectively model system components and their interactions. Models must capture the evolving availability of services and resources, which can potentially be combined to provide many different types of applications. Models must also capture the rules for recruiting and combining resources and services. Current modeling approaches do not support the complex, dynamically changing characteristics of TerraSwarm systems.

We model TerraSwarm systems as a dynamic hierarchical graph of components that comprise the system. The nodes of the graph represent services. Since these graphs are hierarchical, a node may itself be a graph aggregating sub-services to define a new service. The edges in the graph represent (i) communication paths between components; (ii)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

authority relations between components; (iii) use relationships (i.e., service x uses service y); (iv) ownership relations; (v) coordination; (vi) controllability; and (vii) observability. A *configuration* of a TerraSwarm system is a particular graph structure that selects specific capabilities of the nodes in the graph (i.e., subsystems).

Verification of TerraSwarm systems' functionality will be difficult. The large number of components, their heterogeneity, and the dynamically changing structure will render exhaustive formal verification impractical. Instead, we will need compositional and incremental techniques. Compositional techniques hierarchically infer properties of compositions from properties of components. Incremental techniques infer properties of a configuration from properties of a similar configuration.

Compositional verification is enabled by assume-guarantee reasoning, which requires models of the environment. (Assume-guarantee contracts are described in [11].) In a dynamic TerraSwarm context, these models will likely be incomplete, and hence will need to be inferred or refined from observations. Such models will be imperfect, and therefore should include metrics of uncertainty that verification techniques can reason about.

Good models provide not only opportunities for formal analysis, but also opportunities for simulation. Because of the complexity of the systems of interest and the uncertainty about the environment in which they operate, simulation models will be more valuable when coupled with systematic mining of requirements. That is, although simulation models will always be valuable for human designers to develop and understanding of a system, they can be even more valuable when combined with automated exploration of the possible system behaviors.

In the dynamic network of a TerraSwarm system, non-interference properties become key. For example, when a node joins or leaves a network, it must not disrupt any service that does not depend on this node. Non-interference of temporal properties becomes particularly important for closed-loop cyber-physical systems, because if one service disrupts the timing of another, it may change the dynamics of a physical system in undesirable ways. Hence, models will need to include temporal specifications that verification techniques can reason about [6].

Not all nodes in a system will be equally trusted. TerraSwarm protocols will need to detect compromises, distinguish trusted from untrusted data and resources, and be robust to the presence of a certain number of malicious nodes. Techniques based on a combination of formal methods and algorithmic game theory (e.g., [12]) can be effective in analyzing the impact of untrusted, potentially malicious agents.

Good models will also play a central role in the adaptiveness of swarm applications. TerraSwarm applications need to deploy resources dynamically in order to achieve mission goals, and these goals may change based on circumstances encountered in the field. Typical optimization strategies for determining how best to deploy resources depend on knowing the spatial probability distribution of relevant events — but in a TerraSwarm system, this distribution will not be known in advance.

By leveraging theoretical and algorithmic tools developed for adaptive systems, we can derive new simple algorithms for complex tasks, such as coverage, source seeking, distributed partitioning, and tracking under uncertain communication constraints. These algorithms do not depend on a model of the environment, exploiting instead event ob-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

servations during deployment. Moreover, they adapt to slowly varying environmental conditions or sudden but infrequent environmental changes.

D TerraSwarm Applications

A key characteristic of the TerraSwarm approach is that infrastructure is shared among multiple swarmlets. A few carefully-chosen applications can help drive the research in the right direction. The applications used for research need not themselves be innovative; indeed, a successful infrastructure will lead to applications that none of us will have anticipated, as has happened with smart phones. Examples of applications that could be useful to drive the research are illustrated below.

- *Consumer Applications.* TerraSwarm systems enable a much richer set of consumer applications because of their interactions with the physical world. Consider, for example, a *smart jukebox*, which is a relatively simple swarmlet that incorporates several key TerraSwarm characteristics. During normal city operation, it uses information about local demographics and listening preferences to generate a customized playlist, which can then be used by restaurants (or other public meeting spaces) to adapt their soundscapes to the preferences of their customers on a dynamic basis. Leveraging the work at the Berkeley CNMAT (Center for New Music and Audio Technology),[5] it is even possible to deliver different soundscapes to different locations within a public forum (using beamforming and very large speaker arrays), and to extend to soundscape synthesis rather than just delivery. Interaction devices such as touchscreen tables could extend the smart jukebox into the social networking world, allowing for participatory soundscapes that go well beyond Karaoke.

  The smart jukebox will require semantic localization, analysis of personal information available from mobile devices and social networking databases, and dynamic resource recruiting and control. The application will be required to construct models of musical preferences, infer models from sample behaviors, find optimization criteria and algorithms, construct statistical models of user populations and system dynamics, improvise subject to constraints, analyze the system dynamics, analyze privacy and security, and optimize the delivery mechanism according to the available resources. It can leverage existing machine learning technology used in (for example) Pandora and Apple iTunes' Genius Bar, both of which aggregate information about musical preferences and make predictions about new songs that are likely to be enjoyed.

  In emergency scenarios, the smart jukebox infrastructure can be used to identify the location of people with relevant skills (e.g., doctors, electricians, off-duty police officers) and alert them via the localized sound system or text message that their skills are needed at a nearby location. By aggregating information about available

[5] http://cnmat.berkeley.edu/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2014.2314600, IEEE Design and Test

human resources and their locations, the system can more effectively direct resources to appropriate locations and optimize emergency response times.

Although its utility in normal, day-to-day operation is not critical, the smart jukebox is a technologically challenging application that is serving as a good test case for key aspects of the TerraSwarm tools and methodologies.

- *Autonomous Vehicle Response.* Advanced TerraSwarm applications can include deploying autonomous vehicles. These may include, for example, cars, aerial drones, or micro-robots, which may be required to operate alone or within coordinated groups. The range of possible uses for autonomous vehicles is huge. For example, in the best of times, they can be used for accident and crime prevention; in the worst of times they may be used for emergency response, rescue efforts, surveillance, construction of ad hoc networks, or delivery of medications. This application leverages considerable expertise in the design of vehicle trajectories, control laws, and decision-making protocols for autonomous vehicles, including micro UAVs (unmanned aerial vehicles). Tasks that must be performed by these vehicles include collecting information using mobile sensors, transporting physical objects and/or people, establishing and maintaining *impromptu* communication — all of which must coexist with other (human-operated) vehicles.

  Under emergency conditions, mobile vehicles must be capable of operating as individual units, in ad hoc groups established by local proximity, or as a city-wide resource, with intermittent communication capability. Real-time, distributed algorithms for aggregation of information, interaction with cloud services, and cooperative control and decision-making can be tested in this context and used to explore new TerraSwarm services and applications.

- *Health-Related Applications.* The TerraSwarm infrastructure (together with the cloud) will have access to a variety of health- and lifestyle-related data, including people's location, activity, and vital signs (via mobile devices and wearable sensors, as well as imagers embedded in the surrounding environment); environmental conditions (via networked sensors); and social connections (via the social networking infrastructure). Some of this information may be provided by streams of data from innovative sensors, such as energy-harvesting wearable sensors, or from wall-size imagers. To close the loop, analysis of data from such sensor streams might be used to guide people towards healthy activities or to optimize the performance of troops, police, and medical personnel.

  TerraSwarm infrastructure also provides a unique opportunity to traverse in time and analyze data and models that were collected in the past to predict or analyze the onset of a disease in future. Wearable sensors can provide details of the unique physiological observations that may not be reproducible in the future. Many medical conditions develop over time, and are not noticed until they have a significant impact on the patient's health. Once the condition has developed, data that details the progression of the condition may have been archived by a TerraSwarm infrastructure.

For example, a neurologist diagnosing a dementia patient may be interested in observing gait parameters from 5, 10, and 15 years ago. Data collected from fitness-oriented swarmlets could be used in diagnosis if stored and retrieved properly using the TerraSwarm framework.

E Towards an Interdisciplinary TerraSwarm Community

Progress towards the TerraSwarm vision requires an astounding breadth of expertise, in large-scale, adaptive, cyber-physical control systems; programming models and tools for heterogeneous, real-time, and distributed cyber-physical systems; security in systems with dynamic topologies; machine learning; privacy; networked sensor and actuator platform design; signal analytics; wireless networking and distributed systems; system architecture; human-computer interaction; energy-aware system design; and application platforms. To nurture the development of a TerraSwarm research community, the authors led the organization of the First International Workshop on the Swarm at the Edge of the Cloud, held September 29, 2013, in Montreal, Canada, in conjunction with ESWeek.[6] Only a truly multidisciplinary approach will bring the TerraSwarm vision to reality. As such, this paper serves as an open invitation to anyone interested to join this exciting endeavor.

[6] http://www.terraswarm.org/conferences/13/swarm/

References

[1] P. Druschel and A. Rowstron. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. In Proc. of ACM SOSP, 2001.

[2] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, Theory of Cryptography, volume 3876 of Lecture Notes in Computer Science, pages 265–284. Springer Berlin / Heidelberg, 2006. doi:10.1007/11681878_14.

[3] K. Fu, M. F. Kaashoek, and D. Mazières. Fast and secure distributed read-only file fystem. In Proc. of USENIX Symp. on OSDI, 2000.

[4] Adam Greenfield. Against the smart city (The city is here for you to use). Do Projects, New York City, 2013.

[5] J. Kubiatowicz et al. Oceanstore: An architecture for global-scale persistent storage. In Proc. of ASPLOS, 2000.

[6] Edward A. Lee. Computing needs time. Communications of the ACM, 52(5):70–79, 2009. doi:10.1145/1506409.1506426.

[7] D. Mazières, M. Kaminsky, F. Kaashoek, and E. Witchel. Separating key management from file system security. In Proc. of ACM SOSP, 1999.

[8] Jan M. Rabaey. The swarm at the edge of the cloud the new face of wireless (keynote presentation). In Proceedings Symposium on VLSI Circuits, pages 6–8, Kyoto, 2011.

[9] Jan M. Rabaey, Dan. Burke, Ken Lutz, and John Wawrzynek. Workloads of the future. Proceedings of the IEEE, 25(4):358–365, July-August 2008.

[10] Alberto Sangiovanni-Vincentelli. Quo vadis SLD: Reasoning about trends and challenges of system-level design. Proceedings of the IEEE, 3:467–506, March 2007.

[11] Alberto Sangiovanni-Vincentelli, Werner Damm, and Roberto Passerone. Taming Dr. Frankenstein: Contract-Based Design for Cyber-Physical Systems. European Journal of Control, 18(3):217–238, June 2012.

[12] Sanjit A. Seshia and Alexander Rakhlin. Quantitative analysis of systems using game-theoretic learning. ACM Transactions on Embedded Computing Systems (TECS), 11(S2: 55), 2012.