What is Pervasive Computing?

Alois Ferscha

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

> *"Dazu wurden volltönende neue Begriffe geprägt,*
> *die bei näherer Betrachtung recht dürftig gewandet sind,*
> *zum Teil sogar sich als Leerformel herausstellen*."
>
> Peter Rechenberg

Abstract

To the inspired observer, computer science 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 over the past years, the growing availability of wireless communication technologies in the wide, local and personal area, and the evolving ubiquitous use of mobile and embedded information and communication technologies are examples for challenges posed by quantitative growth. We perceive a shift from the "one person with one computer" paradigm, which is based on explicit man machine interaction, towards a ubiquitous and pervasive computing landscape, in which implicit interaction and cooperation is the primary mode of computer supported activity. This change – popularly referred to as "Pervasive Computing" – poses serious challenges to the conceptual architectures of computing, and the related engineering disciplines in computer science. In this work, I will reflect on some of the emerging pervasive and ubiquitous computing trends and potentials, and in particular on the software engineering issues associated with the provision of context aware systems.

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 [Weis 91] 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”* [WMG 93] 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 [WMG 93], tangible interface research [GOI 98] 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 [PRI 02], object shape modeling interfaces using brick like blocks or triangular tiles [GOI 98].

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. Pervasive computing initiatives and projects have emerged at major universities worldwide, and national and international research funding authorities (IST Future and Emerging Technologies programme of the EU, DARPA, NSF, etc.) have accelerated the efforts of a rapidely growing, vibrant research community. Preliminarily suffering from a plethora of unspecific terms like “Calm Computing”, “Hidden or 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.

The Basic Elements

The challenges of pervasive computing [Bana 00] 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*" [Dey 01] smart appliances [ELMM 99] [ScLa 01] and smart spaces [MIT] [Essa 99] [RoCa 00] – 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, souvereign, 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 [Moze 99] and responsive to their needs, habits and emotions, as well as ubiquitously accessible via natural interaction [SATT 99]. 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 [RCDD 98] 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*” [Dey 01]. A key architecture design principle for context-aware applications will be to decouple mechanism for collecting or sensing context information [SDOA 99] and its interpretation, from the provision and exploitation of this information to build and run context-aware applications [Schi 95]. 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 [FBN 01].

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.

Enabling Technologies

Having identified the basic elements of pervasive computing systems, and being concerned with the provision and establishment of software methods and development processes for such 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 software related 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 [Else 99]. 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 theses issues. The dynamic interconnection of devices and the discovery of services is approached with coordination software systems [OZKT 01] like HAVI [LGDE 00], Java/Jini [Wald 99], JXTA [WDKF 02][Gong 02], Java-Spaces, UPnP [ChLa 99], Salutation [Mill 99], Tspaces, etc. While service registration and discovery, lookup services, self configuration, caching and differencing methods have working solutions today (see Table 1 for a comparison), context based networking [Esle 99] and the context based coordination of entities and activities must still be considered as research issues.

	Jini/Java	UPnP	Salutation	Tspaces
Originator	SUN Microsystems	primarily Microsoft	Industry / Academic consortium	IBM
Register / Announcing Presence:	Unicast/Multicast to Jini lookup service Lease expiry for up to date information	SSDP Protocol, working with or without Proxy Service Unicast/Multicast HTTP - UDP	through a Salutation Manager (SM) works as a service broker	making services available on a Tuple Space
Discovering other devices	Querying lookup services Optionally using an RMI proxy	SSDP (Simple Service Discovery Protocol) used to announce their presence as well as find other devices	queries on SM	Database indexing and query on the TupleSpace
Describing capabilities	Registration Information can have attribute pairs RMI interface with known methods	XML documentation is made available URIs are used	functional units which know features of a service	Java RMI Method invocation interfaces with known methods
Self configuration	not considered	DHCP or AutoIP multicast DNS	not considered, difficult because of transport independence	Not considered
Invoking services:	Through the lack of device driver difficult, device must be Java enabled, RMI	Invoking Services via Browser, or direct API calls through applications	flexible, vendor specific, realised through SM	Java RMI framework
Security:	Java Security	not finished yet	allows user authentification	access controls by User
Transports:	TCP/IP – Proxies	TCP/IP – Proxies	transport independent	TCP/IP
Supported Devices	all kind of devices, mobile, household equipment	all kind of devices, mobile, household equipment	printer, FAX, telephone, PDA , computer (of any size) but not limited to them	Printers, PDAs, Computers
Remarks	no self configuration makes it weak RMI Method invocation is difficult to handle + Security	self configuration is considered as the strongest feature of this technology	advantages in the functionality of the SMs, problems with transport independence	enhancement of Linda TupleSpace concept

TABLE. 1 A COMPARISON OF APPROACHES FOR “COORDINATION”

As far as the ubiquitous access to the Internet is concerned, a major enabling technology is the miniaturization of devices able to accept and to respond to HTTP requests, i.e. the provision of Web servers to be weaved into arbitrary fabrics of daily use objects. Figure 2 demonstrates popular embedded webserver miniatures (as they are readily available from e.g. HYDRA, Dallas Semiconductors, Tini, Boolean, etc.), envisioning the technological options for e.g. embedded Internet appliances.

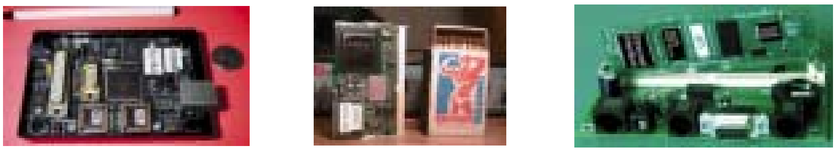


FIGURE. 2 MINIATURIZED EMBEDDED AND ON-CHIP WEBSERVERS

New output technologies like light emitting polymers (see Fig. 3, right), foldable displays, laser diodes, electronic inc, electronic paper etc. together with new materials able to serve as sensors or input technologies for monitoring the environment (digital and analog), appliances like wearable computers, interactive workspaces like smart rooms, walls, tables, chairs etc., spawn a whole new space for information appliance designs. The implementation of implicit input and output will involve sensor and actuator technologies, with sensor technologies ranging from optical and opto-electrical (photodiodes, -conductors, charge coupled devices, light-emitting diodes and polymers, lasers, liquid crystals), acoustic (e.g. acoustic and sonar transducers, surfaces acoustic wave devices), mechanical (silicon pressure sensors and accelerometers, solid-state displacement transducers, piezoelectric field-effect transducers (see Fig. 3, left)), to thermal, magnetic and chemical sensors. The maturing of voice-vision-text technologies and gesture, eye- and body-movement recognition technologies gives further ground to a "continuously present" human computer interface.

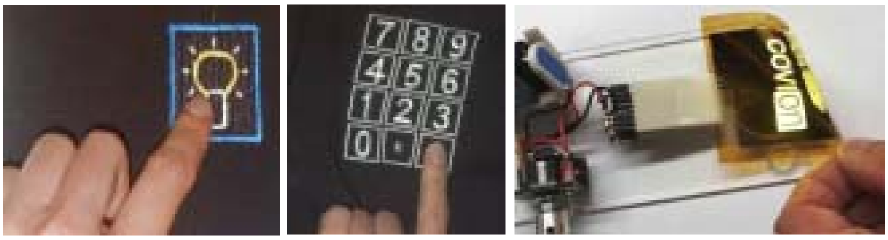


FIGURE. 3 TECHNOLOGIES FOR "NATURAL INTERACTION"

Above these technologies, the capability of an object to *identify*, *localize* and track other objects, and to *coordinate* its activites with respect to and relative to the other objects is essential in pervasive computing systems:

(i) Identification (sensing the identity of a real world object),
(ii) localization (sensing its position and movement in space), and
(iii) coordination (relating it semantically to other objects and behavioral rules)

are central issues for embedded applications, and, moreover, are essential to all context-aware applications. They are among the capabilities that are fundamentally different from conventional 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). An RFID system consists of a tag (or transponder), and a reader device. The transponder as a passive component responds by replying to an interrogation request received from an interrogator. The reader as an active component induces an interrogation request, and receives back data from the transponder (such as an identity code or the value of a measurement) at rates ranging from 1K to 16K bits/sec. Different frequencies of the radio system of the reader result in different reading ranges (10 cm to 1 m) and properties of the system. Commonly available tags have an operating frequency in the range from 60 kHz to 5.8 GHz, 125 kHz and 13,56 MHz, but MF, VHF, UHF and microwave frequencies are also being used. The reasons why transponders have recently received considerable attention in electronic identification are their contact-less and powerless operation, low cost, "unlimited" lifetime, ISO standard compliance (14443 A/B and 15693-2), a wide choice of form factors (cards, tags, inlets, smart labels, etc.), short range operation, proximity and vicinity communication with one and the same technology and cryptographic security (i.e. the protection against unauthorised product copies or data modification). Figure 4 displays contemporary RFID tags with 48 up to 2048 Bytes of read/writable memory capacity.

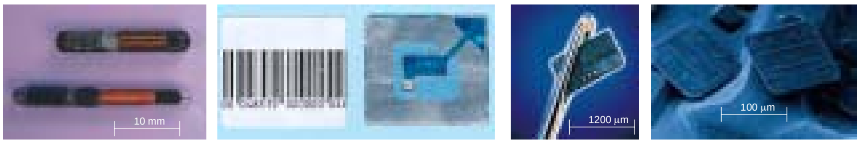


Figure. 4 Technologies for Identification: RFID

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 [Cian 97] 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, hese 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 hich a coordination model can be used for implementing an application. Both data-driven models (Linda based models, models based on muliset rewriting) and process-oriented models (IWIM, Manifold, Contextual Coordination) of coordination have been proposed in the literature [OZKT 01], 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.

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 (see Fig. 5) [Fers 02] [FVB 02]. 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 [PRM 99] [KBMB 00] [KHCR 00] [FCE], the most prominent ones [HP] 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 [Brow 98] or modifies the physical world objects via a *set of actuators*.

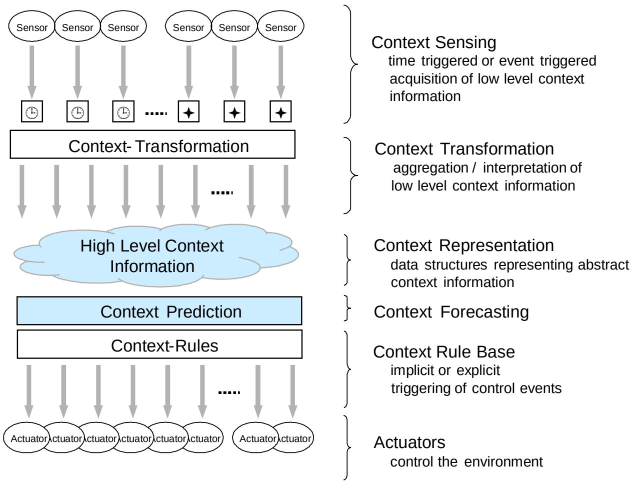
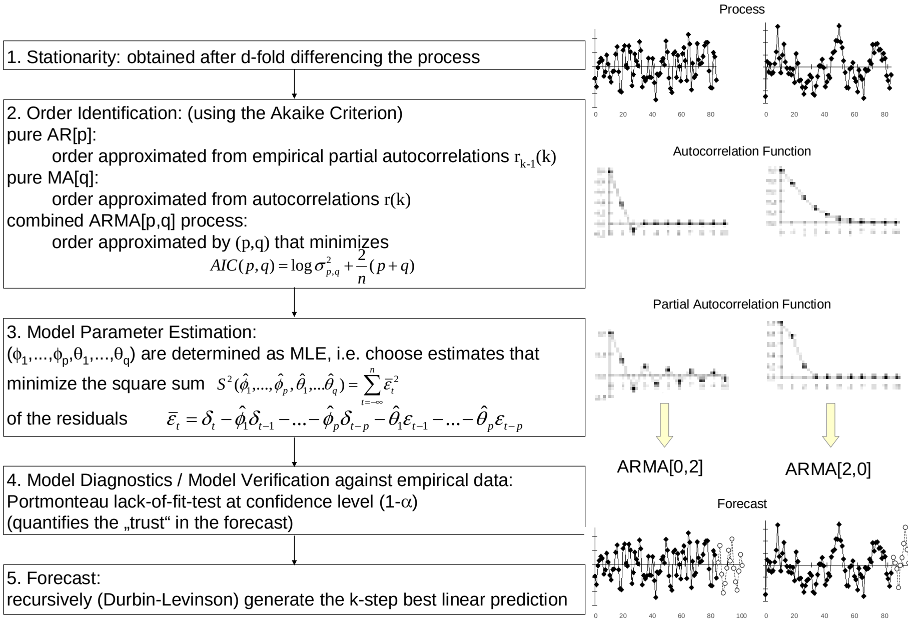


FIGURE 5: FRAMEWORK ARCHITECTURE FOR CONTEXT AWARE SYSTEMS

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) [RDF 99] as RDF statements over instances of the abstract object classes person, thing and place, and their contextual interrelatedness. Context information is also time indexed, allowig 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 [Fers 99]. 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 6 explains the prediction systems and demonstrates the procedure for two independent sensor data processes, together with forecasted data values).



1. Stationarity: obtained after d-fold differencing the process

2. Order Identification: (using the Akaike Criterion)
pure AR[p]:
order approximated from empirical partial autocorrelations $r_{k-1}(k)$
pure MA[q]:
order approximated from autocorrelations $r(k)$
combined ARMA[p,q] process:
order approximated by (p,q) that minimizes

$$AIC(p,q) = \log \sigma^2_{p,q} + \frac{2}{n}(p+q)$$

3. Model Parameter Estimation:
$(\phi_1,...,\phi_p,\theta_1,...,\theta_q)$ are determined as MLE, i.e. choose estimates that minimize the square sum $S^2(\hat{\phi}_1,...,\hat{\phi}_p,\hat{\theta}_1,...\hat{\theta}_q) = \sum_{t=-\infty}^{n} \bar{\varepsilon}_t^2$
of the residuals $\bar{\varepsilon}_t = \delta_t - \hat{\phi}_1\delta_{t-1} - ... - \hat{\phi}_p\delta_{t-p} - \hat{\theta}_1\varepsilon_{t-1} - ... - \hat{\theta}_p\varepsilon_{t-p}$

4. Model Diagnostics / Model Verification against empirical data:
Portmonteau lack-of-fit-test at confidence level (1-α)
(quantifies the „trust“ in the forecast)

5. Forecast:
recursively (Durbin-Levinson) generate the k-step best linear prediction

FIGURE 6: CONTEXT PREDICTION BASED ON SENSOR DATA TIME SERIES ANALYSIS

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 [BCFM 03]. 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.

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 suitecase has been developed. The hardware for the suitecase demonstration prototype uses an embedded single board computer integrated into an off-the-shelf suitcase (see Fig. 7), 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 suitecase 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.

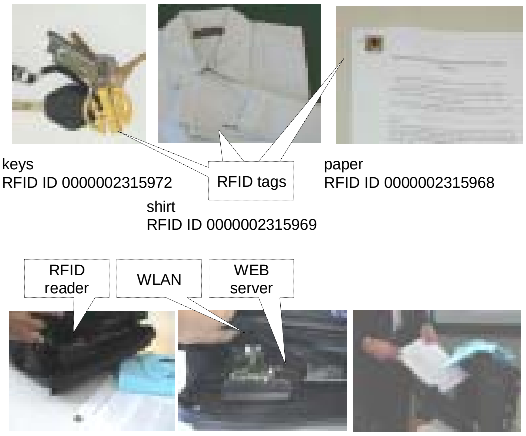


Figure 7: Context-Aware Suitcase hardware and RFID Tagged Objects

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 suitecase 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 suitecase 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 expresses geometrical containment of an object within anotherone, 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 8 gives an example of some of these relations.

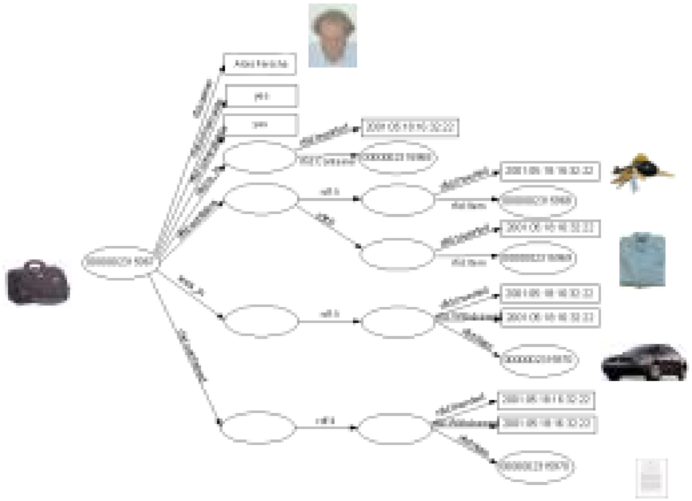


FIGURE 8: RELATIONS AMONG REAL WORLD OBJECTS EXPRESSED AS RDF STATEMENTS

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 e.g. when notifying or reminding the user.

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” [ECPS 02]. According to D. Norman, computing devices well become more specialized in purpose, and will be designed to serve a well defined set of tasks only [Norm 98].

Augured as a fertile source of research challenges [Saty 01], pervasive computing has at least started to broaden the discourse on priciples and methods in distributed computting, mobile computing and embedded systems. Our research group has started to address some of the (primarily) software engineering related issues as outlined in Figure 9, and has brought “Pervasive Computing” to a topical research effort at the Johannes Kepler Universität Linz.

Awareness	**Intelligence**	**Software Engineering**	**Natural Interfaces**	**Appliances**
Identification /Tagging	Pro-Active Software	Coordination	Perception	Mobile Appliances
Authentification	Context Forecasting	Context Frameworks	Implicit I/O	Wearables
Positioning	Machine Learning	P2P Frameworks	Tangible Interfaces	Walls / Surfaces
Tracking	Reasoning	Component Technologies	Augmented Interfaces	SmartThings
Telepresence	Planning	Service Management	Gesture Recognition	Smart Rooms

FIGURE 9: PERVASIVE COMPUTING RESEARCH FOCUS OF OUR GROUP

In our work we encounter the need for a commonly agreed reference model of computation for pervasive computing 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. 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”. Both conceptual models and more advanced methods are needed for “awareness” and “intelligence”. In our current work on “context frameworks” we are challenged to build networked embedded software systems able to

(i) describe, gather, trans-form, interpret and disseminate context information within ad-hoc, highly dynamic and frequently changing computing environments,
(ii) dynamically discover, inspect, compose and aggregate software components in order to identify, control and extend context, as well as to overcome context barriers (like time, position, user preference, etc.),

(iii) allow for dynamic interactions among components in a scalable fashion and satisfying requirements such as QoS, fault-tolerance, reliability, safety and security,
(iv) integrate heterogeneous computing environments and devices with different functionality, ability, form factor, size and limited resources wrt. processing power, memory size, communication, I/O capabilities, etc.,
(v) support the adaptation of novel forms of sensitive, tangible, situative, non-distracting user interfaces not limited to particular modes of interaction or input- and output devices, and
(vi) to allow for the implementation of learning, self-adaptive, plan-oriented, intelligent system behaviour.

Conclusive?

To the computer scientist and pioneer Peter Rechenberg all this must appear more than doubtful – from a computer science prospect, from a research prospect, and even from a social prospect. Vague and ambiguos terminology, non-technical wording and non-formal problem statements seem to indicate a “Leerformel” behind pervasive computing. Where are the models? Where are the concepts? Where is the theory?

To the best of my knowledge, they are not there (yet). But there is hope to find some, at least is there evidence of an increasing demand, and eagerness to have them.

Alan Turing’s paper “*On computable Numbers with an Application to the Entscheidungsproblem*” delivered a foundational result to the computer science community (that 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 ouptut (data) by executing algorithms. Computer science has greately 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”* (from the abstract of my presentation at the University of Linz on December 10, 1999). 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”* [Miln 93] originating from the CCS “concurrency” thought model, another one was Peter Wegner’s *“Interaction Machines”* [Wegn 93] originating from an “interaction” thought model.

It is greatly noteworthy at this point, that Milner's assertion –established models of computation are insufficient– articulated at his Turing Award lecture, was already stated by Peter Rechenberg before:

> *"The concept of algorithm should not remain the basic concept of computer science any longer … it lacks any notion of communication with an environment, excluding input and output while the algorithm is running. This definiton is not suited as a basis for computer science. The concept of algorithm should be replaced with another concept that is based on communicating objects in perpetual processes containing algorithms as a special case."* [Rech 90]

He claims for an interactive Turing machine, as simple and as generic as the original Turing machine in concept, but with the ability to communicate – with other such machines and with the environment. He is well decided to pinpoint this issue:

> *"Ich empfinde es als einen Skandal, daß fast allen Informatikern dieses Problem nach fünfzig Jahren Informatik noch gar nicht zu Bewusstsein gekommen zu sein scheint."* [Rech 98]

His approach to extend the classical Turing paradigm reads: *"Meine Idee ist es, von der Kommunikation zwischen realen Computern auszugehen und zu fragen, was das technische Urelement ist, das die Kommunikation eines einzelnen Computers mit der Außenwelt erlaubt. Ich komme dabei auf ... "* [Rech 98] – which is an indication of his unbreakable ambition in search for *„a complete model for the services of today's* (pervasive ?) *computing systems*" [WeGo 03].

References

[Bana 00] G. BANAVAR ET AL.: Challenges: An Application Model for Pervasive Computing, *Proceedings 6th Annual Intl. Conference on Mobile Computing and Networking (MobiCom 2000),* 2000.

[BCFM 03] W. BEER, V. CHRISTIAN, A. FERSCHA, L. MEHRMANN: Modeling Context-Aware Behavior by Interpreted ECA Rules, *Proceedings of the International Conference on Parallel and Distributed Computing (Euro-Par 2003)*, Springer Verlag, LNCS, 2003, to appear.

[Brow 98] 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

[CaDe 00] D. CASWELL, P. DEBATY: Creating a Web Representation for Places. *Proceedings of the Ninth International World Wide Web Conference,* 2000. http://cooltown.hp.com/papers/placeman/PlacesHUC2000.pdf

[ChLa 99] B. CHRISTENSSON, O. LARSSON: Universal Plug and Play Connects Smart Devices. *WinHEC 9*9, 1999. http://www.axis.com/products/documentation/UPnP.doc

[Cian 97] P. CIANCARINI: Lectures on Coordination Models and Languages, 1997. http://www.cs.unibo.it/~cianca/wwwpages/lipari.html

[CCB 00] J. L. CROWLEY, J. COUTAZ, F. BERARD: Perceptual user interfaces: Things That See. *Communications of the AC*M, Vol. 43, Nr. 3, pp. 54-64, 2000.

[Dey 01] A. K. DEY: Understanding and Using Context. Personal and Ubiquitous Computing, *Special Issue on Situated Interaction and Ubiquitous Computing,* 5(1), 2001.

[ELMM 99] K. F. EUSTICE, T. J. LEHMAN, A. MORALES, M. C. MUNSON, S. EDLUND, M. GUILLEN: A universal information appliance. *IBM Systems Journa*l, Vol. 38, Nr. 4, 1999. http://www.research.ibm.com/journal/sj/384/eustice.html

[Esle 99] M. ESLER ET.AL.: Next Century Challenges: Data Centric Networking for Invisible Computing. *Proceedings of the* 5^{th} *Annual International Conference on Mobile Computing and Networking (MobiCom'99),* 1999.

[Essa 99] I. ESSA: Ubiquitous Sensing for Smart and Aware Environments: Technologies toward the building of an Aware House. *DARPA/NSF/NIST Workshop on Smart Environment*s, 1999. http://www.cc.gatech.edu/gvu/perception/pubs/se99/pp.pdf

[ECPS 02] 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.

[FCE] Future Computing Environments: The Context Toolkit. http://www.cc.gatech.edu/fce/contexttoolkit/

[FBN 01] 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.

[Fers 99] A. FERSCHA: Adaptive Time Warp Simulation of Timed Petri Nets, *IEEE Transactions on Software Engineering,* Vol. 25, No. 2, pp. 237-257, 1999.

[Fers 02] A. FERSCHA: Contextware: Bridging Virtual and Physical Worlds. Reliable Software Technologies, AE 2002. Springer Verlag, LNCS 2361, pp 51-64, 2002.

[Gong 02] L. GONG: Peer-to-Peer Networks in Action. *IEEE Internet Computing,* Vol. 6, No. 1, pp. 36-39, 2002.

[GOI 98] Gorbet, M.G., Orth, M., Ishii, M.: Triangles: Tangible Interface for Manipulation and Exploration of Digital Information Topography. *CHI 1998*, pp. 49-56, 1998.

[HP] Hewlett Packard: CoolTown Appliance Computing: White Papers. http://cooltown.hp.com/papers.htm

[KBMB 00] 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,* 2000. http://cooltown.hp.com/dev/wpapers/WebPres/WebPresence.asp

[KHCR 00] 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.

[LGDE 00] R. LEA, S. GIBBS, A. DARA-ABRAMS, E. EYTCHISON: Networking Home Entertainment Devices with HAVi", *Computer*, Vol. 33, No. 9, 2000.

[Mill 99] B. MILLER: Mapping Salutation Architecture APIs to Bluetooth Service Discovery Layer. Bluetooth Consortium 1.C.118/1.0, 1999.

[Miln 93] R. MILNER: Elements of Interaction: Turing Award Lecture, *CACM.* Vol. 36, No. 1, pp.78-89, 1993.

[MIT] MIT Media Lab: Smart Rooms. http://ali.www.media.mit.edu/vismod/demos/smartroom

[Moze 99] M. C. MOZER: An intelligent environment must be adaptive. *IEEE Intelligent Systems and Their Application*s, Vol. 14, No. 2, pp. 11-13, 1999. http://www.cs.colorado.edu/~mozer/papers/ieee.html

[Norm 98] 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.

[OZKT 01] A. OMICINI, F. ZAMBONELLI, M. KLUSCH, R. TOLKSDORF (EDS.): Coordination of Internet Agents. Models, Technologies, and Applications. Springer-Verlag, Berlin, 2001.

[PRI 02] B. Piper, C. Ratti, H. Ishii: Illuminating Clay: A 3-D Tangible Interface for Landscape Analysis. *CHI 2002*, pp. 355-362, 2002.

[PRM 99] 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.

[RDF 99] 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/.

[Rech 90] P. Rechenberg: Programming Languages as Thought Models. *Structured Programming*, Vol. 11, pp. 105-115, 1990.

[Rech 98] P. Rechenberg: Leserbrief zu "Bemerkungen zu Peter Wegners Ausführungen über Interaktion und Berechenbarkeit. *Informatik-Spektrum*, Vol. 21, No. 6, p. 379, 1998.

[RoCa 00] M. Roman, R. H. Campbell: GAIA: Enabling Active Spaces. *9th ACM SIGOPS European Workshop*. 2000.
http://choices.cs.uiuc.edu/2k/papers/sigopseu2000.pdf

[SDOA 99] 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. ftp://ftp.gvu.gatech.edu/pub/gvu/tr/1999/99-29.pdf

[Saty 01] M. Satyanarayanan: Pervasive Computing: Vision and Challenges. *IEEE Personal Communications*, pp. 10 – 17, 2001.

[Schi 95] W. N. Schilit: *A System Architecture for Context-Aware Mobile Computing (Ph. D. Dissertation).* New York: Columbia University, 1995.

[SATT 99] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. Laerhoven, W. V. Velde: Advanced Interaction in Context. *HandHeld and Ubiquitous Computing*, 1999.

[ScLa 01] A. Schmidt, K. Laerhoven: How to Build Smart Appliances?, *IEEE Personal Communications*, Vol. 8, No. 4, pp. 66-71, 2001.

[Wald 99] J. Waldo: Jini Technology Architectural Overview, White Paper, Sun Microsystems, Inc., January 1999.

[WDKF 02] 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.

[WeGo 03] P. Wegner: Why Interaction is more powerful than algorithms. *CACM*, Vol. 40, No. 5, pp. 80-91, 1997.

[Wegn 93] P. Wegner, D. Goldin: Computation Beyond Turing Machines. Technical Opinion. *CACM*, Vol. 46, No. 4, pp. 100-102, 2003.

[WMG 93] P. Wellner, W. Mackay, R. Gold: Computer Augmented Environments: Back to the Real World. *CACM*, Vol. 36, No. 7, 1993.

[Weis 91] M. Weiser: The Computer of the Twenty-First Century. *Scientific American*, pp. 94-100, 1991.