

Coordination in Pervasive Computing Environments

Alois Ferscha

Johannes Kepler Universität Linz, Institut für Praktische Informatik
Altenberger Straße 69, 4040 Linz
ferscha@soft.uni-linz.ac.at

Abstract

Computer science and engineering nowadays appears to be challenged (and driven) by technological progress and quantitative growth. Among the technological progress challenges are advances in sub-micron and system-on-a-chip designs, novel communication technologies, micro-electro-mechanical systems, nano and materials sciences. The vast pervasion of global networks, the growing availability of wireless communication technologies for the wide, local and personal area, and the evolving ubiquitous use of mobile and embedded information and communication technologies are indicators for accelerated quantitative growth. We perceive a shift from the “one person with one computer” paradigm, which is based on explicit human machine interaction, towards a ubiquitous and pervasive computing paradigm, in which implicit interaction and cooperation is the primary mode of computer supported activity. This, however, poses serious challenges to the conceptual architectures of computing, and related engineering disciplines in computer science.

1. Historical Background

“The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it” was Mark Weiser’s central statement in his seminal paper [43] in Scientific American in 1991. His conjecture, that *“we are trying to conceive a new way of thinking about computers in the world, one that takes into account the natural human environment and allows the computers themselves to vanish into the background”* has fertilized the embedding of ubiquitous computing technology into a physical environment which responds to people’s needs and actions. Most of the services delivered through such a “technology-rich” environment are services adapted to context, particularly to the person, the time and the place of their use. Along Weiser’s vision, it is expected that context-aware services will evolve, enabled by wirelessly ad-hoc networked, mobile, autonomous special purpose computing devices (i.e. “smart appliances”), providing largely invisible support for tasks performed by users. It is expected that services with explicit user input and output

will be replaced by a computing landscape sensing the physical world via a huge variety of sensors, and controlling it via a manifold of actuators in such a way that it becomes merged with the virtual world. Applications and services will have to be greatly based on the notion of context and knowledge, will have to cope with highly dynamic environments and changing resources, and will need to evolve towards a more implicit and proactive interaction with users.

A second historical vision impacting the evolution of pervasive computing claimed for an intuitive, unobtrusive and distraction free interaction with technology-rich environments. In an attempt of bringing interaction “back to the real world” [42] after an era of keyboard and screen interaction, computers started to be understood as secondary artefacts, embedded and operating in the background, whereas the set of all physical objects present in the environment were started to be understood as the primary artefacts, the “interface”. Instead of interacting with digital data via keyboard and screen, physical interaction with digital data, i.e. interaction by manipulating physical artefacts via “graspable” or “tangible” interfaces, was proposed. Inspired by the early approaches of coupling abstract data entities with everyday physical objects and surfaces like Bishop’s Marble Answering Machine, Jeremijenko’s Live Wire and Wellner’s Digital Desk [42], tangible interface research [18] has evolved, where physical artefacts are considered as both (i) representations and (ii) controls for digital information. A physical object thus represents information while at the same time acts as a control for directly manipulating that information or underlying associations. With this seamless integration of representation and control into a physical artefact also input and output device fall together. Placed meaningfully, such artefacts can exploit physical affordances suggesting and guiding user actions, while not compromising existing artefact use and habits of the user. Recent examples for “embodied interaction”, where input and output are fused into physical object manipulation, include architecture and landscape design and analysis [29], object shape modeling interfaces using brick like blocks or triangular tiles [18].

Although the first attempts of the ubiquitous and pervasive computing vision in the early nineties fell short due to the lack of enabling hard- and software

technologies, are now, about ten years later, viable due to technological progress and quantitative growth. Preliminarily suffering from a plethora of unspecific terms like “Calm Computing”, “Hidden/Invisible Computing”, “Ambient Intelligence”, “Sentient Computing”, “Post-PC Computing”, “Universal Computing”, “Autonomous Computing”, “Everyday Computing”, etc., the research field is now consolidating from its foundations in distributed systems and embedded systems, and is starting to codify its scientific concerns in technical journals, conferences, workshops and textbooks. This process, however, is by far not settled today, so that even the term “Pervasive Computing” must be regarded verdant.

2. The Basic Elements

The challenges of pervasive computing [1] are dominated by the ubiquity of a vast manifold of heterogeneous, small, embedded and mobile devices, the autonomy of their programmed behaviour, the dynamicity and context-awareness of services they offer, the ad-hoc interoperability of services and the different modes of user interaction upon those services. This is mostly due to technological progress like the maturing of wireless networking, exciting new information processing possibilities induced by submicron IC designs, carbon nano tube transistor technology, low power storage systems, smart material, and motor-, controller-, sensor- and actuator technologies. A future computing service scenario appears possible, in which almost every object in our everyday environment will be equipped with embedded processors, wireless communication facilities and embedded software to percept, perform and control a multitude of tasks and functions. Many of these objects will be able to communicate and interact with the background infrastructure (e.g. the Internet), but also with each other. Terms like “*context-aware*” [8] smart appliances [9] [37] and smart spaces [25][11][32] – have appeared in the literature to refer to such technology-rich environments. Context aware environments intelligently monitor the objects of a real world (like persons, things, places), and interact with them in a pro-active, autonomous, responsible and user-authorized way.

From an applied research prospect, pervasive computing is motivated to empower users through an environment that is aware of their presence, sensitive, adaptive [26] and responsive to their needs, habits and emotions, as well as ubiquitously accessible via natural interaction [36]. Pervasive computing applications are characterised by the following basic elements:

- (i) ubiquitous access,
- (ii) context awareness,
- (iii) intelligence, and
- (iv) natural interaction.

Ubiquitous access here refers to a situation in which users are surrounded by a multitude of interconnected embedded systems, which are mostly invisible and weaved into the background of the surrounding, like furniture, clothing, rooms, etc., and all of them able to sense the setting and state of physical world objects via a multitude of sensors. Sensors, as the key enablers for implicit input from a “physical world” into a “virtual world”, will be operated in a time-driven or event-driven way, and actuators, as the generic means for implicit output from the “virtual” to the “physical world”, will respond to the surrounding in either a reactive or proactive fashion.

Context awareness refers to the ability of the system to recognise and localise objects as well as people and their intentions. The context of an application is understood as “*any information that can be used to characterize the situation of an entity*”, an entity being “*a person, place or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves*” [8]. A key architecture design principle for context-aware applications will be to decouple mechanism for collecting or sensing context information [32] and its interpretation, from the provision and exploitation of this information to build and run context-aware applications [35]. To support building context-aware applications, software developers should not be concerned with how, when and where context information is sensed. Sensing context must happen in an application independent way, and context representation must be generic for all possible applications [14].

Intelligence refers to the fact that a technology-rich environment is able to adapt itself to the people that live (or artefacts that reside) in it, learn from their behaviour, and possibly recognise as well as show emotion. Natural interaction finally refers to advanced modalities like natural speech- and gesture recognition, as well as speech-synthesis which will allow a much more human-like communication with the digital environment than is possible today.

3. Enabling Technologies

Having identified the basic elements of pervasive computing systems, a closer look at technological implementation options is advised.

Ubiquitous access is promisingly implemented based on a wireless communication infrastructure involving broadband satellite systems, cellular radio communication (e.g. GSM, GPRS, TETRA, DECT, EDGE, UMTS/IMT2000), personal and local area radio communication (e.g. Bluetooth, HomeRF, IEEE802.11,

HiperLAN, HomeCast), infrared (IrDA) and ultrasonic communication. Above wireless communication networks, also power line and wireline communication (USB, IEEE1394, HomePNA) appear as ubiquitous access implementation options. The primary challenge here lies in the maintenance of seamless connections as devices move between different areas of different network technology and network connectivity. While communication problems like routing and handover can be handled at the network level, others cannot be solved at this level as they relate to the interaction semantics at the application level [10]. Device heterogeneity and differences in hardware and software capabilities requires a communication infrastructure that maintains knowledge about device characteristics and manages coherent device interactions (e.g. among wearable devices, home appliances and outdoor appliances). Personal area network (PAN) technologies are supposed to address these issues. The dynamic interconnection of devices and the discovery of services is approached with coordination software systems [28] like HAVI [22], Java/Jini [38], JXTA [39][17][WDKF 02], Java-Spaces, UPnP [5], Salutation [23], Tspaces, etc. While service registration and discovery, lookup services, self configuration, caching and differencing methods have working solutions today, context based networking [10] and the context based coordination of entities and activities must still be considered as research issues.

Above the technological options, the capability of an object to *identify*, *localize* and track other objects, and to *coordinate* its activities with respect to and relative to the other objects is essential in pervasive computing systems. A plenty of ready-to-use technologies for the automated recognition (identification) of real world objects can be accounted: technologies based on optical (barcode and OCR), magnetic (SmartCard), ultrasonic (Active Badge and iButton) sensors, voice and vision based systems, biometrical systems (fingerprint, retina, face recognition), etc. Many of those are also suitable for short distance positioning and tracking (localization), and are already in use for locator services in many different fields of application. Global positioning technologies based on GSM, GPS, dGPS extend the range of options for long distance localization. An identification and localization technology with a certain appeal for embedded pervasive computing applications is radio frequency identification (RFID).

Coordination, finally, is the way how the activities and interactions among objects in a pervasive computing system are organized from an architectural viewpoint. While early definitions of the term read like: „*Coordination is the process of building programs by gluing together active pieces*“ by the software engineers

Carriereo and Gelernter in 1990, „*Coordination is the integration and harmonious adjustment of individual work efforts towards the accomplishment of a larger goal*“ by Singh in 1992 and „*Coordination is the act of managing dependencies between activities*“ by Malone and Crowston in 1994, also more formal attempts have been made to characterise the architecture of activities in complex dynamic systems.

Ciancarini [6] defines a “coordination model” as a triple (E, M, L). E stands for the “coordinable entities”, the active components of a systems which are coordinated. In a very abstract sense, these can be humans, software processes or agents, data tuples etc. M are the coordinating media (or semantic connectors) which serve to aggregate the coordinable entities to form a “configuration”. Communication channels, shared variables, tuple spaces or multiset data structures can be instances of coordinating media. The coordination laws ruling the actions of coordinable entities are referred to by L. These laws define the semantics of the coordination mechanisms allowed in a coordination model. The linguistic embodiment of a coordination model into a programming language is called a coordination language, offering the syntactical means with which a coordination model can be used for implementing an application. Both data-driven models (Linda based models, models based on multiset rewriting) and process-oriented models (IWIM, Manifold, Contextual Coordination) of coordination have been proposed in the literature [28], and have been embodied into programming languages like PASCAL, Ada, C, C++, Smalltalk and Java. The concept of separating the programming concern (using programming languages) from the coordination concern (using coordination models) appears promising for the implementation of pervasive computing systems.

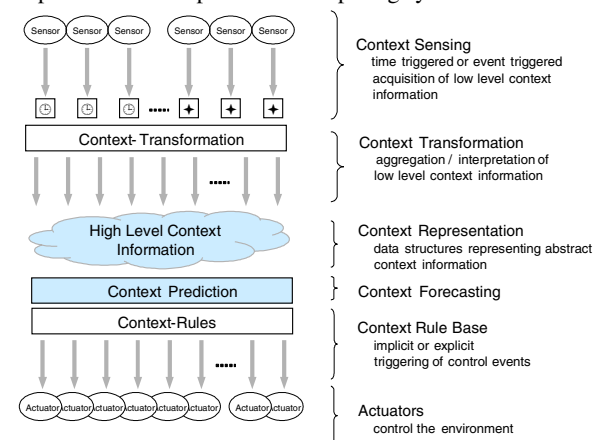


Figure 1: Framework Architecture for Context Aware Systems

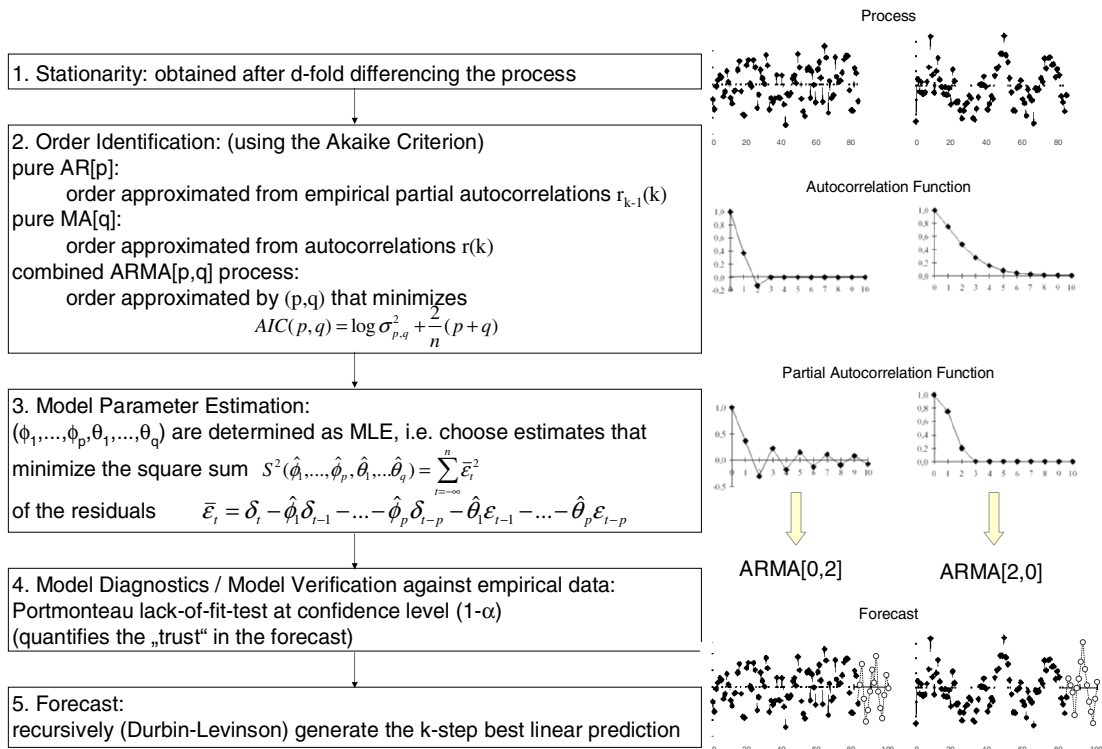


Figure 2: Context Prediction based on Sensor Data Time Series Analysis

4. A Software Framework for Context-Aware Applications

Observing the technology trends in the aforementioned domains of sensors and actuators, processing devices, embedded systems and wireless communication, we have proposed a framework architecture for building “context-aware” systems [16]. The adoption of a world model representing a set of objects and their state in the physical (or “real”) world is suggested, with mechanisms to sense, track, manipulate and trigger the real world objects from within the world model. Several frameworks for such world models have appeared recently [30][20][21][13], the most prominent ones [19] identifying *persons*, *things* and *places* as the primary abstract classes for real world objects. People living in the real world, acting, perceiving and interacting with objects in their environment are represented in the world model by “virtual objects” or “proxies”. Proxies of persons, things and places are linked to each other in the virtual world, such that this “linkage” is highly correlated with the “linkage” of physical persons, things and places in the real world. A context-aware application now monitors the state and activity of

the real world objects via a *set of sensors*, *coordinates* the proxies according to the rules embodied in the application, and notifies, triggers [3] or modifies the physical world objects via a *set of actuators*.

Within our framework, a set of software components acting as wrappers for physical sensor hardware is collecting low level sensor data, which is then transformed into high level context information, i.e. information that is semantically meaningful for the application. Context information is represented in the metadata model RDF (Resource Description Framework) [31] as RDF statements over instances of the abstract object classes person, thing and place, and their contextual interrelatedness.

Context information is also time indexed, allowing to express the aging of context information, and to analyse the “context history” of an object. With this approach we are also able to implement proactive behaviour based on predicted future states of the system. A self contained and fully automated context prediction system has been developed based on time series analysis and mechanisms for statistical forecasting [15]. The context prediction system assumes a stationary time series underlying the sensor data process, and uses Akaike’s criterion to

determine the order of a priori unknown Auto-Regressive and/or Moving Average (ARMA) components in the process. After identifying the ARMA model order, model parameters are estimated, and the resulting model is used to forecast future sensor data, based on which anticipated context states are calculated (Figure 2 explains the prediction systems and demonstrates the procedure for two independent sensor data processes, together with forecasted data values).

In our framework, the triggering of the software components acting as wrappers for the physical actuators is done based on exposing the current (or predicted) context state to a set of coordination rules [2]. ECA (event condition action) rules are used to invoke actuator actions upon the observation of an event, given that certain context conditions hold. An ECA rule management systems allows for the deployment of rules at runtime, thus supporting dynamic changes to the coordination architecture.

5. Demonstration Scenario: A Context-Aware Appliance

Since our framework aims to support the development of context-aware applications, we refer to it as “contextware” framework. To demonstrate the potentials of the contextware framework as well as the hard- and software engineering issue emerging when building pervasive computing systems, a context-aware suitcase has been developed. The hardware for the suitcase demonstration prototype uses an embedded single board computer integrated into an off-the-shelf suitcase (see Figure 3), which executes a standard TCP/IP stack and HTTP server, accepting requests wirelessly over an integrated IEEE802.11b WLAN adaptor. A miniaturized RFID reader is connected to the serial port of the server machine, an RFID antenna is integrated in the frame of the suitcase so as to enable the server to sense RFID tags contained in the suitcase. A vast number of 125 kHz and 13,56 MHz magnetic coupled transponders are used to tag real world objects (like shirts, keys, PDAs or even printed paper) to be potentially carried (and sensed) by the suitcase. The suitcase itself is tagged and possibly sensed by readers integrated into home furniture, car or airplane trunks, conveyor belts etc., so as to allow for an identification and localization at any meaningful point in space of the application.

Within the contextware framework, an application specific abstraction of the real world is generated from three generic classes for persons, things, and places. The person object represents the concepts about a person that are necessary to link their physical properties and activities to the virtual world. The thing object

encapsulates the basic abstraction for objects relevant to the application and has instances like shirt, key, etc. The place object is used to represent the essential features of a physical location like an office room or a waiting lounge. The contextual interrelatedness among those object instances is expressed by a set of (bilateral) object relations within RDF. The contextual situation of the suitcase is expressed by RDF statements, i.e. denoted as subject, predicate and object triples. A subject in RDF is represented by a resource, that could be a URI reference, a URL or even a simple HTML file. A predicate is represented by a property describing a resource or the relation among resources, and an object is represented by a property value. To represent state changes in the real world within the framework, sensors like the RFID reader embedded into the frame of the suitcase are used to dynamically modify the set of relations at runtime. Such relations are for example the *owns* relation, which expresses ownership of a real world object by another, the *contains* and *is_in* relations to express geometrical containment of an object within another one, the *contained* and *was_in* relations that trace the history of containment. The *containable* attribute defines whether an object may contain another, the *controllable* attribute allows to prevent or enable the modification of an object RDF by the object itself. The unique ID associated with every real world object is the ID encoded in its RFID tag. Once sensed by the RFID reader, it triggers a script to update the involved object RDFs. Inserting e.g. the shirt into the suitcase would cause the RFID reader to identify the shirt tag and to update (among others) both the shirts RDF relation *is_in*, as well as the suitcases RDF relation *contains* by cross-referring URIs.

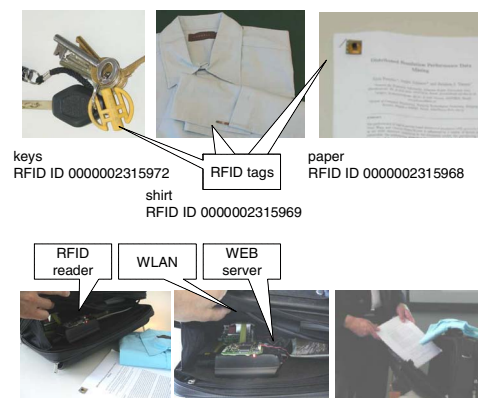


Figure 3: Context-Aware Suitcase hardware and RFID Tagged Objects

Since all the contextual information about the suitcase and the related objects are updated instantly upon the observation of events that change the real world situation, and since the framework makes this information

accessible via HTTP, the demonstrator can safely be said to possess all important characteristics of a pervasive computing system: it is ubiquitously accessible (from almost wherever it resides and from any node in the Internet), it is context aware in that it can reason about its current situation and history, it offers natural interaction to its users, and might even exhibit “smart” behaviour (intelligence) e.g. when notifying or reminding the user.

6. Conclusions

Recent advances in microprocessor-, communication- and sensor-/actuator technologies envision a whole new era of computing, popularly referred to as “pervasive computing”. Autonomous computing devices in great number and embedded into everyday objects will deliver services adapted to the person, the time, the place – or most generally: the context – of their use. The nature of computing devices will change to invisible networked, augmented environments, in which the physical world is sensed and controlled in such a way that it becomes merged with a “digital world” [12]. According to D. Norman, computing devices will become more specialized in purpose, and will be designed to serve a well defined set of tasks only [27].

Augured as a fertile source of research challenges [33], pervasive computing has at least started to broaden the discourse on principles and methods in distributed computing, mobile computing and embedded systems. We encounter the need for a commonly agreed reference model of computation upon which software abstractions can be built: much like the von Neumann architecture and the concept of the von Neumann variable supported the creation of programming languages, compiler technologies and (machine independent) algorithm development. A first evidence for such a reference model goes back to Alan Turing’s paper “*On Computable Numbers with an Application to the Entscheidungsproblem*”, which delivered a foundational result to the computer science community (which was not in existence at that time): computers cannot completely prove (all) mathematical assertions. It proved the inability of Turing machines to solve (all) mathematical problems, and later (with Church) “confined” the abilities of computers to the class of algorithmically solvable problems. Theoretical computer science adopted the Turing machine model, and a computer science of “algorithms” emerged (since the 1950s), viewing computation as the transformation of input (data) to output (data) by executing algorithms. Computer science has greatly expanded since then, and so have the technological opportunities to build computers and computer applications. At least from what was framed within this paper, it should be obvious, that traditional computational models that explain system components in

terms of “what they compute” appear less expressive than interaction models expressing how components “interact”, and how these interactions are coordinated”. But this observation, and claim for a model able to express “interaction” is (and was) by no means new, and many approaches to extend computation beyond algorithmics (beyond Turing machines) have been attempted. One such concept was Robin Milner’s “Elements of Interaction” [24] originating from the CCS “concurrency” thought model, another one was Peter Wegner’s “Interaction Machines” [42][40] originating from an “interaction” thought model. In our own work we have proposed a “spontaneous interaction” thought model, in which things start to interact once they reach physical proximity to each other: Explained using the metaphor of an “aura”, which like a subtle invisible emanation or exhalation radiates from the center of an object into its surrounding, a “digital aura” is built on technologies like Bluetooth radio, RFID or IrDA together with an XML based profile description, such that if an object detects the proximity (e.g. radio signal strength) of another object, it starts exchanging and comparing profile data, and, upon sufficient “similarity” of the two profiles, starts to interact with that object (see Figure 4). A “digital aura” depending on the implementation technology, is dense in the center of the object, and thins out towards its surrounding until it is no longer sensible by others. Profiles described as semistructured data and attached to the object, can be matched by a structural and semantic analysis. Peer-to-peer concepts can then be used to implement applications on top of the digital aura model for spontaneous interaction.



Figure 4: A Digital Aura Approach for Spontaneous Interaction

We encounter the need for a separation of concerns: formal models and syntactical means are sought to engineer “coordination”, much like formal languages are needed to engineer “computation”. “Contextware” is needed to build context-aware systems able to describe, gather, transform, interpret, disseminate and use context information. Finally, both conceptual models and methods are needed to implement “awareness” and “intelligence”.

7. References

- [1] G. Banavar et al.: Challenges: An Application Model for Pervasive Computing, *Proceedings MobiCom 2000*.
- [2] W. Beer, V. Christian, A. Ferscha, L. Mehrmann: Modeling Context-Aware Behavior by Interpreted ECA Rules, *Euro-Par 2003*, Springer Verlag, LNCS, 2003.
- [3] P. J. Brown: Triggering Information by Context. *Personal Technologies*, Vol. 2, Nr.1, pp. 18-27, 1998. <http://www.cs.ukc.ac.uk/pubs/1998/591/content.html>
- [4] D. Caswell, P. Debaty: Creating a Web Representation for Places. *WWW 2000*.
- [5] B. Christensson, O. Larsson: Universal Plug and Play Connects Smart Devices. *WinHEC 99*, 1999.
- [6] P. Ciancarini: Lectures on Coordination Models and Languages, 1997.
- [7] J. L. Crowley, J. Coutaz, F. Berard: Perceptual user interfaces: Things That See. *Communications of the ACM*, Vol. 43, Nr. 3, pp. 54-64, 2000.
- [8] A. K. Dey: Understanding and Using Context. Personal and Ubiquitous Computing, *Special Issue on Situated Interaction and Ubiquitous Computing*, 5(1), 2001.
- [9] K. F. Eustice, T. J. Lehman, A. Morales, M. C. Munson, S. Edlund, M. Guillen: A universal information appliance. *IBM Systems Journal*, Vol. 38, Nr. 4, 1999.
- [10] M. Esler et al.: Next Century Challenges: Data Centric Networking for Invisible Computing. *MobiCom'99*.
- [11] I. Essa: Ubiquitous Sensing for Smart and Aware Environments: Technologies toward the building of an Aware House. *DARPA/NSF/NIST Workshop on Smart Environments*, 1999.
- [12] D. Estrin, D. Culler, K. Pister, G. Sukhatme: Connecting the Physical World with Pervasive Networks. *Pervasive Computing*, Vol. 1 No. 1, pp. 59-69, 2002.
- [13] Future Computing Environments: The Context Toolkit. <http://www.cc.gatech.edu/fce/contexttoolkit/>
- [14] A. Ferscha, W. Beer, W. Narzt: Location Awareness in Community Wireless LANs. *Proceedings of the Informatik 2001: Workshop on Mobile internet based services and information logistics*, 2001.
- [15] A. Ferscha: Adaptive Time Warp Simulation of Timed Petri Nets, *IEEE Transactions on Software Engineering*, Vol. 25, No. 2, pp. 237-257, 1999.
- [16] A. Ferscha: Contextware: Bridging Virtual and Physical Worlds. *Reliable Software Technologies, AE 2002*. Springer Verlag, LNCS 2361, pp 51-64, 2002.
- [17] L. Gong: Peer-to-Peer Networks in Action. *IEEE Internet Computing*, Vol. 6, No. 1, pp. 36-39, 2002.
- [18] Gorbet, M.G., Orth, M., Ishii, M.: Triangles: Tangible Interface for Manipulation and Exploration of Digital Information Topography. *CHI 1998*, pp. 49-56, 1998.
- [19] Hewlett Packard: CoolTown Appliance Computing: White Papers. <http://cooltown.hp.com/papers.htm>
- [20] T. Kindberg, J. Barton, J. Morgan, G. Becker, I. Bedner, D. Caswell, P. Debaty, G. Gopal, M. Frid, V. Krishnan, H. Morris, J. Schettino, B. Serra, M. Spasojevic: People, Places, Things: Web Presence for the Real World. *WWW'2000*.
- [21] F. Kon, C. Hess, Christopher, M. Roman, R. H. Campbell, M. D. Mickunas: A Flexible, Interoperable Framework for Active Spaces. *OOPSLA'2000 Workshop on Pervasive Computing*, 2000.
- [22] R. Lea, S. Gibbs, A. Dara-Abrams, E. Eytchison: Networking Home Entertainment Devices with HAVi", *Computer*, Vol. 33, No. 9, 2000.
- [23] B. Miller: Mapping Salutation Architecture APIs to Bluetooth Service Discovery Layer. Bluetooth Consortium I.C.118/1.0, 1999.
- [24] R. Milner: Elements of Interaction: Turing Award Lecture, *CACM*. Vol. 36, No. 1, pp.78-89, 1993.
- [25] MIT Media Lab: Smart Rooms. <http://ali.www.media.mit.edu/vismod/demos/smartroom>
- [26] M. C. Mozer: An intelligent environment must be adaptive. *IEEE Intelligent Systems and Their Applications*, Vol. 14, No. 2, pp. 11-13, 1999.
- [27] D. Norman, "The invisible computer: why good products can fail, the personal computer is so complex, and information appliances are the solution", MIT Press, 1998.
- [28] A. Omicini, F. Zambonelli, M. Klusch, R. Tolksdorf (Eds.): Coordination of Internet Agents. Springer, Berlin, 2001.
- [29] B. Piper, C. Ratti, H. Ishii: Illuminating Clay: A 3-D Tangible Interface for Landscape Analysis. *CHI 2002*, pp. 355-362, 2002.
- [30] J. Pascoe, N. Ryan, D. Morse: Issues in developing context-aware computing. In H-W.Gellersen, editor, *Handheld and Ubiquitous Computing*, Springer-Verlag LNCS, pp. 208-221, 1999.
- [31] O. Lassila, R. R. Swick: Resource Description Framework (RDF): Model and Syntax Specification. Recommendation, World Wide Web Consortium, 1999. <http://www.w3c.org/TR/REC-rdf-syntax/>.
- [32] M. Roman, R. H. Campbell: GAIA: Enabling Active Spaces. *9th ACM SIGOPS European Workshop*. 2000.
- [33] D. Salber, A. K. Dey, R. Orr, G. D. Abowd: Designing for Ubiquitous Computing: A Case Study in Context Sensing. GVU, Technical Report GIT-GVU-99-29, July 1999.
- [34] M. Satyanarayanan: Pervasive Computing: Vision and Challenges. *IEEE Personal Communications*, pp. 10 – 17, 2001.
- [35] W. N. Schilit: *A System Architecture for Context-Aware Mobile Computing (Ph. D. Dissertation)*. New York: Columbia University, 1995.
- [36] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. Laerhoven, W. V. Velde: Advanced Interaction in Context. *HandHeld and Ubiquitous Computing*, 1999.
- [37] A. Schmidt, K. Laerhoven: How to Build Smart Appliances?, *IEEE Personal Communications*, Vol. 8, No. 4, pp. 66-71, 2001.
- [38] J. Waldo: Jini Technology Architectural Overview, White Paper, Sun Microsystems, Inc., January 1999.
- [39] St. Waterhouse, D. M. Doolin, G. Kan, Y. Faybishenko: Distributed Search in P2P Networks. *IEEE Internet Computing*, Vol. 6, No. 1, pp. 68-72, 2002.
- [40] P. Wegner: Why Interaction is more powerful than algorithms. *CACM*, Vol. 40, No. 5, pp. 80-91, 1997.
- [41] P. Wegner, D. Goldin: Computation Beyond Turing Machines. Technical Opinion. *CACM*, Vol. 46, No. 4, pp. 100-102, 2003.
- [42] P. Wellner, W. Mackay, R. Gold: Computer Augmented Environments: Back to the Real World. *CACM*, Vol. 36, No. 7, 1993.
- [43] M. Weiser: The Computer of the Twenty-First Century. *Scientific American*, pp. 94-100, 1991.