An Architecture for Context Prediction

18. April 2004, Linz

Doctoral Colloquium of PERVASIVE 2004

Rene Mayrhofer
Institut für Pervasive Computing
Johannes Kepler Universität Linz, Austria
rene@soft.uni-linz.ac.at

Vision

A „Personal Digital Assistant“ that can live up to its name.

Outline

- Introduction
  - Motivation and Problem Statement
  - Context awareness
  - Proactivity
- Approach
- Architecture
- Implementation
- Preliminary Results
- Contribution
- Open Issues

Motivation

Problem:

- Most information appliances are difficult to use for non-technology-savy users
- Devices only react to user input

Aim:

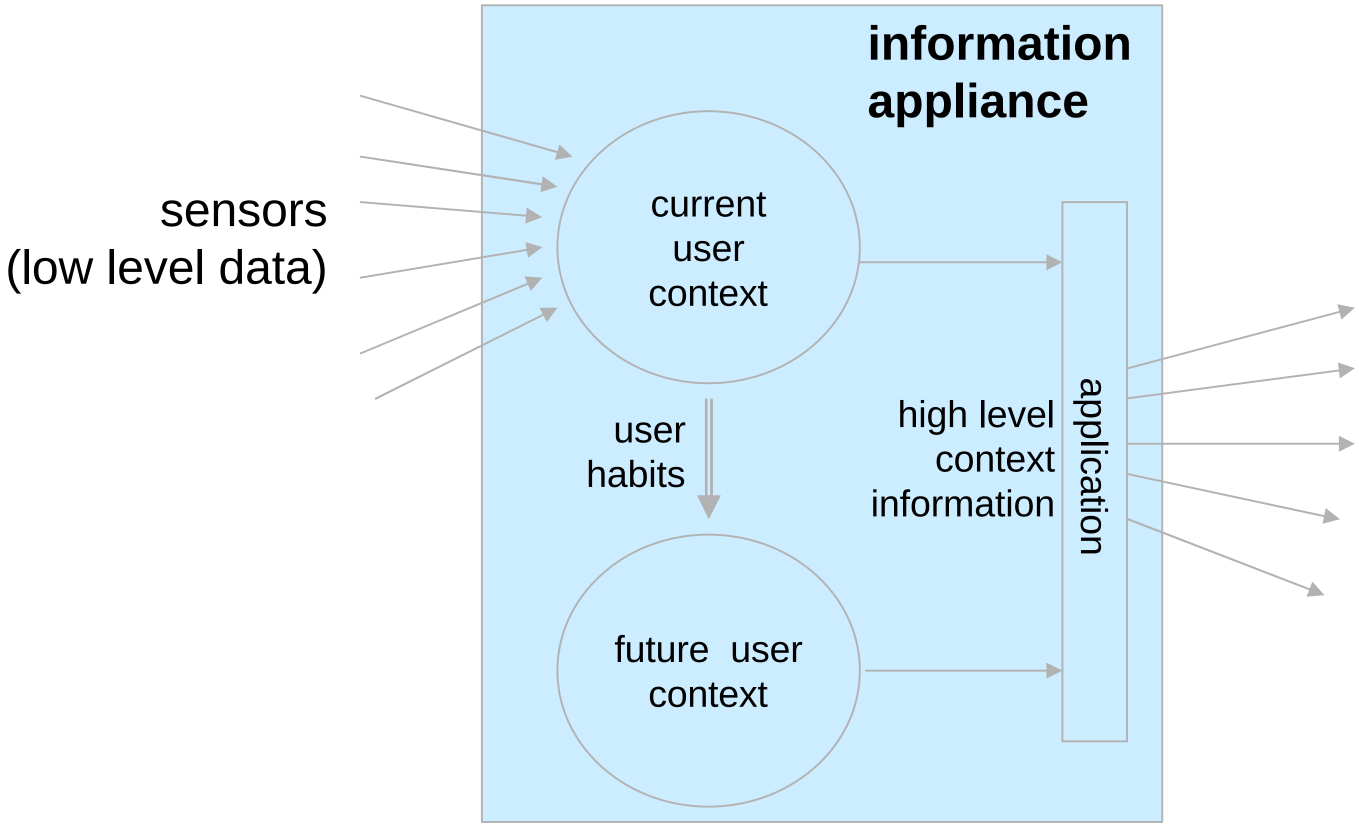
- Make information appliances „smarter“ in a sense that they are easier to use
- Devices should be proactive
- Devices should adapt to the user

Problem statement:

How can an information appliance infer its (or its users) current context and predict future context ?

General Approach

- Personal information appliances should learn from user‘s habits
- Exploiting multiple sensors for context awareness [Schmidt 2002]
- Predicting future user context by learning from the past



[Schmidt 2002] A. Schmidt: “Ubiquitous Computing – Computing in Context”, PhD Thesis, Lancaster Univ.

Context awareness

- Many definitions for context, e.g. by Dey [Dey 1999] as

  any information that can be used to characterize the situation of an entity, where an entity can be a person, place or a physical or computational object

- Context has many aspects
- Using multiple simple sensors seems more reasonable to capture different aspects of context

INFORMATIK
UNIVERSITÄT LINZ

[Dey 1999] A. Dey, G.D. Abowd, D.Salber: „A Context-based infrastructure for smart environments“

Proactivity

Proactive vs. reactive behavior of a system

- Determines if an information appliance merely reacts to changes in its environment or if it can act in advance
- Definition of proactivity based on system states [MRF 2003a]
- Internal state of a (Moore) state machine depends on last state and system inputs: $q_t = d\left(q_{t-1}, a_{t-1}\right)$
- Reactive system: output depends on current (and implicitly on past) states:

$$b_t = l\left(q_t\right)$$

- Proactive system: output depends additionally on predicted future system states:

$$b_t = l\left\langle q_t, \overline{q}_{t+1}, \overline{q}_{t+2}, \ldots, \overline{q}_{t+m}\right\rangle$$

[MRF 2003a] R. Mayrhofer, H. Radi, A. Ferscha: „Recognizing and Predicting Context by Learning From User Behavior“, Proceedings of MoMM2003, OCG, September 2003

Related Work

- TEA
- Smart-Its
- Context Toolkit
- Robotics
- CIS
- Neural Network House, Aware Home, MavHome
- …

Outline

- Introduction
- Approach
- Architecture
- Implementation
- Preliminary Results
- Contribution
- Open Issues

Research Approach

1. Define research goal
2. Derive requirements
3. Select architecture according to requirements
4. Select algorithms according to requirements
5. Verify algorithms using test data
6. Validate architecture and algorithms using real world data

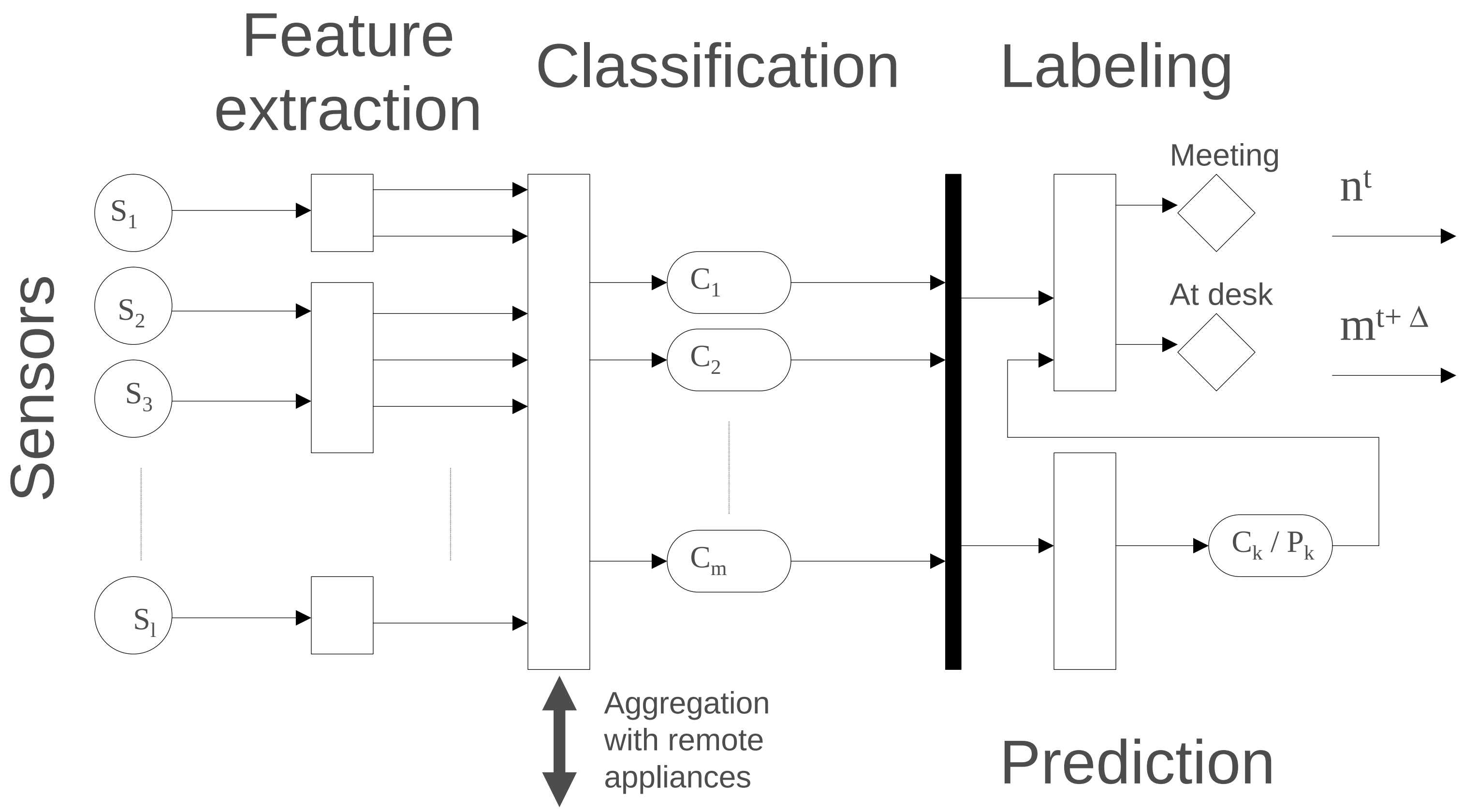
Motivation for “yet another” context awareness middleware:

- Many research projects on context awareness are either ad-hoc applications designed after available sensing technology or infrastructure oriented architectures
- “Ready to use” software frameworks for resource limited devices (e.g. PDAs, mobile phones) do not seem to be available

Outline

- Introduction
- Approach
- Architecture
  - Sensor Data Acquisition
  - Feature Extraction
  - Aggregation
  - Clustering
  - Labeling
  - Prediction
- Implementation
- Preliminary Results
- Contribution
- Open Issues

Architecture



$<s_1, s_2, \ldots, s_l>_t \longrightarrow <f_1, f_2, \ldots, f_n>_t \longrightarrow <c_1, c_2, \ldots, c_m>_t \longrightarrow <c_1, c_2, \ldots, c_m>_{t+\Delta}$

Input vector (Sensor vector) — Feature vector — Class vector — Future class vector

[MRF 2003a] R. Mayrhofer, H. Radi, A. Ferscha: „Recognizing and Predicting Context by Learning From User Behavior“, Proceedings of MoMM2003, OCG, September 2003

Challenges

- Context recognition and prediction should be embedded in information appliances with limited resources
- Learning and adaptation should happen on-line without explicit training
- User interaction should be kept to a minimum and be un-obtrusive
- Feature vectors are typically highly heterogeneous
- Prediction should respect trends and periodic patterns in context history

Sensors for (mobile) information appliances

Typical „sensors“ available for monitoring the user context:

- Time
- Application/Window manager
- Brightness
- Microphone
- Bluetooth
- Wireless LAN
- Docked / undocked

Other suitable sensors can be connected:

- GPS
- GSM
- Compass
- Accelerometer
- Tilt sensor
- Temperature sensor
- Pressure sensor

Sharing of sensor data between appliances

Feature Extraction

- Raw sensor data is transformed into more meaningful features
- Feature extraction exploits domain-specific knowledge
- Multiple features extracted from a single sensor

⇒ **High-dimensional input vectors**

- Different types of features:
  - Numerical (continuous): e.g. brightness sensor
  - Numerical (discrete): e.g. number of access points in range
  - Ordinal: e.g. day of week
  - Nominal: e.g. WLAN-SSID, list of Bluetooth devices in spatial proximity

- Only two operations necessary for each feature:
  - Distance metric
  - Adaptation operator

[MRF 2003b] R. Mayrhofer, H. Radi, A. Ferscha: „Feature Extraction in Wireless Personal and Local Area Networks“, Proceedings of 6th IFIP MWCN 2003, World Scientific, October 2003

Classification: Introduction

- Classifies feature vectors and finds common patterns in sensor data
- Different types of classification algorithms
  - Type (partitioning / hierarchical)
  - Soft / hard classification
  - Supervised / unsupervised

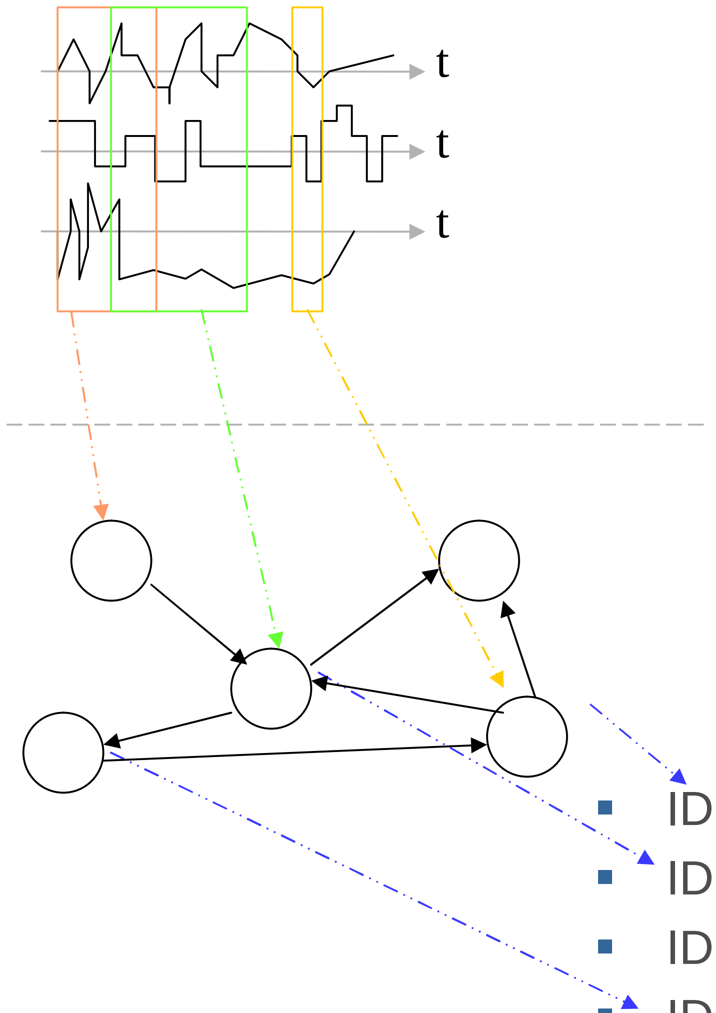
- Requirements for classifying user context in information appliances:
  - On-line learning
  - Adaptivity
  - Variable number of classes and variable topology
  - Detecting clusters in sub-spaces
  - Soft classification
  - Noise resistance
  - Limited resources
  - Simplicity
  - Interpretability of classes / protection of data privacy

Classification: Algorithms

Algorithm	Network topology	Topology preserving	Competitive
SOM [SAT 1999]	fixed	yes	soft
RSOM	fixed	yes	soft
K-Means	fixed	no	hard
Leader	variable	no	hard
Growing K-Means [DWM 2002]	variable	no	hard
Neural Gas	variable	no	soft
Neural Gas + Competitive Hebbian Learning	variable	yes	soft
Growing Neural Gas [Fri 1995]	variable	yes	soft
Incremental DBSCAN [SWX]	variable	No	hard

[Fri 2003] B. Fritzke: „A Growing Neural Gas Network learns Topologies", Advances in Neural Information Processing Systems (7), MIT Press

Labeling: Assigning user-defined labels to user context



- 1:{0,1} assignment of (meta-) clusters to context names
- Two possibilities:
  - If the (meta-) clusters at the output of the classification step are stable, direct assignment to names
  - If the (meta-) clusters are unstable due to learning and adaptation, use a second, simple classification step [Lae 2001]

- ID 1 = “at work“
- ID 2 = “meeting“
- ID 3 = “lunch”
- ID 4 =
- ID 5 = “at home“

[Lae 2001] K. van Laerhoven and S. Lowette: “Real-time analysis of data from many sensors with neural networks”, Proceedings of ISWC, IEEE Press, October 2001

Prediction of User Context

- Recognized context classes can be regarded as “states“ of an abstract state machine
- Monitoring the state trajectory allows to predict future states

- Prediction algorithm requirements for predicting context “states”:
  - Unsupervised model estimation
  - On-line learning
  - Incremental model growing
  - Confidence estimation
  - Automatic feedback
  - Manual feedback
  - Long-term vs. short-term

- Architecture allows prediction algorithms to be realized as plug-ins

⇒ Algorithms can be changed according to the specific application needs

Prediction: Options for predicting future context

- Prediction should be based on context class vectors
- Advantage: future class vector can be handled like current ones (e.g. labeling)
- In principle two options:
  - using each dimension of the context class vector (i.e. each class membership) as a continuous time series

    ⇒ relationship between context classes is not accounted for

  - using the whole vector for prediction as aggregated, categorical time series

    ⇒ explicitly taking relationship into account by constructing a single, unified model over all context classes
    But: no algorithm found during literature search which performs a multi-dimensional forecast and fulfills other criteria, thus only the best matching context can be taken into account

AAACCBCCAAADDDDDDDEEEEEEBBAAAA……..

Prediction: Aspects of predicting future context

(At least) two different aspects of prediction:

- Periodic patterns (regular events) in the class vectors: context classes which are active at regular time intervals
  e.g. meetings, working times, lunch, dinner, etc.

  Intervals can not only be daily, weekly, monthly, etc. but also e.g. every 2 hours, every 3 days, etc.

- Sequential patterns: sequences of context classes
  e.g. preparing for a conference

Both should be accounted for, but it is unlikely that a single algorithm will be capable of both

Prediction: Possible algorithms (1)

- ARMA: suitable for single-dimension, continuous time series forecast (option 1)
  But: relationship between context classes not regarded

- ANN (e.g. back-propagation MLP): suitable for multi-dimension, continuous time series forecast
  But: needs known output (supervised learning) and long training time with many samples; no online mode, no incremental model growing, no confidence estimation

- HMM: allows discrete forecast
  But: assumes statistical independence of events, which is definitely not the case for subsequent contexts

Prediction: Possible algorithms (2)

- MavHome project [Das 2002] uses prediction algorithms (currently) for predicting user locations
- Two algorithms for covering both aspects:
  - *Active Lempel-Ziv* for finding and predicting sequential patterns: online / incremental
  - *Episode Discovery (ED)* for identifying events at some regular intervals or in response to other events
- Back propagation ANN for mixing results of both algorithms

[Das 2002] S.K. Das, D.J. Cook, A. Bhattacharya, E. Heiermann, T. Lin: „The role of prediction algorithms in The MavHome smart home environment", IEEE Wireless Communications 9(6)

Outline

- Introduction
- Approach
- Architecture
- Implementation
- Preliminary Results
- Contribution
- Open Issues

Implementation as Software Framework

Cross-platform software framework

- Written in C++ with Java interface
- Runs on Win32, Windows CE (>=3.0), Linux IA32 and ARM and Symbian OS
- Based on a plug-in concept:
  - Feature containers ($\equiv$ sensors) and Features
  - Classification methods
  - Prediction methods

  which are configurable and dynamically loaded during startup
- Labeling is loosely coupled via COM+ or SOAP
  $\Rightarrow$ separation of UI and background context inference
- Written with performance constraints in mind, for embedded devices

Current Status

- Architecture finished
- Step 0 (sensor data acquisition):
  - finished for currently available sensors
  - new sensor can easily be integrated by writing new adapters
  - finished two general-purpose sensors for string lists: arbitrary commands or PHP scripts
- Step 1 (feature extraction):
  - finished for currently available sensors
  - new features (for new sensors or for currently used ones) can be added by implementing sub classes of abstract base classes
- Step 2 (classification):
  - implementation finished
  - Large data sets with better results than with comparable SOM
  - tuning of parameters might still necessary for new data sets
- Step 3 (labeling):
  - Tray bar application allows to interactively assign labels
  - Communication with background process via open protocols (COM+, SOAP)
- Step 4 (prediction):
  - Implementation of Active LeZi (option 2, categorical)
  - Implementation of averaging predictor (option 2, categorical)
  - Implementation of incremental covariance function (option 1, numerical)
  - ARMA and ANN on real world data sets in Matlab and ITSM2000

Outline

- Introduction
- Approach
- Architecture
- Implementation
- Preliminary Results
- Contribution
- Open Issues

Preliminary Results

Classification:

- Evaluation of K-Means, SOM and extended LLGNG
  - K-Means: lowest error for 6 clusters: 0.7451
  - SOM: lowest error with 71x21 (=1491) units in output layer: 0.5659
  - Extended LLGNG: 9 meta clusters out of 109 clusters with error: 0.0069
- GNG achieves lower overall classification error (for unsupervised clustering) with less clusters than SOM

Prediction:

- Evaluation of averaging predictor, ARMA, back propagation MLP and Active LeZi
  - Best results with seasonally corrected ARMA model for single dimension (option 1)
  - MLP nearly unusable
  - Active LeZi shows performance comparable to simple averaging predictor
- Still no good algorithm found

Outline

- Introduction
- Approach
- Architecture
- Implementation
- Preliminary Results
- Contribution
- Open Issues

Contribution

- Development of an architecture for context prediction with:
  - Flexibility due to simple interfaces between well-defined steps
  - Clear separation of concerns
- Implementation of the architecture with:
  - Use of heterogeneous feature vectors
  - Context prediction
  - Cross-platform compatibility

Outline

- Introduction
- Approach
- Architecture
- Implementation
- Preliminary Results
- Contribution
- Open Issues

Open Issues

- Evaluation of the architecture as a whole
- Selection of a prediction algorithm and (possibly) implementation as plug-in
- Parameter optimization with different data sets
- Live-tests with mobile devices instead of offline processing of collected data
- User feedback for improving classification and prediction quality
- Interpretation of automatically created context classes at application level and visualization ("how is a class composed and why is it different from another one ?")
- Framework enhancements (e.g. additional platforms, additional language interfaces, etc.)
- …

Summary and Outlook

- A "*Personal Digital Assistant*" and information appliances in general should become proactive to achieve a wider acceptance
- Context awareness is one possibility to achieve proactivity in applications
- Architecture for *Predicting Context* has been created and implemented as a plug-in framework
- Context recognition (Steps 0 – 2) finished for artificial and preliminary real world test data, more real world data still to be collected
- Labeling (Step 3) needs research on user interfaces and context hierarchies
- Prediction (Step 4) needs a quantitative comparison of known algorithms and probably some way of mixing results of different algorithms

- Next steps in research: comparing prediction algorithms on real world data sets, qualitatively and quantitatively
- First candidates:
  - ARMA (simplicity) + automatic model selection (covariance)
  - ANN, SVM (good results on other data sets, but slow training)
  - Active Lempel-Ziv (first implementation finished), Episode Discovery
  - HMM, FHMM, VDHMM

“If we knew what it was we were doing, it would not be called research, would it?”

Albert Einstein