Lecture Notes in Telecooperation 3: Ubiquitous Computing

Chapter 0: Introduction
Lecturer: Prof. Dr. Max Mühlhäuser
Contributors: Dr. Erwin Aitenbichler, Dr. Dirk Schnelle-Walka
Dr. Sebastian Ries, Dr. Melanie Hartmann

Copyrighted material – for TUD student use only
Basic Information

- Lectures Monday 14:25–16:05 (14.30–16.00) S1 01/C120
- Exercises follow right afterwards
  - type: programming assignments + project
  - rather self-sustained, selected in-depth topics
  - start: 18.04.2011 (next week) today: “Das intelligente, senioren-angepasste Haus” (C110)
- Appointments: after lectures & via eMail
- Grade V2+Ü2: exam + bonus from exercises
- Exam: 18.07.2011 10:00 – 12:00 C205
Primary Reference, available as PDFs (Details see Web Site):

Additional Primary References:
4. J. Krumm (Ed.): Ubiquitous Computing Fundamentals, CRC Press 2010
5. D. Cook, S. Das (Ed.): Smart Environments, Wiley 2005

SECONDARY Books:
3. B. Walke: Mobilfunknetze und ihre Protokolle 1, 2, Teubner Stgt. 2000
4. K. Kelly: Out of Control, Perseus Books, Reading, MA; USA, 1994
7. J. L. Encarnaco: True Visions. The Emergence of Ambient Intelligence, Springer 2006
Contents of TK3

0. Introduction to Ubiquitous Computing

1. Connectivity
   includes *mobile computing* as an important prerequisite:
   mobile communications/networks/Internet

2. Scalability

3. Adaptivity (Context & Location)

4. Ease-of-use

5. Liability

Slides contributed by lecturers and TK researchers;

Some illustrative slides from external sources
⇒ for internal use & TK3 students only
Outline - What this chapter covers

- A first introduction: definition, importance
- UC challenges – a first approach: humane & integrated UC
- The S.C.A.L.E. classification of Ubiquitous Computing
- A brief history of Ubiquitous Computing (UC): Mark Weiser et al.
- From buzzword Babylon towards a taxonomy
- A Reference Architecture for Ubiquitous Computing (optional)
Ubiquitous Computing describes the 3rd “era” of computing

- **Era 1: Mainframe Computing MC** - many users use one computer
- **Era 2: Personal Computing PC** - one user uses one computer
- **Era 3: Ubiquitous Computing UC** - one user uses many computers

**Note, for Era 3:**
- some computers are **worn** (mobile phone, PDA; body sensors, life recorder...)
- other computers are **encountered** on-the-move, more or less consciously (cf. info kiosk / data projection versus badge-reader / surveillance cam)

**UC means “networked computers everywhere”**
- μC’s were embedded in VCRs, washing machines etc. since decades already!
- **BUT:** now, with UC, the embedded computers become
  - networked, i.e. part of a “web” of interacting computers surrounding us
  - “programmable”: changing firmware / SW is easier, less noticeable to user (good: cf. bug fixes; bad: cf. undesired functionality, e.g. spying)
1.1 Definition: What is UC?

- UC connects general & special purpose computers
  - the distinction is more a spectrum than two distinct classes
  - the power / resource spectrum (CPU, memory, bandwidth, ...) is huge!

How to ‘define’ Ubiquitous Computing?

A simple definition is given above: UC is the third era of computing according to the aforementioned categorization

Our more detailed and self-sustained definition:

Ubiquitous Computing is the dawning new era of computing\(^1\), in which individuals are surrounded by many networked, spontaneously yet tightly cooperating computers, (i) some of them worn or carried, (ii) some of them encountered on the move, (iii) many of them serving dedicated purposes as part of physical objects, (iv) all of them used in an intuitive, hardly noticeable way with limited attention”.

\(^1\) following the ‘mainframe’ and ‘personal computing’ eras
1.1 Importance: Sample Nodes in UC networks

Tiny, rather general purpose computers acting, e.g., as nodes in sensor networks (cf. UC Berkeley’s motes, Particle Computer GmbH’s particles)

Wearables and mobile computing devices. cf. handhelds for warehouse picking, washable computer jackets, or companions like the “lovegetty” pictured: stores user’s profile, beeps when a “compatible” person – with lovegetty – appears

Networked appliances, e.g., “smart vending machines” that transmit fill status, errors etc. to head offices; press rumors about adaptive pricing caused protests

Smart labels such as RFIDs (above) and active badges (below) identify (option: locate, characterize) objects, humans, animals; benefits and privacy threats widely discussed in press
issue #1: on the networking side, the exponential growth (# of nodes connected in the Internet) continues way beyond the world’s population consequence: scalability becomes even more crucial
2.3 Importance: exploding # of nodes, consequence

issue #2: different doubling rates for networks, CPUs (Moore’s law) etc.; impact on: system architectures, algorithms, etc. (e.g., RPC becomes cheaper, mobile computing maybe more expensive?)

- CPU (Moore’s Law)
- Internet (# of nodes)
- graphic cards (#polygons/sec)
- wireless-net bandwidth (bps)
- battery capacity
- distributed computing:
  - disconnected-op?
  - mobile code?
  - server architectures
At the 30,000 ft. level, UC is “the next key technology to change our society” for 3 reasons:

1. **UC describes *the* next era of computing**
   - therefore, all of computer science / IT is potentially impacted
   - ... it will be difficult to be selective in designing this course/book (but we will see that a course / book on UC still makes a lot of sense)

2. **UC has potential impact on every facet of our lives**
   - computing is no more “what we do when we sit at the computer” + “what is encapsulated/hidden deep inside VCRs etc.”
   - see the more / less comparison on next slide

3. **UC is impossible and inevitable at the same time**
   - components (“gadgets”) are developed and deployed, cf. slide “UC sample nodes” above
   - more and more UC scenarios become profitable → industry will push
   - many issues of “the whole” i.e. integrated UC systems still unresolved
2.1 From Importance to Challenges

A first glance (not yet structured, non exclusive) at challenges...

Anytime/anywhere presence of networked computers means:

- more sensitivity ↔ less protection (cf. security)
- more dependence ↔ less perfection (cf. reliability)
- more obtrusion ↔ less attention (cf. HCI)
- more garrulity ↔ less throughput (cf. networking)

In other words: as computers become ubiquitous ...

- there is a risk that “the whole may be way less than the sum of the parts” i.e. the desired integral functionality may lack way behind
- some known problems aggravate considerably – we’ve got to care!
- in short, UC is a problem of complex integrated systems, not of gadgets compare to: new Airbus, Space Shuttle, not better rocket-fuel, headlights
2.1 Humane & Integrated UC - Overview

Top Level Challenges:
A. **Systems Integration** (Strength of Europe)
B. **Humane Computing** (Chance for Europe)

Outlook (see below):
- Scalability
- Connectivity
- Adaptability
- Liability (=Security ++)
- Ease-of-Use

features of UC “gadgets” are not the problem ...

... but making them „humane“ and integrating them into a useful whole
Mainstream UC Research emphasizes selected issues, such as ...

- **(Integrating, Using) tiny devices (sensors, ...)**

- **Context-Aware Computing**
  - ill defined; basically, the quest for adaptive software that ...
    - understands the „situation in which the user currently works“
    - adapts to that situation by
      - **reducing** complexity (e.g., adjusting UI to current needs & options)
      - **automating** input (e.g., retrieving data from „environment“, not user)
      - **optimizing** functionality

- **Multiple Modalities**
  - again, ill defined; modality is a „way of doing I/O“, often an
    - input and/or output „channel“ (what is that?) like graphics, voice, keyboard/mouse, gesture (note: examples from different categories!)
    - **UC** applications on appliances (cell phone, PDA... today, more tomorrow)
    - **→** Multiple modalities to be supported
      - truly multimodal UI: flexible combination of „channels“, user’s choice

- **We will put these two issues in a larger context**
2.3 Challenges: UC Integration

Pertinent Application: Real-Time Enterprise.
full coverage: production/supply chain, retail, post-sales

Global Internet: seamless context-aware (service!) network
3.1: S.C.A.L.E. Overview: Breadth vs. Depth

- **Problem:** UC touches almost every area of Computer Science, and more!
  - readers need background for understanding why what will be emphasized later
  - a categorization of issues (taxonomy) is needed for organizing the topics and for getting oriented → cf. the S.C.A.L.E. taxonomy

- **So, is it a research/teaching subject in its own right?**
  - **yes,** compare to Distributed Systems:
    - there is distributed simulation, programming, algorithms, databases, AI, ...
    - yet, there are courses on Distributed Systems
  - **but:** how to master breadth vs. depth?
    - be broad first, organize “problem space” → S.C.A.L.E., see below
    - then emphasize pertinent problems (new in UbiComp or much more relevant)
    - standard computer scientist should hardly need add’l background
    - ... but will sometimes want to dig deeper in some subtopic ... which is normal

- **But then, is course/book going to be “complete”?**
  - no, but more complete than any other book/course before
  - ... and more well organized, due to S.C.A.L.E. scheme, see below
Major UC challenges, on a very high level:

- **S – SCALABILITY**
  - how to support cooperation of “zillions” of components?
  - how to support nomadic users around the globe?

- **C – CONNECTIVITY**
  - how to “easily” connect these zillions?
  - wireless networks – a blessing and a curse (unreliable!); important but...
  - most issues above wired/wireless net (note: overlaps scalability): how to find/understand peers, enable zero configuration, design huge networks w/o server/bottleneck

- **A – ADAPTABILITY**
  - usage during daily work, surrounded by 100s of components: need minimal interaction
  - major approach: context-aware computing – use it to automate tasks & reduce options
  - adapting-to-user (user *modeling*) must be focused beyond context-awareness

- **L – LIABILITY**
  - term indicates: we must go beyond today’s IT security solutions (not goals)
  - today’s solutions do not scale (root PKI?), are not “humane”
  - & don’t flexibly consider conflicting (privacy, traceability) / related goals (dependability...)

- **E – EASE-OF-USE**
  - adaptability permits “minimal” ..., ease-of-use means “optimal” interaction (related!)
  - issue: optimal use & combination of modalities, advancement of specific modalities
  - issue: “understanding” natural input: a) xxx-to-text; b) text-to-meaning; “intelligence”?
3.2 Relation to Computer Science Research

S.C.A.L.E.

scalability
connectivity
adaptability
liability
ease-of-use

Bionics
Socionics
Dependability

Distributed Systems
Context aware computing
User modelling

Computer Networks

Dependability

IT Security
Quality-of-Service, SLA, Accounting

HCI

AI (NLP/KM)

Software Engineering
4.1 History: Mark Weiser

- *1952, †1999 (died before UC really took off)
- Coined the term, spread the vision of UC
  

- Worked at Xerox PARC (now: PARC), the (?) world leading center of research combining computing with humanities (≈ “birthplace” of mouse, windows-based UIs, desktop metaphor, laser printer, many CSCW contributions and much more)

- Co-Developed prototype UC devices, in particular 3:
  - Pad, a prototype PDA
  - Tab, a prototype TabletPC
  - Liveboard, cf. Smartboards™ --- all in the late 80es!!

  he investigated their *integration* in group work (CSCW) scenarios, imagined the ubiquitous availability of Tabs (laying around in meeting rooms, personalized as users grab them), ...
4.1 History: Mark Weiser

- MW’s most frequently quoted statement: “The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.”

- But MW’s vision comprised more, see below

- We will try to recall Mark’s view on three conflicting issues:
  - UC vs. VR (virtual reality)
  - UC vs. AI (artificial intelligence)
  - UC vs. UA (user agents)

- Thereby, we imagine Mark at panel discussions, as a UC advocate trying to argue why VR, AI, and UA are “dead end” research roads while UC is the open road

- Later, we will take a less dramatic standpoint
VR is based on (3D and semantic) models of the real world
- (ever larger, ever more detailed) cut-out of the world is modeled in the computer
- put to an extreme: the world is moved into the computer
- ... and even the user becomes a computer peripheral (hmd, data glove)
- with UC, in contrast, the computer is moved into the world!!

→ not one “big boss” computer, but many small ones with dedicated task & responsibility
→ networking for making sense of the small parts
→ not the computer is in the center, but the human!
around 1980, AI had been over-hyped
around 1990, frustration reigned: AI had not lived up to its promise
Mark’s argument: analogy computer = brain is exaggerated
  terms “intelligent” and “knowledge” raise too many expectations
  the AI vision of intelligence concentrated in a computer is wrong
With UC, in contrast, we aim at *smart* components
  they figure out a tiny cut-out of the world only (just temperature or just presence of object, at just a small location, ...)
  smart computers compare to intelligent computers like neurons to brains
  higher-level “sense” comes from networking smart components
4.1 History: Mark Weiser – UC vs. UA

- In contrast to VR and AI, user agents are not very prominent any more; some basic concepts have remained though.
- UAs were thought as intelligent intermediaries between the user and the computer world, thus an approach towards ease-of-use / HCI.
- Mark challenged five requirements for UA as “dead end roads”:
  1. UAs should give advise --- why don’t they do the job themselves?
  2. UAs should obey (like a butler) --- why aren’t they more proactive?
  3. UAs should work at the interface --- why interact and not do things?
  4. UAs should listen to the user --- with immature natural language processing technology, speech recognition etc., how should they understand?
  5. UAs should learn the user’s preferences, wishes etc. by observing --- with immature machine learning technology, how should they do the right thing?
- Note: wrt. 1-3, UAs are “too little”, wrt. 4+5 they are “too much”.
- UC, in contrast, according to Mark, should aim at “agents” which
  - carry out actions and not just mediate
  - do that largely autonomously such as not to bother the user
  - … and therefore have not much of an interface at all

- the complexity of the made (engineering) reaches the complexity of the born (cf. biology/nature, social organisms)
- we should “learn the principles” of “the born” and adopt
- looks at bee hives, evolution, and many more: how do they cope with errors? with change? with control? ...
- looks at industrial evolution, the quest for perfection
- proposes things like:
  - give away control (autonomous responsible behavior of part in whole)
  - accept errors for selection, adaption, constant optimization
  - truly distribute control (no central instance)
  - promote chunks (hierarchies ...) for taming complexity
  - accept heterogeneity, disequilibrium as sound bases for survival
4.2 History: Two more visionaries (2)


- critique: PCs are complex, try to be all-purpose / all-user
- critique: PCs are isolated from daily work & live therefore, Don is the first one to clearly demand:
  - Information appliances: dedicated for specific task/problem
    - way simpler and more optimized
  - Human-Centered Development: design the appliance such as to optimally support its user
  - design axioms simplicity, versatility, & pleasurable
  - „systems“ to be flexibly composed families of appliances
Mark Weiser was not perfect (like any visionary). Recall his 3 points:

- **VR-vs.-UC (embodied virtuality) dispute** – today, we would say:
  - we need „the computer in the world“ ...
  - but also „the world (model, distributed) in the computers“ if the cooperating whole shall make sense of the „smart“ parts
  - (VR approaches this reconciled view with augmented reality (AR) concepts)

- **AI-vs.-UC dispute**:  
  - AI was indeed overhyped ...
  - but we need to find out how to combine „smart computers“ like neurons into a „brain“ that makes sense
  - note: similar argument as above; conclusion: *integration is key*

- **UA-vs.-UC dispute**:  
  - „listen to user and learn“ was (over?) ambitious ...
  - but (1) autonomous-actions instead of „obey/advice“: even more ambitious
  - but (2) machine learning made progress → „learn->advice“ becomes feasible
4.3 History vs. Present

As to Kevin Kelly

- **autonomic computing** and many „soft computing“ disciplines (neural networks, ant colonies, evolutionary algorithms) follow his principles
- ... but scalability *plus* reliability remain a huge challenge
- again: *integration is key*

As to Don Norman & Mark Weiser

- user-centered design, context-aware computing, multimodal UIs etc. follow their quest for „humans in the center“
- ... but „humane“ computers comprise further issues, in particular UC-ready IT security (cf. F. Stoyano’s book on „Ubiquitous Security“)
- in summary: *humane computing is key*
5.1 Buzzwords → Taxonomy: Name of Discipline

UC has many names (sigh! hinders establishment as wide-spread discipline!)

*Warning*: follows a boring myriad of terms but: needed for ‘UC literacy’

- **Post-PC era**: not a good term, only says what it is *not* (PCs)
  - refers to relation users:computers  \( N:1 \) (1960-80) →  \( 1:1 \) (1980-2000) →  \( 1:N \)

- **Pervasive Computing**: more common in industry
  - (coined by IBM?); emphasizes computers „penetrating“ the world

- **Ubiquitous Computing**: more common in academia
  - emphasizes the „final“ state of penetration: computers everywhere

- **Disappearing Computers**: less common, like...
  - **Calm Computing**: also less common, says basically the same:
    - should be hardly noticed by user, should not „disturb“

- **Invisible Computers**: same as above, but *very* demanding
  - (disappearing / calm is less demanding)
5.1 Buzzwords→Taxonomy: Name of Discipline

... continued:

- **Ambient Intelligence**: „invented“ for EU research framework programs (5, 6, 7); „ambient“ refers to Weiser’s quote, „disappearing in the environment“; „intelligent“ is revival of over-hyped term → Amb.I. is *only* common in Europe

- **Mixed-Mode Systems**: not very common either, but pops up every now and then refers to „resource heterogeneity“ in the range RFID – Sensor – PDA – PC – Cluster etc.

- **Tangible Bits**: rather uncommon (1 book?), seen in NL/JP means: computers in appliances, Smart Paper (see below) etc. → „bits“ i.e. Computing become part of physical world

- **Realtime Enterprise**: common in business world, see next slides

- some people argue that the major terms represent subsequent steps, e.g.:
  - pervasive → ubiquitous → disappearing → invisible → ambient ...

  but this should not be tried:
  - the terms are synonymous
  - „proof“: research labs with either of these names work on the same issues!
5.1 Buzzwords ➔ Taxonomy: Name of Discipline

„Hypothesis“:
Ubiquitous Computing = Pervasive Computing = Ambient Intelligence

Google-Hits: 1100 k 800 k 880 k (circa)
5.1 Buzzwords ➔ Taxonomy: RTE

Real time enterprise (RTE): used in enterprise software context
RTE reduces the “human intervention gap” (see next slide) between
- model of the world in the computer (note difference to MarkWeiser-View!!) and
- real world itself, by bringing computer to the world (today mainly: via RFID tags)
- RTE has on line:
  ✓ spaces (environment): via sensors, actuators
  & items: via tags (cf. RFID) ➔ embedded (Internet) appliances
  ✔ humans: mostly hands-free / eyes-free
5.1 Buzzwords ➔ Taxonomy: RTE

- Manual data entry
- Speech recognition
- Bare code scanning
- Passive tags
- Active tags

Gap between physical and digital world
- Cost of data entry

Virtual world
- Inter and cross-company information systems (e.g., ERP systems)
- Local, regional and global communication networks (e.g., Internet)

Real world
- Human beings
- Products
- Production means

Human intervention required | No human intervention required
Towards taxonomy of UC components (reconsider UC definition): 2 classes

1. those “attached to” human (“...some of them worn...”) – more precisely:
   - **carried:** some major terms (“categories) are
     - mobile device (‘full computer’), smart badge (identity & rights)
   - **worn:** body sensor etc.; range [“smart cloth” ... Laptop-in-Backpack] is blurred!
   - **implanted:** hype about RFID implants; eHealth implants → communication?

2. those encountered
   - **entities:** called, e.g., smart items, smart objects, or smart products
     - terms not settled, possible difference: composition level
       - item: atomic, part of functionality often externalized (e.g., “data shadow”)
     - usually “tangible” i.e. attached to physical object
     - we will stick to wide spread term **smart item**
   - **environments:** called, e.g., federated smart products/environments/spaces
     - again - terms not settled, possible difference: composition level
     - usually “intangible”, “hidden” (smart environment middleware, platform...)
     - may actually be remote (e.g., server), but effect is experienced
5.2 Buzzwords → Taxonomy: Smart Items

- Pragmatic classification of Smart Items (“increasingly smart”):
  1. **Smart Tag** (syn.: Smart Label):= \{ID, communication, add-ons\}
     - brings identity & (passive) communication to physical entities (cf. 1.3.3)
     - active comm., processing, memory: all optional (cf. 1.3.3)
     - further classification 1: communication technology (IR, RF, Ultrasound, …)
     - further classification 2: active/passive, with/out memory / CPU
     - famous representative: RFID tags (active and passive)
  2. **Internet** (or: networked) **Appliance**:= \{embedded system, communictn.\}
     - often: exists since decades, now communication added
     - but! effect equivalent to “protozoa → higher organisms” in biology??
  3. **Sensor network**:= cooperating specialized devices
     - not an appliance: no individual use
     - often based on “tiny computer” plattform that allows sensors to be attached (UC Berkeley “Motes”, Particle Computer GmbH’s “particles”…)
     - research focus on cooperation / networking
5.2 Buzzwords ➔ Taxonomy: An Attempt

We may start to organize UC components in a real taxonomy:
5.2 Need-to-Know: Further Buzzwords

Above taxonomy is helpful, will be used in the remainder ...

- ... but: not settled → other buzzwords cannot be fully classified yet
- like, above: federated ..., smart objects/products/spaces
  (also: see our own definition of smart products, later)

Further UC components to be mentioned, but not fully classified yet:

- **Smart Dust:** alludes to zillions of *very small* sensors
  - vision 1: auto-decay (organic? → compost 😊)
  - vision 2: edible (health examination etc.) → inhalable?
    - variant: picked up “afterwards”, data read out → not truly a network
  - feasible today (example): aircraft sheds sensors over contaminated area, sensors cooperate → deliver environment data

- **Things-that-think** (project, N. Negroponte, MIT media lab)
  slogan: “in the past, shoes could *stink*. In the present, shoes can *blink*. In the future, shoes will *think*”

- **Smart Paper** (new category: smart materials?)
  - originally: re-usable carrier for “daily news” etc.
  - today, term sometimes misused by press/marketing

- **Smart / Intelligent / Ambient <YouNameIt>:**
  - terms discovered by marketing
  - “product will sell better” 😊
### 5.3 Need-to-Know: Internet-of-Things

- **The Internet of things**: term favoured by press
  - emphasizes 50% of ubiquitous computing (cf. taxonomy): “encountered”
  - discussion dominated by emerging smart tags standards, in particular AutoID
  - ... and by Internet appliance standards, in particular OSGi

- **AutoID**: center at M.I.T. standardizing RFID-based successor of “Barcode”
  - first, a look at RFID chips:
    - actual chip may be only 4 mm\(^2\)
    - „giant“ antenna
    - on chip: ID-no. „burnt into ROM“
    - may contain RAM
    - ... and active communication (, even CPU)
  - for AUTO-ID: ID is 96-Bit electronic product code EPC
  - EPC succeeds barcode, has serial-no!
  - compare to “class ID” vs. “object ID”
  - for mass markets (Walmart etc.), currently used mainly for “palette & case level tagging”, not for “item level tagging” yet

- **Generation 2 (2006): improvements**
  - smaller, cheaper
  - reader reads hundreds of tags “simultaneously”? 
  - “printable”: paper labels with embedded RFID?
5.3 Need-to-Know: AutoID / EPC

Electronic Product Code EPC: “EAN/UPC + serial no.”

Header: 8 Bit  Manufacturer: 28  Product: 24  Serial Number: 36

“Service Discovery”, e.g., ONS Server (cf. Internet-DNS)

Server: XML-Data in Product Markup Language PML

step 1: read ID# → ask, e.g., ONS server for URL

step 2: use URL to access product data “somewhere”

three „tiny changes“:
1. serial no.
2. reading efficiency (d, %, Σ)
3. distributed system = Internet

„big effect“!!!
... leads to new understanding of distributed systems

- for > 40 years, distributed systems were defined as:
  \[ DS = \{AS\} \cup CSS \] – a set of autonomous systems interconnected through a communications subsystem CSS
  
- AS: nodes, i.e. processor-memory-pairs; more precisely: 4 elements – CPU – memory – comm(unication capabilities) – identity (e.g., IP adr.)
  
- client-server world: 2 roles of ASes were distinguished (client/server), peer2peer world: “all nodes are created equal”

**UC world:** this definition must be revised!

- a node’s “memory” may reside elsewhere, e.g., on a WebServer as “data shadow” of an object
  
- considering (e.g., RFID) tags, a node’s elements are:
    
    - identity \textit{required}  
    - communication \textit{required (active? passive?)}  
    - memory \textit{optional}  
    - CPU \textit{optional}  

![Diagram](image_url)
5.3 Need-to-Know: OSGi

OSGi is the second “need to know” standard
(of course there are myriads more, but less dominant,
cf. further chapters)

OSGi enables deployment / revision of code ("services") over the net
- initially for: SetTopBox, Car, “ServiceGateway”, consumer electronics, ...
  - e.g., car’s “smartness” software upgraded at red light
- may be basis for “smart environment downloading code to appliance”

OSGi is based on Java!
- defines VM, downloadable-code (program) format
- standardizes part of program interface, “discovery” of other programs(!)
- ... relies on other “service discovery” standard (UPPI or else, see later)
Smart Home := Smart Environment Category: Your Home

- many projects worldwide
- some prestigious projects
  - industry: Microsoft eHome, Philips AmbientIntelligence, ...
  - academia: GeorgiaTech AwareHome, MIT House_n, ...
  - platforms: HP Cooltown

- but many projects “terminated”: business case? user acceptance?
  - current hope: business case assisted living (cf. AgingSocieties like Europe9
    - e.g., keeping elderly selfsustained for 1 more year in life:
      safes ~.5 bn € in Germany alone?
    - in this area, projects with large user studies: Zwijndrecht (B), Tønsberg (N), ...
  - other hopes: home security, energy conservation, home entertainment

- BUT – Bottomline:
  - high likelyhood that large scale deployment is in business, not home!
    - e.g., logistics, inspection, manufacturing, services ...
    - why? business cases more obvious: companies invest! (heavily)
5.3 Need-to-Know: More on Smart Items

- Introduced by UCB, UCLA; now in Europe (SmartIt’s etc.), ...
- Small, resource limited devices
  - CPU, disk, power, bandwidth, etc.
  - with simple scalar sensors – temperature, motion
- some: customized to single domain / task (ecology ... health ... military)
- ad-hoc wireless network (ZigBee or private, wLAN? power hungry!)
- examples below (UCB “motes”), but note: getting smaller!!
- ingredients, e.g., Atmel CPU, TinyOS, TinyDB, power conservation ...
## 5.3 Need-to-Know: Berkeley Motes

*cf. (Levis & Culler, ASPLOS 02)*

<table>
<thead>
<tr>
<th>Mote Type</th>
<th>WeC</th>
<th>rene2</th>
<th>rene2</th>
<th>dot</th>
<th>mica</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>9/99</td>
<td>10/00</td>
<td>6/01</td>
<td>8/01</td>
<td>2/02</td>
</tr>
</tbody>
</table>

**Microcontroller**

<table>
<thead>
<tr>
<th>Type</th>
<th>AT90LS8535</th>
<th>ATMega163</th>
<th>ATMega103</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prog. mem. (KB)</td>
<td>8</td>
<td>16</td>
<td>128</td>
</tr>
<tr>
<td>RAM (KB)</td>
<td>0.5</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

**Nonvolatile storage**

<table>
<thead>
<tr>
<th>Chip</th>
<th>24LC256</th>
<th>AT45DB041B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection type</td>
<td>I2C</td>
<td>SPI</td>
</tr>
<tr>
<td>Size (KB)</td>
<td>32</td>
<td>512</td>
</tr>
</tbody>
</table>

**Default Power source**

<table>
<thead>
<tr>
<th>Type</th>
<th>Li</th>
<th>Alk</th>
<th>Li</th>
<th>Alk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>CR2450</td>
<td>2xAA</td>
<td>CR2032</td>
<td>2xAA</td>
</tr>
<tr>
<td>Capacity (mAh)</td>
<td>575</td>
<td>2850</td>
<td>225</td>
<td>2850</td>
</tr>
</tbody>
</table>

**Communication**

<table>
<thead>
<tr>
<th>Radio</th>
<th>RFM TR1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rate (Kbps)</td>
<td>10</td>
</tr>
<tr>
<td>Modulation type</td>
<td>OOK</td>
</tr>
</tbody>
</table>

Meanwhile: a variety of manufacturers following Motes/TinyOS, Alternative ‘platforms’ like SunSparks & others
6.1 Reference Architecture: Motivation

Why a reference architecture?

- “Old definition” of Distributed Systems not appropriate
  - OLD: “a distributed system is a collection of autonomous systems AS, interconnected through a communication subsystem CSS”: DS := \{AS\} \cup CSS ...
  - ... where communication happens via message exchange only
    - (note 1: an AS may be a multi processor with shared memory i.e. shared mem not entirely “forbidden”)
    - note 2: AS defined to consist of: 1. processor, 2. memory, 3. communication, 4. identity (e.g., IP address)
  - NEW: nothing but (half) “3” required: passive communication; rest optional!

- Further “old” views:
  - client-server world: two kinds of nodes exist (clients, servers), maybe blurred
  - peer2peer world: deliberately no distinction between nodes
  - for both views: all nodes are created equal

- However, in UC world, there are 2 main reasons to introduce distinctions:
  - resource heterogeneity: special-purpose inappropriate for general-purpose tasks
  - role heterogeneity: very personal nodes should not be treated like very public ones?
6.1 Reference Architecture: Motivation

- What kind of architecture? In general, two options:

  1. **Layer architecture** – describes HW/SW layers, “abstract machines”
     - assumption: useful for special purposes, but declining general use:
       - OSI and TCP/IP layering increasingly violated, considered inefficient
         replaced today by approaches for customized layering / function selection
       - rather, layering used today to describe the role & “place” of a specific
         abstraction layer / API
  2. **Component architecture** – describes interacting components
     - towards global interaction of UC components in a meaningful “whole”,
       a component architecture is very promising
     - OSI “ODP” model had limited success: overly complicated, not yet urgent
     - in the remainder, suggestions are made
       - in particular, the MUNDO reference architecture
     - note: no common agreement exists yet
     - the MUNDO reference architecture will be used for
6.1 Reference Architecture: 1st Approach

UC systems: software is only one kind of participant

Other kinds:
- humans (undervalued in the past → bad usability)
- real world (“tangible”) objects (with embedded SW?)
- huge (exploding) amounts of information (cf. Google success)
- as to software, we need yet a better component based approach

Leads to four kinds of UC components:
- **humans** – most of them *not* using a desktop PC
- **smart services** - distributed context-aware software components
- **smart items** of all kinds - wearable computers, appliances, “spaces”, etc.
- **smart content** - knowledge media based on multimedia documents & streams

A UC system, then, is a collection of such components:
Mundo departs from the following assumption:
- UC makes everyday life “computer assisted”
- many actions will be machine2machine, many of them representing user’s will / preference / general-permission AND: these actions will have legal impacts (cost, liability, ...)
- acceptance is only feasible with a “digital persona” in which the user has full trust
- digital persona: a digital representative of the users, acting on her behalf
- we believe in the need for a tangible persona which the user can always carry we call this item:

\[
\text{ME} = \text{Minimal Entity}
\]
Requirements for a ME:
- full trust by user: device must be understandable, “under control”
  - e.g., updates change under user control, i.e. infrequent (trust must grow!)
    - exception, maybe: alien software operated in a sandbox mode
  - actions delegated to device must be intuitive, understandable
- in order not to rely on other devices (such as a USB stick relying on a Laptop for PIN entry!), the ME must possess:
- for reasonable minimal functionality, it is useful to add
  5. context-sensitivity (location, head orientation found useful!)
- as a fully trusted party, it must be able to act as a
  6. Persona Support capable of providing Ubiquitous Security; this functionality should be bootstrapped via highly secure credentials (e.g., fingerprint)

As to the User I/O:
- the device is to be carried at all times (since it accompanies everyday actions)
- interaction will often happen in limited-attention, hands-/eyes-busy mode
- therefore, voice interaction appears to be a natural choice
  - complemented by touch/gesture in silent & very noisy environment
  - complemented by the full range of I/O via device association, see below
Other devices to be operated on the user’s behalf are granted functionality & controlled through the ME: they are associated with the user.

This leads to a second class of components:

**US = User aSsociable**

Secure protocols for temporarily associating particular devices have been developed;

For other issues, only partial answers exist, in particular:
- how to prove /enforce the desired properties / non-properties of devices?
- (display: shall temporarily show my data, but not store/send it, etc.)

ME and US form a trusted PicoNet named:

**MINE = Mundo Integrated Network Environment**
As several users with MEs and USes meet, they may be supported in forming a

**WE** = Wireless group Environment

Association of devices may change (e.g., think of one user lending her DigiCam to another user: during re-association, the old user’s photos may be hidden, the temporary user’s photos may be send to the appropriate storage, ...)

More on “WE” type configurations: see chapter on Opportunistic Networking
Devices encountered on the move may be associable (→ turn into USes), but they may also lack full association capabilities. Rather, they may deliberately be designed for cooperative or shared use. Such components are called **IT = smart Item**.

For the user, the US – IT distinction is crucial in terms of liability!
Finally (and maybe most important), functionality will be provided through “intangible” services in the network; since such services will be provided in overlay networks of all kinds (see corresponding chapters), the encompassing set of such services is denoted as

**THEY** = Telecooperative Hierarchical ovErlaY networks

In terms of the “first attempt” before, such services provide access to software based functionality *and* to smart content.
The final “Mundo Architecture” may serve as a reference
- for checking the functionality of UC approaches,
  Smart Environment platforms, etc., for necessities & completeness
- for discussing required functionality of its elements (cf. the discussion about “ME” functionality above
- for furthering the architecture as experience with UC systems grows
- for anchoring the UC issues and topics, e.g., as covered in the following
6.3 Other Reference Architectures

3 architectures (out of many) worth mentioning:

- **OSI ODP (Open Distributed Processing):** mix of layered/component
  - rather old, little success
  - tried to tame the complexity of large distributed SW systems
  - introduced views for different stakeholders (developer, user, ...)
  - Follow-on: OSDI (open distr. services infrastr., rather component than layered), influenced some of the UbiComp-Architectures built recently

- **“I-Centric Services” by Fraunhofer Fokus Berlin** (⇒ OMG, ...)
  - shares with Mundo: concept of user-centric (“ME”) approach
  - yet, based on concept of “the” universal component interface: called Super Distributed Object (SDO): has standard interfaces for:
    - discovery // maintainance // reservation // configuration
    - plus further custom interfaces

- **“Gator Tech” (Univ. Florida) ⇒ Atlas** (commercial) layered arch. below
Gator Tech: Prof. Helal & Groudn: [www.icta.ufl.edu](http://www.icta.ufl.edu); Atlas company: [www.pervasa.com](http://www.pervasa.com)

- Emphasis: sensors (+actuators) & networked embedded devices - layer 1
- OSGi exploited for customizing and maintaining L.1 Hardware in dedicated L.2
- L.3 contains 3 large parts (cf. UC ‘world’):
  - context management layer
  - service layer: SOA dominating paradigm for autonomous software components in UC
  - knowledge layer: reflects problems of large-scale service composition: service discovery & interaction via machine readable descriptions of service semantics @ runtime;
- strict SoA concept → appl. development = service composition → top layer: composition tools

Atlas architecture = “modified” GatorTech:
Better **product-to-human** communication required
- Guide all users along product lifecycle

Better **product-to-product** communication required
- Less top-down, more bottom-up integration
- Open communication and self-organization

**Embedding of proactive knowledge** in products
SmartProducts Research Agenda

- Human - Computer Interaction
- Ubiquitous Computing
- Knowledge Management
- Smart Products & Proactive Knowledge

- Life-cycle spanning user support
- Support for procedural & problem solving knowledge
- Emerging knowledge
- Multimodal interaction
- Proactivity (also: Knowl. Mgmt.)
- Context-awareness
- Self-organized embedding in SP environments
- Distributed storage of proactive knowledge
Application Domain – Aircraft Manufacturing

Account for diversity of product lines

- Support blue-collars in manufacturing aircrafts
- Each aircraft is linked to its digital representation
- Smart tools to check working procedure and update knowledge about aircrafts
Enabling collaboration of various smart products in an open environment for coaching the end-user

Cooking Guide, Shopping Support & Recipe Recommendation
- Suggests recipes based upon available food, utensils, nutritional needs, etc.
- Supports and influences user while shopping
- Guides the user step-by-step through cooking procedure
Application Domain – Smart Vehicle

Reducing complexity of technical products

- **Manufacturing**
  - Support operator in assessing correct installation and functioning of vehicle components

- **Use**
  - Support driver in using vehicle functionalities (e.g. mounting snow chains)
  - Provide feedback to driver regarding his driving behavior to maintain value of the car

- **Maintenance**
  - Involve user in solving main dysfunctions of vehicle
Ubiquitous Computing:
- smart devices, roaming users, zillions of nodes
- inevitable, even reality today...
- ... but integration remains a grand challenge
- Challenge as five large research issues: S.C.A.L.E.
  - scalability
  - connectivity
  - adaptability
  - liability
  - ease-of-use

- what else to remember:
  - the many termy & buzzwords 😞
  - reference architecture and “ideas behind” visions, organizations