Towards a Common Platform for Ubiquitous Computing

Fritz Hohl
hohl@sony.de

Advanced Development Center Stuttgart (ATCS)
Telecommunication Research & Development Europe (TRDE)

SONY
International (Europe) GmbH
Observation

If we want to have ubicomp applications like this:

- controlling the surrounding temperature by taking into account the temperature history of a user (context-aware)
- navigation of two users so that they can meet (location-aware)
- finding all Italian restaurants that are open before 15:00 hrs (GIS)
- finding and controlling devices in the room the user is currently located in (ad-hoc computing)
- visualizing water pipes in a wall (augmented reality)

we experience three main problems
Hardware Problems

- **lack of cheap ubiquitous wireless data networks**
  - will be solved with the appearance of Bluetooth, GPRS, UMTS, and their successors

- **lack of easy-to-install positioning systems that work everywhere**
  - while (D)GPS seems to be applicable for a broad range of applications (but not for all) outdoors, *indoors*, equivalent technologies are missing
  - for some indoor application areas, cell-based techniques such as GSM- and CDMA-based user tracking, infrared beacons or barcode tags can be used
Software Problems

- there are some software components
- but
  - which to select?
  - they do not inter-operate
- these problems result from a lack of an open global platform for ubicomp applications
  - two consequences:
    - every application has to be developed from scratch
    - only self-contained applications are possible, i.e. such that have everything “on board”

we need such a platform!
What Means “Open Global Platform”? 

A family of interoperable, yet separately usable component interfaces (APIs or protocols)

- requirements:
  - components shall be suited for programs, not humans to allow maximum processability and to avoid problems
  - a separation between applications and services so shared functionality can be separated
  - an initial set of services shared by applications so a first support exists for applications
  - a mechanism to discover services on a global scale so a world-wide coverage is in range
  - a library so applications do not have to implement themselves
Platform: Requirements

- **a framework for adding new services**
  as not every service needed in the future can be foreseen
  as new services might be offered based on existing ones

- **the possibility of an incremental growth**
  to allow a global system by distributed effort

- **an open infrastructure**
  to allow incremental growth
  open means: everybody can operate parts of the platform
  no need for central authority

- **a scalable infrastructure**
  to be suitable for small as well as for global coverage
How A Such Platform Could Look Like?

- a single name concept for:
  - objects
  - servers
- a name server, associating names and areas to addresses
- an extensible data model (preferably the same among all components)
- a service directory and trading mechanism
- a uniform low-level communication mechanism (sockets in the worst case)
- an event observation mechanism
- some shared high-level communication mechanisms (e.g. events, geocast)
- a public key infrastructure (just select one)
How A Such Platform Could Look Like?

- server components for e.g.:
  - object data (as in the Active Spaces approach)
  - context data (if different from above)
  - spatial model data
  - navigation
  - basic metaphors such as VITs, Post-Its, Virtual Displays
  - legacy information systems (e.g. web servers)
  - annotation to objects
  - user display access
  - etc.
How A Such Platform Could Look Like?

- using such a platform, creating applications would be easy:
  - application on user’s device:
    - get `anItem` from `user`
    - append `anItem` to `shoppinglist(aShop)`
  - application on server:
    - `onEvent((user approaches aShop) && (aShop.isOpen()))`
      - `user.openWindow("You wanted to buy", shoppinglist)`
- Voila, here is your location-aware shopping list
Advantages

- an application can be built using out-of-the-shelf components
- no need for re-inventing wheels (might become square then)
- interaction between components possible
- single components can be operated permanently, thus being used by all applications
- data can be exchanged (e.g. spatial models)
- a platform normalizes the language used by the applications...
Conclusion

- if our aim is to support ubiquitous computing applications
- we need an open global platform that specifies a family of component interfaces
- such a platform would offer many advantages

فاقن We should even try to combine our efforts to come up with a common platform!
- easier achievement of the critical mass
- more impact of results
Thank You