International Journal of Advanced Science and Technology
Vol. 15, February, 2010

# Architecture for Automatic Management of ParcTab Ubiquitous Computing

Maricel O. Balitanas and Taihoon Kim

*Hannam University, Department of Multimedia Engineering, Postfach, 306 791*
*jhe_c1756@yahoo.com, taihoonn@empal.com*

***Abstract***

*All models of ubiquitous computing share a vision of small, inexpensive, robust networked processing devices, distributed at all scales throughout everyday life and generally turned to distinctly common-place ends. This paper presents a solution for Parctab Ubiquitous Computing experiment previously been studied*

***Keywords:*** *RFID, Security, Threats, Anti-Collision Protocol*

## 1. Introduction

The PARCTAB system integrates a palm-sized mobile computer into an office network. Although computers are becoming ever more common in appliances such as VCRs, microwave ovens, and personal digital assistants, they remain largely isolated from one another and from more powerful desktop and laptop machines.

Ubiquitous Computing philosophy which is described briefly in the next section. The system is based on palm-sized wireless PARCTAB computers (known generically as "tabs") and an infrared communication system that links them to each other and to desktop computers through a local area network (LAN).

## 2. Ubiquitous Computing

### 2.1 Ubiquitous Computing Challenges

Handheld and wearable computers, mobile phones, digital cameras, satellite navigation, and a host of similar devices join the Personal Computer as commonplace digital tools. We are increasingly becoming accustomed to using a heterogeneous collection of computing devices to support a growing range of activities. These embryonic forms of ubiquitous computing technology have already had a major impact on the ways that people work, learn, entertain themselves, and interact.

The current generation of interconnected devices represents only the start of a shift towards a world of ubiquitous computing. Such devices will continue to diversify in the ways in which they sense and impact the physical world. We will increasingly share the world we inhabit with a massive set of embedded computational elements capable of sensing our activities and responding to them in a variety of ways. This shift requires us to change our view of computing from its current device centred perspective to one where "it is invisible, everywhere computing that does not live on a personal device of any sort, but is in the woodwork everywhere."1 This has fundamental consequences for how we might reason about, construct and use computer systems. The technology has developed apace, far ahead of our ability to reason about these new systems, to develop the engineering principles underpinning their construction, and to understand how we might experience the ubiquitous environment enabled by them. Let us illustrate the scope of this challenge by considering a medical scenario. Envisage that the entire population of the UK is to be instrumented as part of the NHS for systematic monitoring of metrics such as heartbeat, skin conductivity, blood sugar etc. In fact, we are already seeing embryonic explorations of this arrangement for people at risk, allowing medical staff to take prompt remedial action. If this arrangement could be applied to all it would have significant benefits for health care, health promotion and medical research. We might envisage dynamic medical records that are both up-to-date and consistent, between patient, hospital and even emergency services. We could consider timely intervention and diagnostics. We might even envisage autonomic responses where major incidents are dealt with in a timely manner. For example, these responses may trigger defibrillators for cardiac patients undergoing unattended attack. This scenario illustrates some of the key issues that need to be resolved, as a world unfolds in which ubiquitous computing is the norm. In essence, ubiquitous computing is a challenge that affects all of computer science. It asks fundamental questions about how we might reason about computer systems and computability, how we might develop complex ubiquitous systems, and how we might understand the experience of environments that are supported by ubiquitous computing. The challenge draws together researchers from three distinct perspectives:

• The experience perspective focuses on how people might share a world with ubiquitous computing environments. What interactive principle underpins our interaction with them, and how might a ubiquitous computing society be shaped from a socio-technical perspective?

• The engineering perspective focuses on the architectural and network challenges posed by the large scale, heterogeneous and dynamic nature of ubiquitous computing. What engineering principles are needed to allow a vast array of devices to be interconnected in a system, and how might we understand and respond to the system's emergent behaviour?

• The theoretical perspective focuses on concepts and rigorous models that capture the behaviour of ubiquitous systems at varying levels of abstraction. How do we reason about such a system, in order to understand its aggregate behaviour in terms of the behaviour of its subsystems? The core of Ubiquitous Computing lies in the convergence of these different perspectives, leading to a successful blend among them. This requires fundamental research into each of the constituent areas. While each of these has its own distinct perspectives and goals, they are closely linked. They may advance with somewhat distinct time-scales, tools, principles and milestones, but their development will be coordinated by projects that contribute to each perspective. Collectively, they constitute a response to a Grand Challenge whose goals are as follows:

• To develop ubiquitous computing methods and techniques that are sensitive both to the needs of individuals and society, and the impact upon them. These will support the realisation of human experiences and will include new forms of interaction and new interaction paradigms that make ubiquitous computing usable by all.

• To define a set of system design principles that pertain to all aspects of ubiquitous computing; are agreed among both academic and professional engineers; are taught regularly in Master's Degree courses; and are instantiated in the design and rigorous documentation of several computational systems with a successful operational history.

• To develop a coherent informatics science whose concepts, calculi, models, theories and tools allow descriptive, explanatory and predictive analysis of ubiquitous computing at many levels of abstraction; to employ these analyses to derive all its systems and software, including languages; and to justify all its constructions by these analytic tools. These are ideal goals, but there is no argument that places a limit on the extent to which they can be achieved.

The Grand Challenge must be addressed by collaboration across these perspectives, developing scientific theory, engineering principles and usage guidelines together in an iterative manner. In the spirit of a Grand Challenge the second and third goals are phrased to allow assessment, after one or two decades, of success in attacking the goals. The unpredictable nature of ubiquitous computing makes it impossible, at present, to formulate criteria of success for the first goal; the desired forms of experience will only emerge incrementally. What are the qualities that characterise a Ubiquitous Computing System (UCS)? It should be embedded in and become part of the environment, allowing the user to focus on the activity at hand and not the system. Its purpose is to serve people; this certainly entails interaction with users and control by them – dealing with breakdowns, setting preferences etc. Nevertheless, much of its management (e.g. configuration, handling of faults and adaptation to context) will be done autonomously and people will not be aware of it. A UCS may involve large – even enormous – populations of entities that deploy themselves flexibly and responsibly in its work. An entity may be a hardware device, a software agent or an infrastructure server; for some purposes it may be a human; it may also be an agglomeration of smaller entities. Thus a UCS that pervades our lives, but remains controllable, will demonstrate many qualities:

- It will be fluid; its structure will vary in the short term and evolve in the long term.
- Each non-human entity will be purposive, whether its purpose is expressed vaguely or formally; this is what explains its actions.
- It will be partially autonomous; some of its actions are determined by its purpose and its interactive experience, rather than by invocation from a higher authority.
- It will be reflective; a subsystem can report its experience to a higher system (perhaps to a human), to permit intervention or guidance.
- It will be trustworthy; it will behave in a dependable manner and will not adversely affect information, other components of the system or people.
- It will be sustainable; its components – hardware and software – are designed and built for long-life, efficient and effective maintenance and eventual decomposition, while its lifetime impact on the environment (including humans and power sources) is appropriate but minimal.
- It will be efficient; any delays in its performance will be tolerable.
- It will be scalable; its subsystems will differ in size by many orders of magnitude, yet unmanageable complexity will be avoided by applying the same principles of design and methods of analysis at each level. Qualities such as these (there will be more) must permeate the whole of a UCS. In the following sections we expand on many of the concerns we have mentioned, and establish a few subgoals. In each of the following sections the reader may repeatedly detect the relevance of these qualities, even when they are not explicitly mentioned. We consider these subgoals from the experience perspective in Section 2, from the engineering perspective in Section 3 and for the theoretical perspective in Section 4. While these perspectives are presented in separate sections, we emphasise that they pertain to a single Grand Challenge and their activities must be closely related. In section 5 we propose what our next steps should be. They will include exploratory projects carried out with modest aims; crucially, they will combine different perspectives of experience, systems and theory. When several such projects are complete we can hope to define a roadmap that predicts a ten-year exercise to achieve the goals of the Challenge.

## 3. Human Interaction in Ubiquitous Computing

The issue of human interaction pervades ubiquitous computing, as part of its holistic view spanning technology, use and users. We need to determine how people will understand and interact with a ubiquitous environment and how they may use it to interact with other people. Ubiquitous computing requires us to develop new techniques to interact with digital devices and poses fundamental questions for theories of human computer interaction. Interaction with Environments As ubiquitous computing environments become increasingly part of our everyday lives, we need to understand how people will interact with and exploit them. We also need to understand how user interaction will help shape these environments. Interaction will interleave influences that are contextual, individual and social and technologies will be appropriated with interaction evolving over time. People face potential challenges when attempting to establish useful or enjoyable interaction with ubiquitous computing technology. This interaction needs to fit with their contexts, interests and aims. The capabilities of different digital technologies, settings of use and individuals' own past experiences act as both resources and constraints in shaping this interaction. Interactive elements in the environment will range from small scale embedded or wearable devices focusing on the individual to large scale installations that focus on the general public. Each interactive element may contribute significant overhead and complexity to users' interactions, if it has a different mode of interaction from other devices and fits badly with people's everyday activities. Reducing this overhead and accommodating varying form of interaction is central to the design of ubiquitous environments. A major challenge for ubiquitous computing is to discover how people arrive at patterns of interaction that are productive rather than problematic. Interaction through Environments

Ubiquitous computing environments will not only provide new technologies for us to interact with; they will significantly affect our interactions with each other. Their communication technologies add to the means by which people in shared and different locations can interact, beyond email, telephone, letters, and so forth. A key element of these environments is that people can interact with each other through a hybrid mix of technologies and interaction devices, including multi-media and multi-modal technologies. The pace of interaction is likely to be closer to that of spoken conversation than a written exchange of letters; people may see and hear enough of each other to become aware of each other's ongoing context and activity. How do we go about understanding these different patterns of communication, and begin to describe and reason about such systems in use? Moreover, how will ubiquitous computing environments affect the existing frameworks and understandings that depend upon shared physical spaces, given that the communication technologies change many properties of these spaces?

- Interaction Techniques

We have to explore a range of new techniques that support interaction with and through diverse new devices and sensing technologies. These include gesture-based approaches exploiting movement in relation to surfaces and artefacts, haptic approaches exploiting the physical manipulation of artefacts, and speech-based interfaces. We need to support interaction and collaboration at multiple scales, from small–scale portable devices through to large–scale interfaces that are designed to support public interaction. As new interactive technologies and materials emerge we need to consider their effect on people's interaction. We need to consider how interaction techniques scale up to support the combined use of multiple interactive elements. We need to not only explore direct and engaged interaction, but also indirect interaction whereby sensors detect and interpret our actions to drive applications. The key challenge is how to admit a diverse and growing set of interactive techniques, while

ensuring that the world we inhabit remains coherent. Theories of Interaction We need to develop conceptual frameworks for evaluating, describing and understanding interaction with and through ubiquitous computing environments.. This requires a broadening of our traditional interdisciplinary frameworks to admit the influence of ubiquitous computing. This implies that we should treat ubiquitous computing as part of language and culture, and opens up powerful associations with other disciplines that handle activity, space and structure. Existing disciplines have developed frameworks useful for design or explanation, and researchers have already drawn upon these disciplines in order to discuss, shape and predict the use of ubiquitous computing. Philosophy of language and interpretation, semiology, linguistics, activity theory, situated action, distributed cognition, ethnographic studies, architecture and urban design theory have become important resources for ubiquitous computing. The challenge is to develop these explorations, in order to provide conceptual and theoretical tools for understanding activity in ubiquitous computing, and relate them to existing approaches in Human Computer Interaction (HCI) and Computer Supported Cooperative Work (CSCW).

## 4. Rigorous Protocol Design in Ubiquitous Computing

Communication protocols will form a key part of any ubiquitous computing system. Traditional Internet communication is dominated by the UDP and TCP transport protocols, together with various routing protocols, above IP. These rely on properties of the existing network — relatively stable connectivity, loss dominated by router congestion, and so on that will not hold for the variety of network technologies in ubiquitous systems. New protocols will be needed.

If these are to be predictable and robust they must be well-understood, for which new design techniques are also needed. Traditional internet protocol design is largely based on natural language specifications and interoperability testing between implementations. This often leads to unnecessary complexity and subtle flaws and implementation differences.

We therefore have an opportunity and test-bed for an integrated systems and semantics approach to protocol design. The goal of this foothill project is to establish a suite of protocols for particular GUC scenarios, developing and using rigorous design techniques for the purpose. It may build on the Netsem project, which has demonstrated a viable approach to the formal specification of existing real-world protocols, expressing their behaviour with operational semantics in the automated proof assistant HOL and developing symbolic modelchecking techniques to validate the specification against captured traces.

The main challenges are:

• The large-scale systems question of broadly what protocols are required, their APIs and design principles.

• Establishing idioms for expressing detailed design that:

o are an effective means of communication in the design and implementation teams, among those with theoretical and practical backgrounds; o support direct and automated conformance testing of production implementations against the protocol designs; and

o can be refined (i.e. resolving any looseness in the specifications) to give prototype implementations that can be used for experimentation and simulation.

• Establishing higher-level (more abstract) models that are suitable for approximate (probabilistic or stochastic) reasoning and simulation, ideally with mathematically rigorous relationships to the detailed designs.

• Verifying (with automated proof and/or model-checking) properties of both detailed and high-level models.

• Deploying the protocols and gaining experience in their use.[8][

## 5. The FOCALE

FOCALE stands for Foundation – Observe – Compare – Act – Learn – rEason, which describes its novel control loop. FOCALE is a model-driven architecture that can dynamically generate code to (re)configure managed elements using a context-aware policy model [14]. The FOCALE context-aware policy model is ideally suited for UbiComp applications, as it enables FOCALE to use context to select policies to orchestrate behavior. A simplified block diagram of FOCALE is shown in Figure 1.

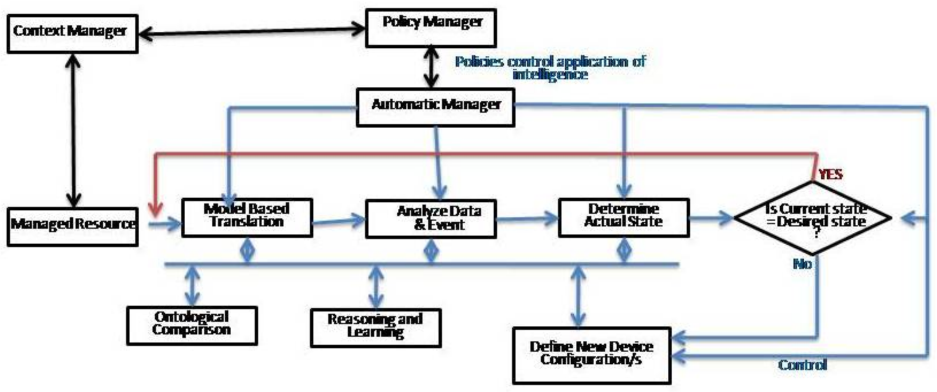

Figure 1. Simplified FOCALE Autonomic Architecture [1]

Sensor data is retrieved from the managed resource (e.g., a router) and fed to a model-based translation process, which translates vendor- and device-specific data into a normalized form in XML using the DEN-ng information model and ontology as reference data [17]. This is then analyzed to determine the current state of the managed entity. The current state is compared to the desired state from the appropriate Finite State Machines (FSMs). Nodes in a FOCALE FSM represent a configuration state; each state has an associated set of one or more configuration actions that define the configuration of an entity. Edges represent state transitions, and connote permission to change the configuration of a managed resource. Static behavior is thus "programmed" into FOCALE by designing a set of FSMs; dynamic behavior is defined by altering one or more FSMs. Context-aware policy management [16] governs the autonomic control loop. This enables context to select the set of policies that are applicable; policies are used to then define the functionality allowed. As context changes, policies change, and system functionality is adjusted accordingly. This is shown in [1]

Each aspect of context, such as time and location, is modeled by a ContextData instance, whereas the Context instance represents the final assembled context with all of its different aspects. Both ContextData as well as Context can affect the set of policy rules that are currently used to govern behavior; this enables granular decisions that are dependent on one or more aspects of Context to change policy rules as well as a higher level change in the entire context to change policy rules. These two changes can also be linked to the changing of state in one or more FOCALE FSMs; this is provided in a two step process. The first step defines the set of policy rules that are used to determine how context affects a State, and the second defines the set of actions for either maintaining that State or transitioning to a new State. These are defined by the two association classes PolicyDeterminesContextDataStateDetails and PolicyDeterminesContextStateDetails, and by the two associations ContextDataHasState and ContextHasState. FOCALE defines a normalized network management lingua franca by mapping vendor-specific data and commands to a vendor-neutral form based on a novel combination of information and data models augmented by ontologies [15]. Information and data models represent facts; these are augmented with semantics to enable machine-based learning and reasoning. It then uses a *model-based translation function* to interact with vendor-specific languages and programming models.

FOCALE supports a dynamically updateable knowledge base – one that can reflect new knowledge at runtime as new knowledge is discovered. Our approach supports this requirement by using semantic reasoning to examine sensor data to see if it is new as well as to determine if it is different (and especially, if it leads to different conclusions) than that stored in the knowledge base. In either case, the semantic reasoning uses first order logic to reason about the validity of the new or changed information with respect to the rest of the knowledge base. In other words, given new or changed information, the system must determine (1) if the new or changed information is valid, and (2) if it is valid, then how much of the existing knowledge base needs to be updated?

The axioms and theories present in the existing knowledge base are used to validate if the new or changed data makes logical sense. This makes use of existing data and relationships in the knowledge base to build assertions and other types of queries to test the implications of the new or changed data. Once the new or changed data are determined to be valid, then additional logic checks the relationships of the changed data to see if those data also need to be changed. Similarly, existing axioms and theories are applied to the new data to hypothesize new relationships. In general, the new or changed data will either be able to be immediately verified through issuing queries that verify one or more hypotheses about the new or changed data, or they will need further proof. In the former case, the new or changed data are immediately added to the knowledge base. Otherwise, they are marked for verification. This is beyond the scope of this paper; however, the essential point is that this set of processes enables the knowledge base for our system to evolve with experience. Both this and the model-based translation function use the notion of semantic relatedness [18] to determine the relevance as well as the validity of the sensor information as well as inferences derived from those data. Semantic relatedness enables entities that are semantically related using synonymy (e.g., “bank” and “lending institution”), antonymy (e.g., “accept” and “reject”), and other lexical relationships such as meronymy (e.g., court is a part of government), as well as defined associations (e.g., router uses protocol). Our original work in this area used linguistic analysis; however, this has a high associated degree of computational complexity. We are thus investigating other means, such as using WordNet [19], which provides a set of APIs for computing common linguistic relations, as well as structural matching algorithms. This enables us to move from offline applications, which require on the order of 2-6 hours of computation, to more near-real-time applications. FOCALE develops and uses a library of models and coded behaviors, much as a library of string processing functions is used by a programming language. This library is made reusable by realizing it in the form of objects, supported by both models and ontologies. The library is a result of the application of policy actions, which in turn are selected by a particular context as previously described. FOCALE uses the concept of the Policy Continuum [17][20][21], which enables policies written using terminology and concepts for one domain, such as business analysts, to be translated to policies written using a different set of terminology and concepts for another domain, such as programmers. [15] develops a Knowledge Continuum that, similar to the Policy Continuum, defines a continuum for expressing knowledge, from conjectures through facts. This enables context-aware policies to be used to orchestrate behavior for business goals, social interaction, and other forms of interaction.

## 6. The Internet Management Architecture for Ubiquitous Computing

We believe that current and future networks and networked applications have vastly different requirements; this means that a single architecture cannot simultaneously meet these different needs. Our approach for designing Future Networks is to build a set of service-aware networks that can exchange semantic information to enable their collaboration. We realize serviceaware networks by extending the FOCALE architecture to manage the difficult demands of UbiComp services as a model for Future Internet services. This takes the concepts of the previous section and defines a set of semantic translators that enable data from a variety of sources to be reasoned about and converted into a normalized form for further processing by the autonomic manager in FOCALE. This is shown conceptually in Figure 2

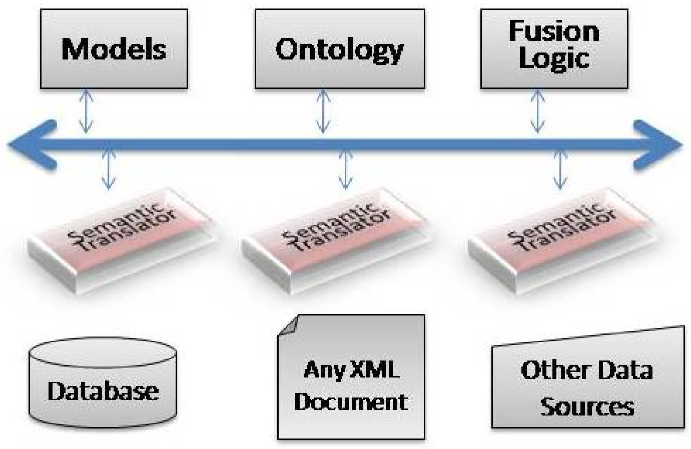

Figure 2. Semantic Translation and Processing [1]

The Fusion Logic in Figure 5 enables vendor-specific data to be translated into machine-understandable semantics. Thus, instead of trying to share fault and other types of data between domains, our approach instead exchanges semantic data describing the effect of those data. For example, consider two network devices from different network manufacturers. Each cannot understand the configuration data of the other. Our approach understands what the configuration is being used for, and instead passes the appropriate semantic data such that the FOCALE autonomic manager can generate appropriate vendorspecific configuration data.

Existing networks are divided into administrative domains. We support and enhance this idea, by ensuring that each domain has a set of context-aware policies that it uses to govern the functionality that it offers. This is shown conceptually in Figure 3. The concept of the current data plane is enhanced by supporting virtual devices and services. Hence, in Figure 6, a physical device can support multiple virtual devices, any of which can be part of an individual data plane. It is important to note that the administrative domain includes a set of physical devices, and therefore the associated virtual devices that are hosted by those physical devices. The set of physical devices that are subject to the same policies (which are now context-aware) are grouped into administrative domains; a FOCALE instance governs each administrative domain using appropriate control mechanisms that are coordinated by autonomic management loops. This is similar to the concept of existing control planes, except that now, each of the different control plane functions are governed by a single management function. In addition, autonomic control planes differ in that they are able to self-manage and -configure many different functions based on semantics, a feature that is notably lacking in existing control plane implementations. Finally, a FOCALE instance orchestrates the overall functionality of the administrative domain according to business goals and objectives; these are represented using the concept of the Policy Continuum, which enables policies written using terminology and concepts for one domain, such as business analysts, to be translated to policies written using a different set of terminology and concepts for another domain, such as programmers.

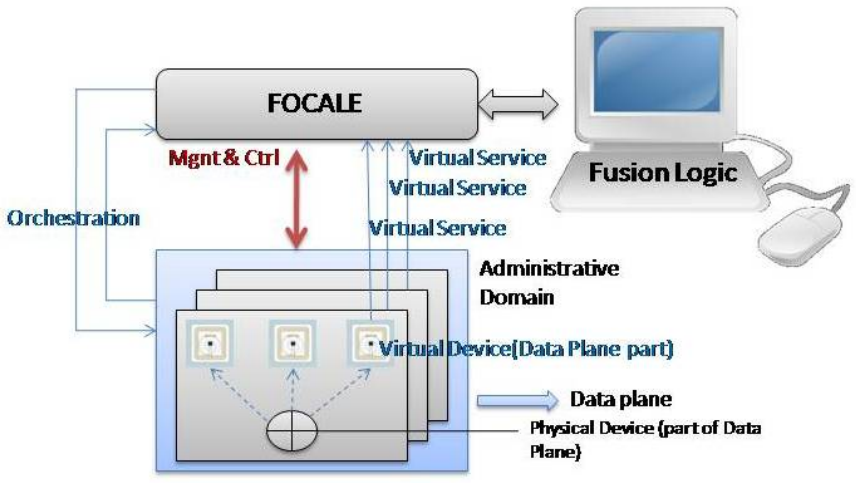

Figure 3. Service-Aware Domain for the Future Internet [1]

Figure 4 shows how service-aware networks are built. A set of administrative domains are concatenated to provide a data plane from the source to the desired destination(s). The set of resources are not important, as they are hidden from the user. Furthermore, the abstraction of an administrative domain is independent of service provider or other governing body; in fact, the only important feature of an administrative domain is that it uses context-aware policies to manage the services it provides. This paradigm can support current Internet services, which are dependent on a particular Service Provider that the user subscribes to, as well as a new genre of services that are not as restricted as this business model. This is inspired by UbiComp applications, where the user is interested not in a particular service provider, but rather in the data and services that can be used. These are represented conceptually as the set of Managed Service(s) that are delivered by the Master FOCALE instance, and is the subject of a separate paper. Each administrative domain is managed as described above and depicted in Figure 6, and exchanges semantic descriptions of its state, services, and other pertinent data to other FOCALE instances that are interested in the services and/or resources managed by it. This enables the FOCALE instances to collaborate and help each other provide high-value services.

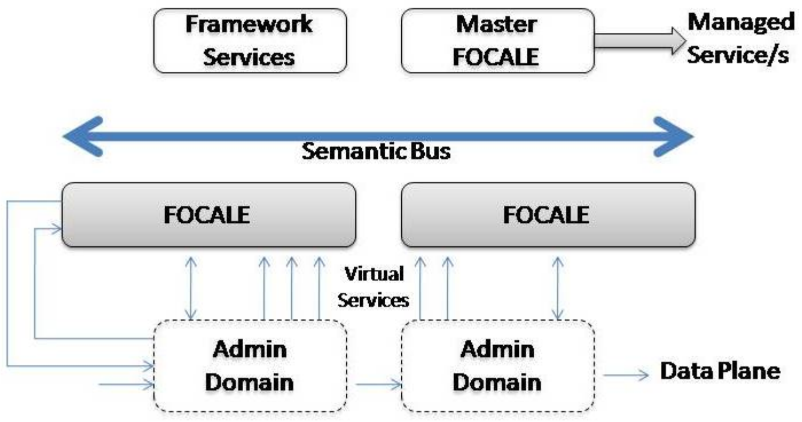

Figure 4. Service-Aware Future Internet Architecture [1]

A set of Framework Services provide the infrastructure necessary to support a distributed implementation. As a minimum, they include registration services (used to support location transparency), repository services (used to provide a consistent logical view of all managed entities in the system, according to access privileges of the user and policies of the system), naming services (used to generate and resolve unique names for entity instances contained in the repository), policy management services (used to coordinate the behavior of the entities in different administrative domains), and security services (used to ensure that the operation and behavior of the system does not violate any security policies). A set of Federation Services enables two types of federation to be performed: (1) based on services offering similar functions (service level federation), or (2) based on matching the underlying data (both system and user) to be shared (repository level federation).

The Framework and Federation Services are connected by a bus, so that the FOCALE system that is governing the set of lower-level FOCALE-based administrative domains can use the Framework and Federation Services as needed. The architecture is conceptually similar to a programmable grid, except that our architecture uses a model-based approach [12][15][17] to build reusable libraries of behavior that are then matched based on their semantics. This combines the features of Service Oriented Architectures with the ability to dynamically generate configuration changes based on business needs [15]. UbiComp applications express their needs to our system using a semantic description of the services that they want to use; this is then parsed and the system allocates appropriate resources based on first, the semantic description of any existing networks that support similar services; second, the ability to construct a service from its behavior libraries that will satisfy the needs of the application; third, the ability to dynamically build a new service that satisfies the needs of the application.

[26] described an implementation and associated simulation for FOCALE's policy-based architecture for end-to-end network management, while [27] describes a set of extensions to [26] that describe our model-based approach to policy tool and language generation. This approach defines a set of Domain Specific Languages to represent the needs of different constituencies, and makes use of a set of novel policy tools that were constructed from open source platforms. We are currently extending both of these implementations, as well as developing a set of policy languages, to automate the transformation of concepts between different levels of the Policy Continuum.

## 7. Conclusion

Next generation Internet demands of UbiComp applications. The model presented provides provides an evolutionary framework for managing Future Internet services by extending the FOCALE model-based approach, which dynamically generate code to make configuration changes. The approach addresses business, socio-political, and other aspects (in addition to technical aspects) by representing knowledge and policies along continua that enable different behavior to be orchestrated based on changing context. Lastly, conceptually it is an intelligent middleware that bridges the different languages and associated semantics used by existing network vendors and produces a single lingua franca that enables diverse data to be combined and concord.

## References

[1] John Strassner, Sven van der Meer, Brendan Jennings, Miguel Ponce de Leon,An Autonomic Architecture to Manage Ubiquitous Computing Networks and Applications, ICUFN 2009

[2] William Buxton and Tom Moran. EuroPARC's Integrated interactive intermedia facility *(iiif): early experiences*. North-Holland, 1990.

[3] George Calhoun. *Digital Cellular Radio*. Artech House Inc, 1988. 38

[4] A. Demers D. Terry, K. Petersen, M. Spreitzer, , M.M. Theimer, and B. Welch. Session guarantees for weakly-consistent replicated data. In *Proc. 3rd International Conference on Parallel and Distributed Information Systems*, pages 140–149, September 1994.

[5] A. Demers, K. Petersen, M. Spreitzer, D. Terry, M.M. Theimer, and B. Welch. The bayou architecture: Support for data sharing among mobile users. In *Proceedings Workshop on Mobile Computing Systems and Applications*. IEEE, December 1994.

[6] Alan Demers, Scott Elrod, Christopher Kantarjiev, and Edward Richley. A nanocellular local area network using near-field rf coupling. In *Proceedings of Virginia Tech's Fourth Symposium on Wireless Personal Communications*, pages 10.1–10.16, June 1994.

[7] Scott Elrod, Richard Bruce, Rich Gold, David Goldberg, Frank Halasz, William Janssen, David Lee, Kim McCall, Elin Pedersen, Ken Pier, John Tang, and Brent Welch. Liveboard: A large interactive display supporting group meetings, presentations and remote collaboration. In *Proc. of the Conference on Computer Human Interaction (CHI)*, pages 599–607, May 1992.

[8] http://www.cl.cam.ac.uk/users/pes20/Netsem

[9] Future Internet Forum (FIF), http://mmlab.snu.ac.kr/fif.

[10] M. Weiser, "Open House", in Review, the web magazine of the Interactive Telecommunications Program of New York University, March, 1996.

[11] J. Strassner, M. Ó Foghlú, W. Donnelly, N. Agoulmine, "Beyond the Knowledge Plane: An Inference Plane to Support the Next Generation Internet", IEEE Global Information Infrastructure Symposium (GIIS 2007), 2-6 July, 2007, pages 112-119.

[12] J. Strassner, N. Agoulmine, E. Lehtihet, "FOCALE – A Novel Autonomic Networking Architecture", ITSSA Journal, Vol. , No. 1, May 2007, pages 64-79, ISSN 1751-1461

[1 ] D. Clark, C. Partridge, J. Ramming, and J. Wroclawski, "A Knowledge Plane for the Internet", Proceedings ACM SIGCOMM 200

[14] J. Strassner, J. de Souza, D. Raymer, S. Samudrala, S. Davy, K. Barrett, "The Design of a New Policy Model to Support Ontology-Driven Reasoning for Autonomic Networking", LANOMS 2007, pages 114- 125, ISBN 9781424411825

[15] J. Strassner, “Enabling Autonomic Network Management Decisions Using a Novel Semantic Representation and Reasoning Approach”, Ph.D. thesis, 2008

[16] J. Strassner, Y. Liu, M. Jiang, J. Zhang, S. van der Meer, M. Ó Foghlú, C. Fahy, W. Donnelly, “Modelling Context for Autonomic Networking”, 5th IEEE International Workshop on Management of Ubiquitous Communications and Services (MUCS), April 11, 2008, Brazil

[17] J. Strassner, “Autonomic Networking – Theory and Practice”, 20th Network Operations and Management Symposium (NOMS) 2008 Tutorial, Salvador Bahia, Brazil, April 7, 2008

[18] A. Budanitsky, “Lexical semantic relatedness and its application in natural language processing” Technical Report CSRG 90, University of Toronto, 1999

[19] http://wordnet.princeton.edu/

[20] J. Strassner, “Policy Based Network Management”, Morgan Kaufman, ISBN 1-55860-859-1