A Context Based Approach for Application Personalization

Ph.D. Proposal

October 2nd 2006

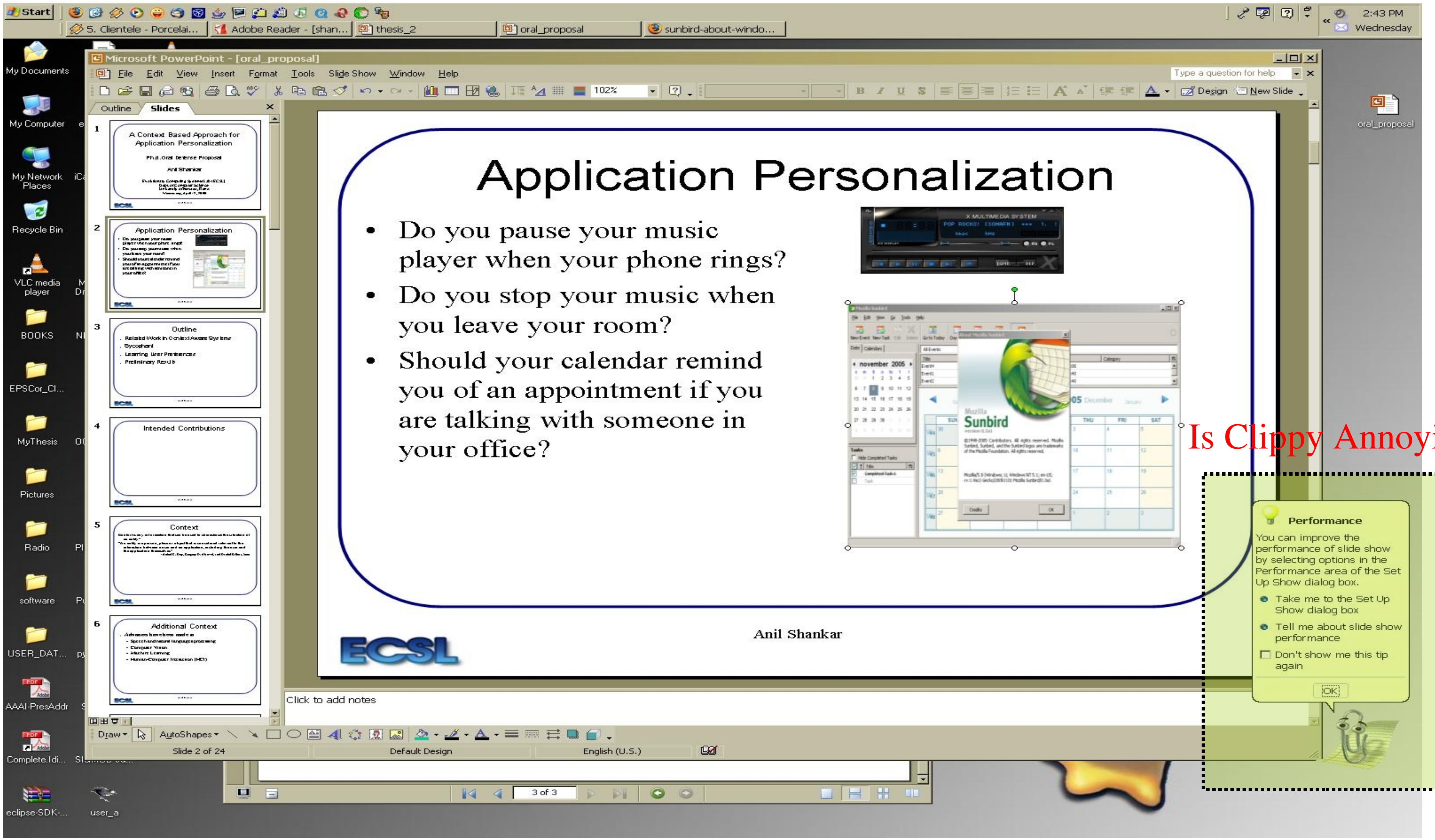
Anil Shankar

Evolutionary Computing Systems Lab (ECSL)

Dept. of Computer Science

University of Nevada, Reno

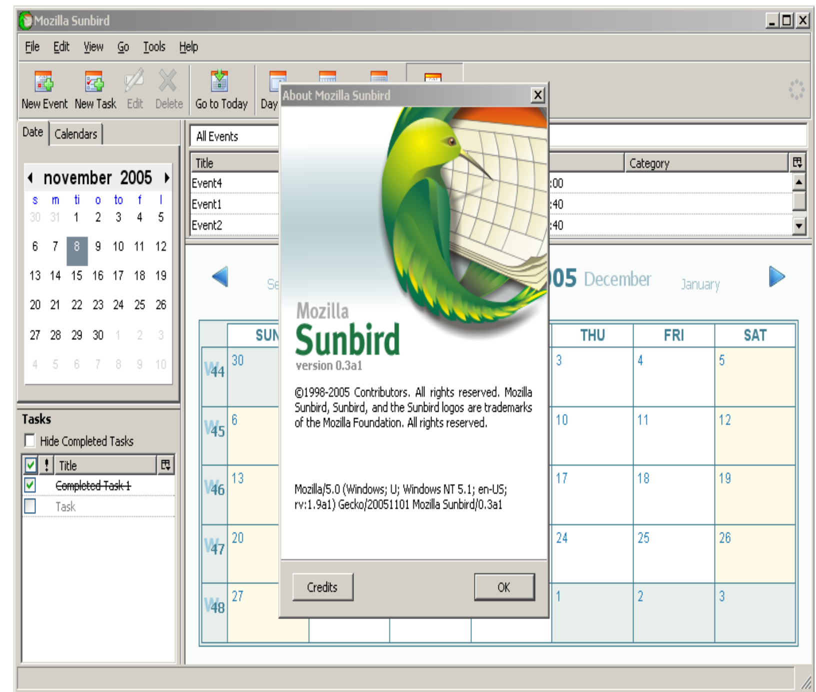
Application Personalization Current Status



Is Clippy Annoying

Application Personalization

- Do you pause your music player when your phone rings?
- Do you stop your music when you leave your room?
- Should your calendar remind you of an appointment if you are talking with someone in your office?



Outline

- Need for learning user preferences
- Related work in context-aware systems
- Sycophant
- Learning user preferences
- New results for 10 users
- Proposed work

Context

"Context is any information that can be used to characterize the situation of an entity"

"An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and the application themselves"

- Anind K. Dey, Gregory D. Abowd, and Daniel Salber, 2001

Present day computers use an internal clock, keyboard activity, mouse movements to provide context for an application's information processing.

Additional Context

- Advances have been made in
  - Speech and Natural Languages Processing
  - Computer Vision
  - Machine Learning
  - Human-Computer Interaction (HCI)

User-Context

"Any information regarding a user's presence or absence in the vicinity of a computer"

Internal user-context: keyboard activity, states of different user processes, mouse activity

External user-context: motion and speech in the user's environment

Hypothesis

- Simple sensors
  - Web-camera instead of a retina scanner or a gaze-tracking device
  - Microphone instead of a voice-authentication system
- Simple user-context
  - motion detection instead of face recognition, speech detection instead of speech recognition, keyboard and mouse usage, activity of different user processes

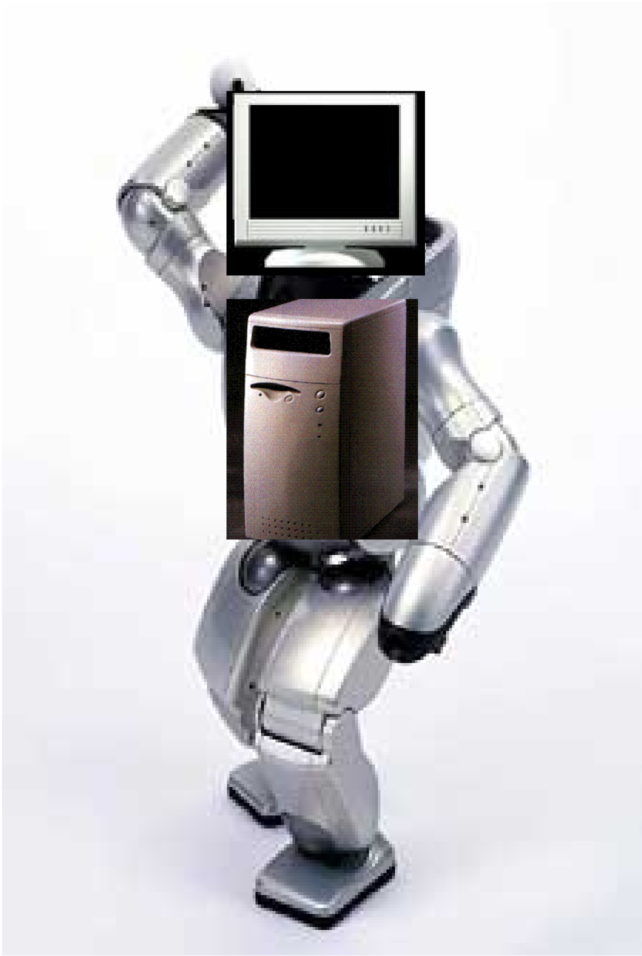
User-context helps applications to better personalize themselves to individual users

Related Work

- Sensor devices to localize a user
  - Kulkarni's *ReBa,* 2002
- Sensor-based statistical models to predict the state of interruptability of a user
  - Fogarty, 2005
- Quantitatively evaluate the effect of interruptions based on internal application events
  - Bailey, 2006

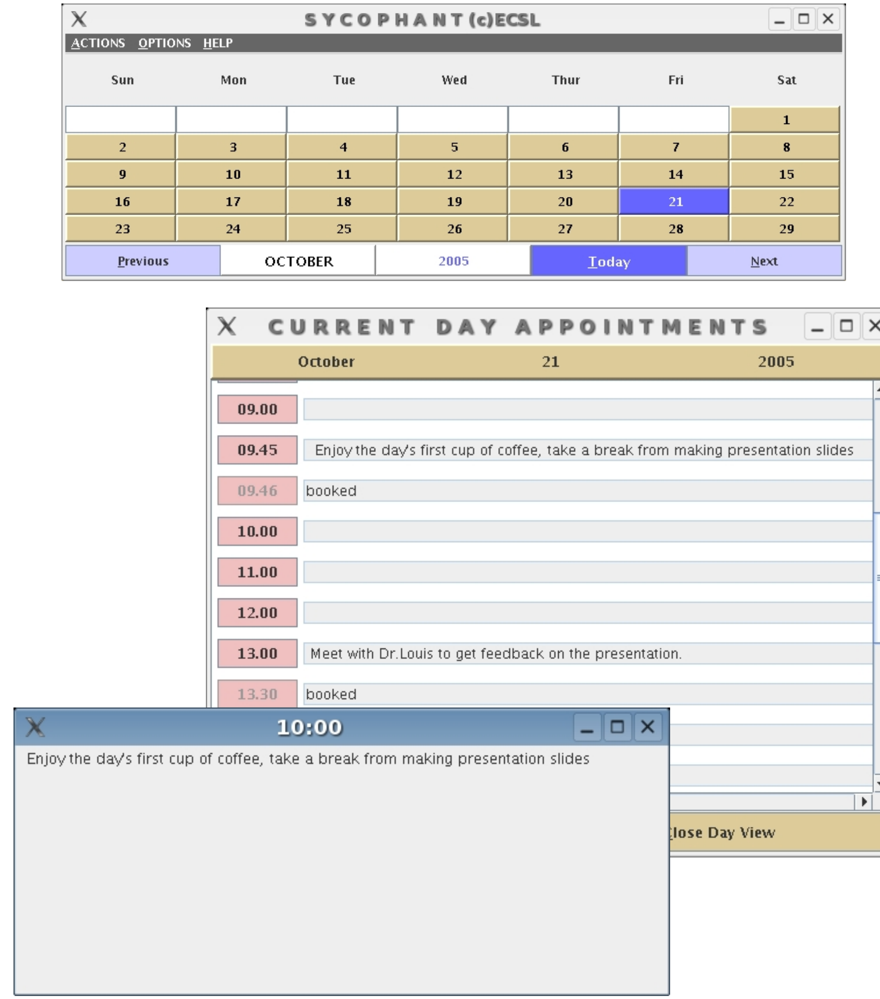
Our Work

- Use real sensors to gather user-context
- Model the desktop PC as a stationary robot with *effectors* (application action) and *actuators* (user model)
- In addition to predicting the state of interruptability of a user, we predict a user-preferred application action

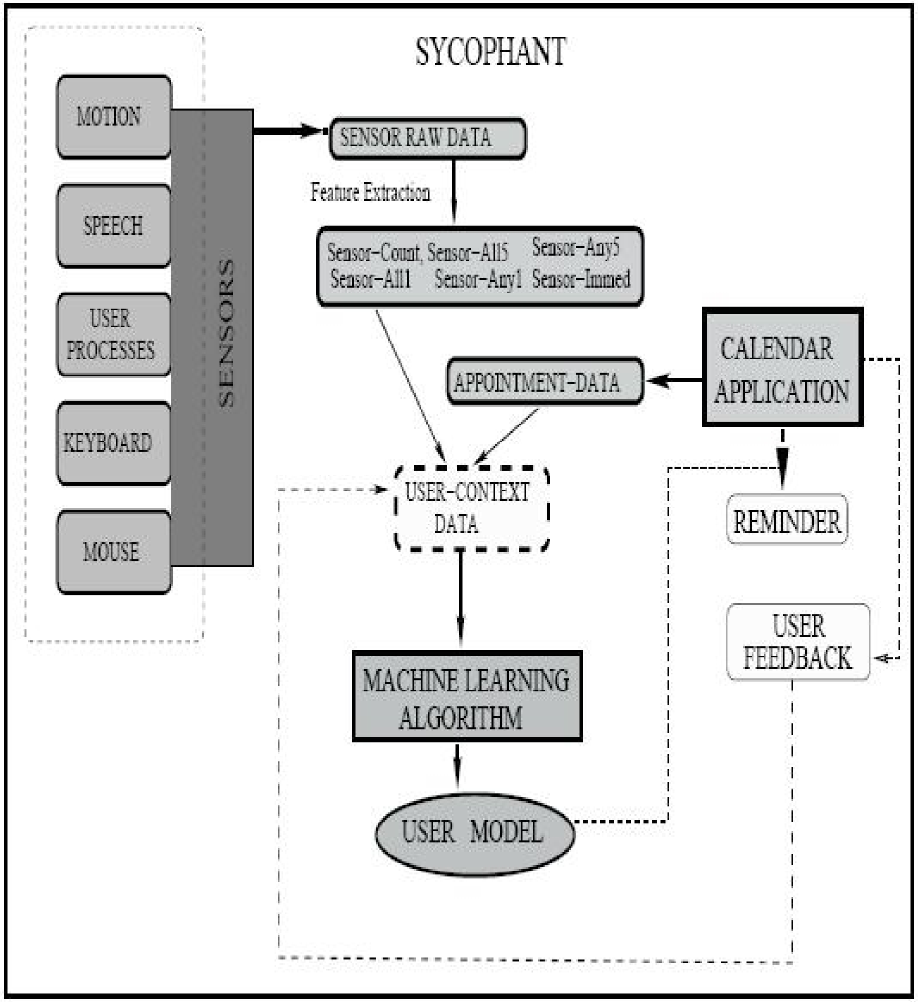


Sycophant's User Interface

- "Sycophant : A servile self-seeker who attempts to win favor by flattering influential people"
  - (source: http://dictionary.com)
- Framework that supports a calendaring application which learns to generate a user-preferred type of reminder for appointments and tasks

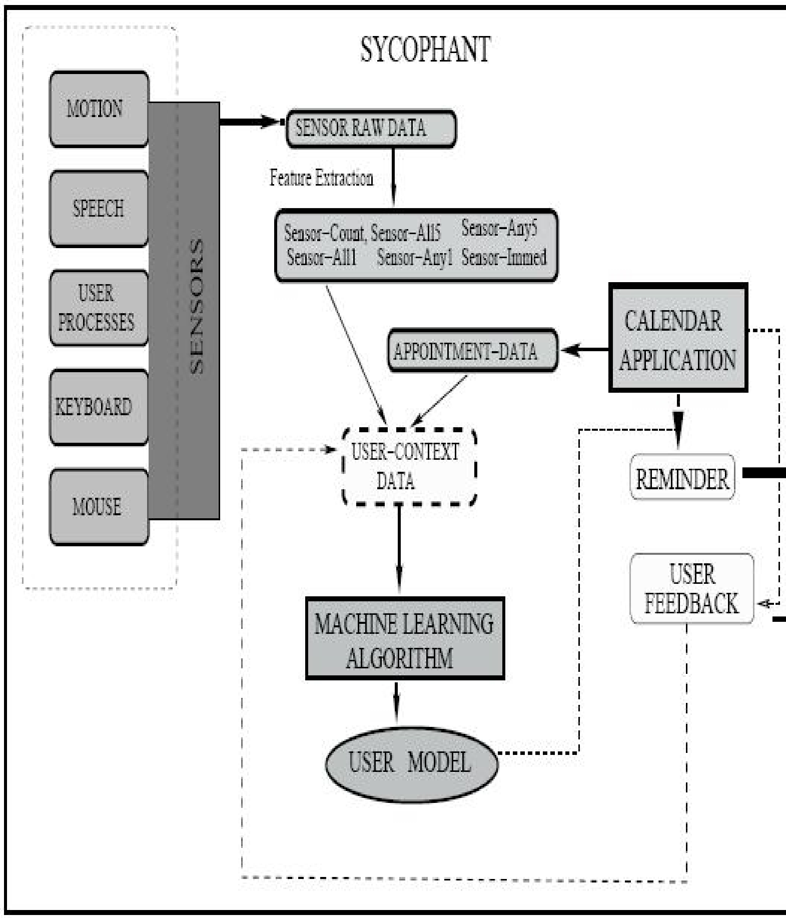


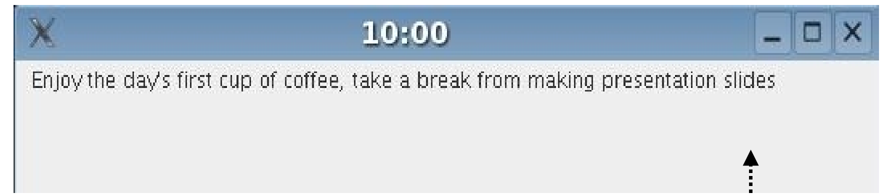
Sycophant's Architecture



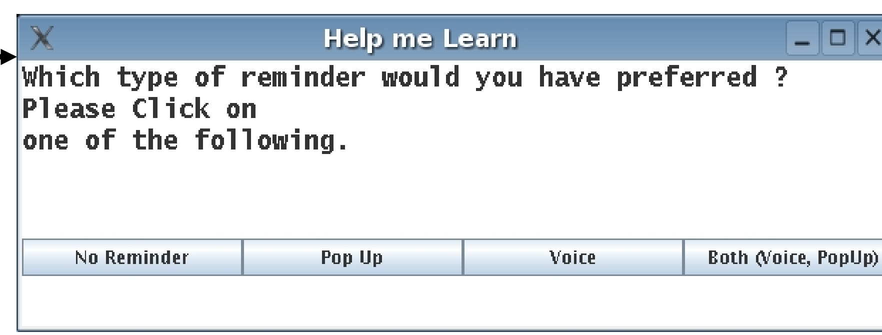
- Sensors collect internal and external user-context
- Machine Learning algorithms for building a user model
- User-feedback for reminder type
- Sensors operate in a binary mode

Sycophant's Reminder Types

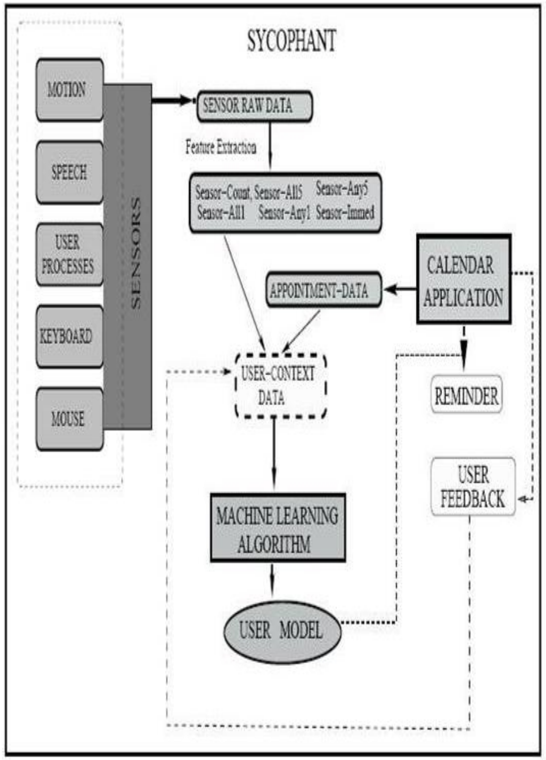




- No-reminder (type-0)
- Visual(type-1)
- Voice (type-2)
- Both Visual and Voice (type-3)



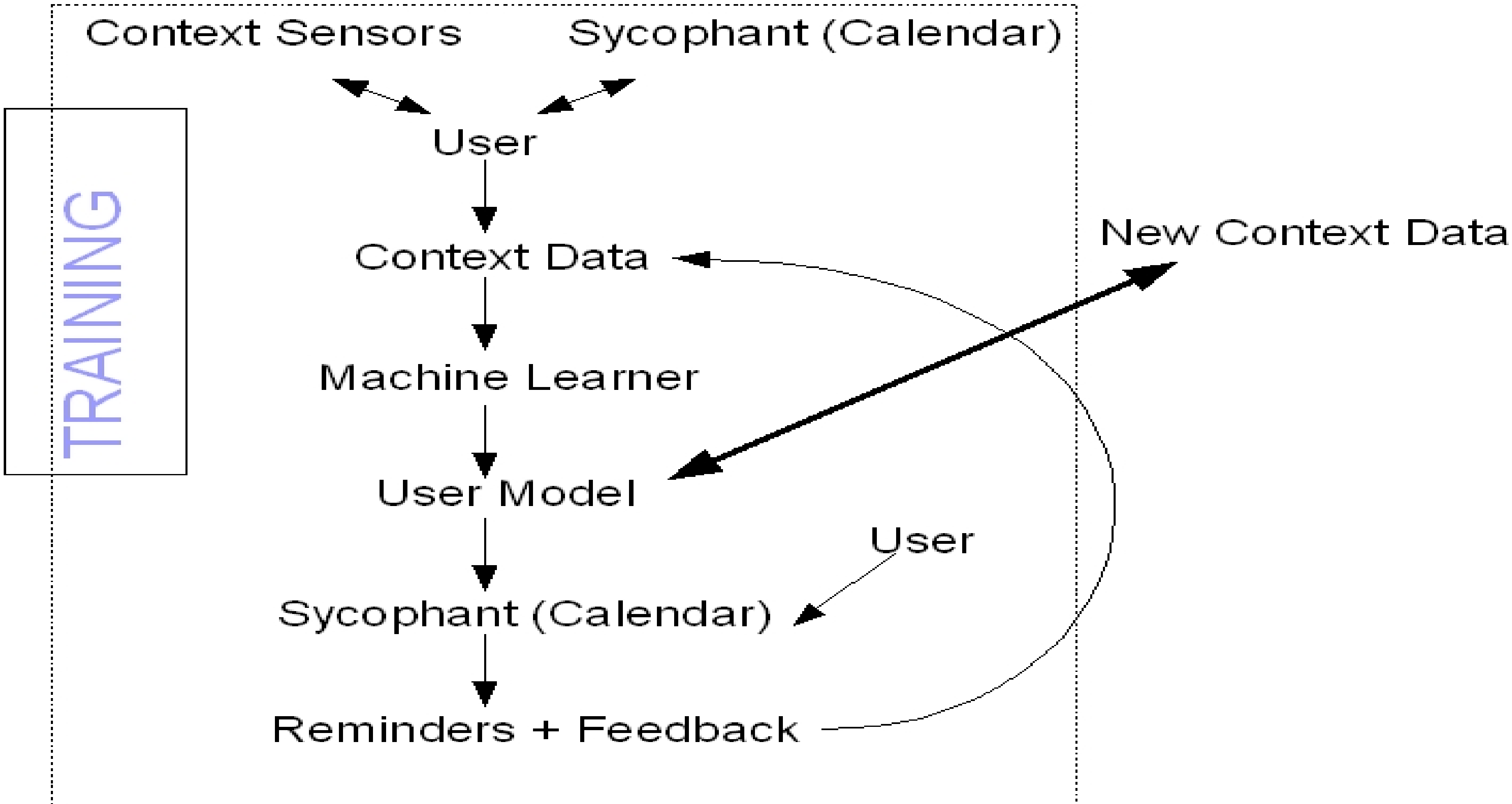
User-Context Data



Feature	Value(s)
User-identifier	user-3
Appointment Time	10.3000
Motion (Count, All5, Any5, All1, Any1, Immed)	10,7,1,0,3,1
Speech (Count, All5, Any5, All1, Any1, Immed)	20,1,1,1,1,1
Process-1(Count, All5, Any5, All1, Any1, Immed)	20,1,1,1,1,1
Process-2(Count, All5, Any5, All1, Any1, Immed)	20,1,1,1,1,1
Process-3(Count, All5, Any5, All1, Any1, Immed)	0, 0,0,0,0,0
Process-4(Count, All5, Any5, All1, Any1, Immed)	20,1,1,1,1,1
Process-5(Count, All5, Any5, All1, Any1, Immed)	0, 0,0,0,0,0
Keyboard(Count, All5, Any5, All1, Any1, Immed)	0, 0,0,0,0,0
Mouse(Count, All5, Any5, All1, Any1, Immed)	0, 0,0,0,0,0

user-3, 10.30, 10, 7, 1, 0, 3, 1, 20, 1, 1, 1, 1, 1, 20, 1, 1, 1, 1, 1, 20, 1, 1, 1, 1, 1, 20, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 20, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 (no-reminder)

System Overview



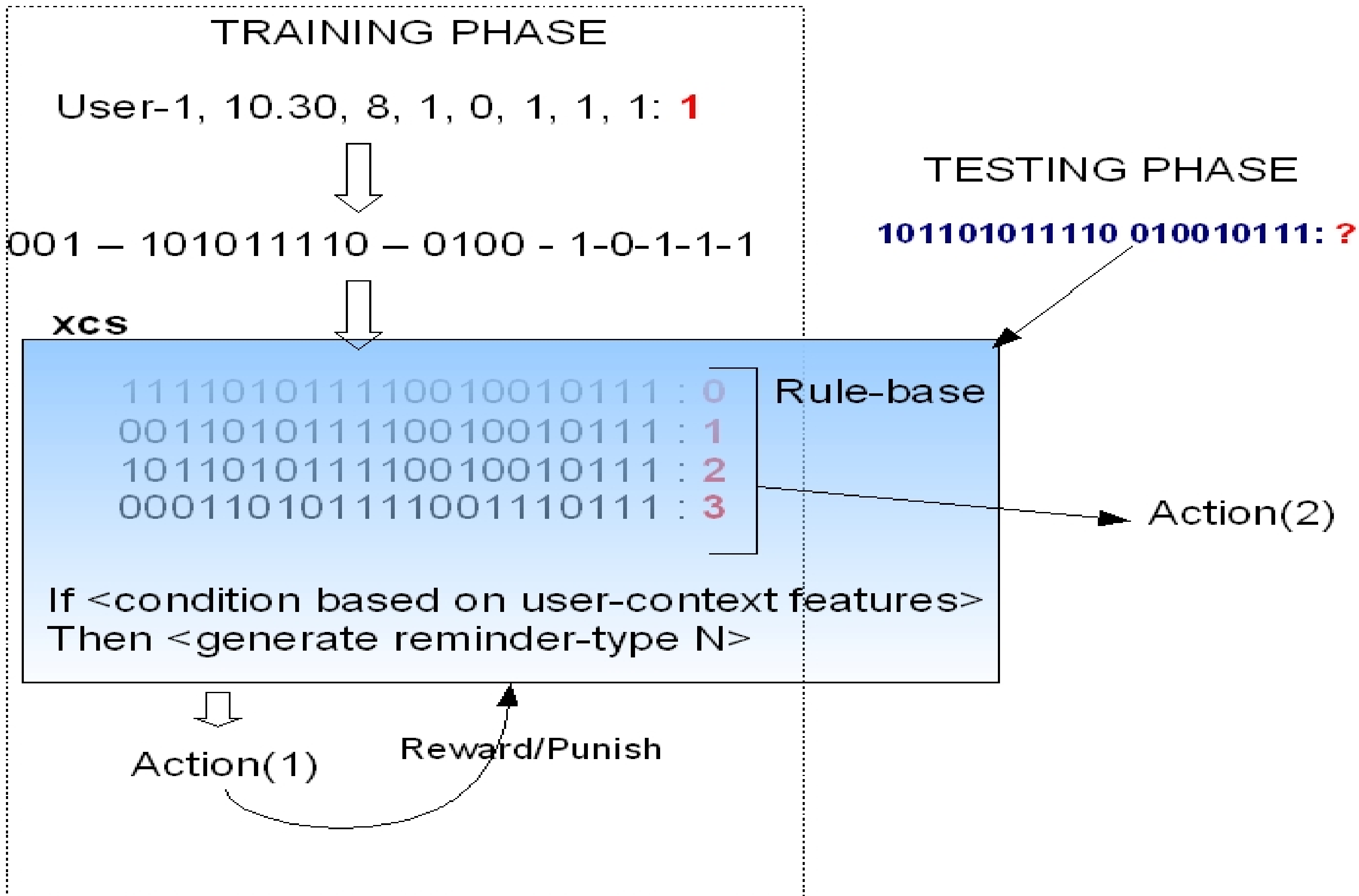
Learning User Preferences

- **Machine Learning**: A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"

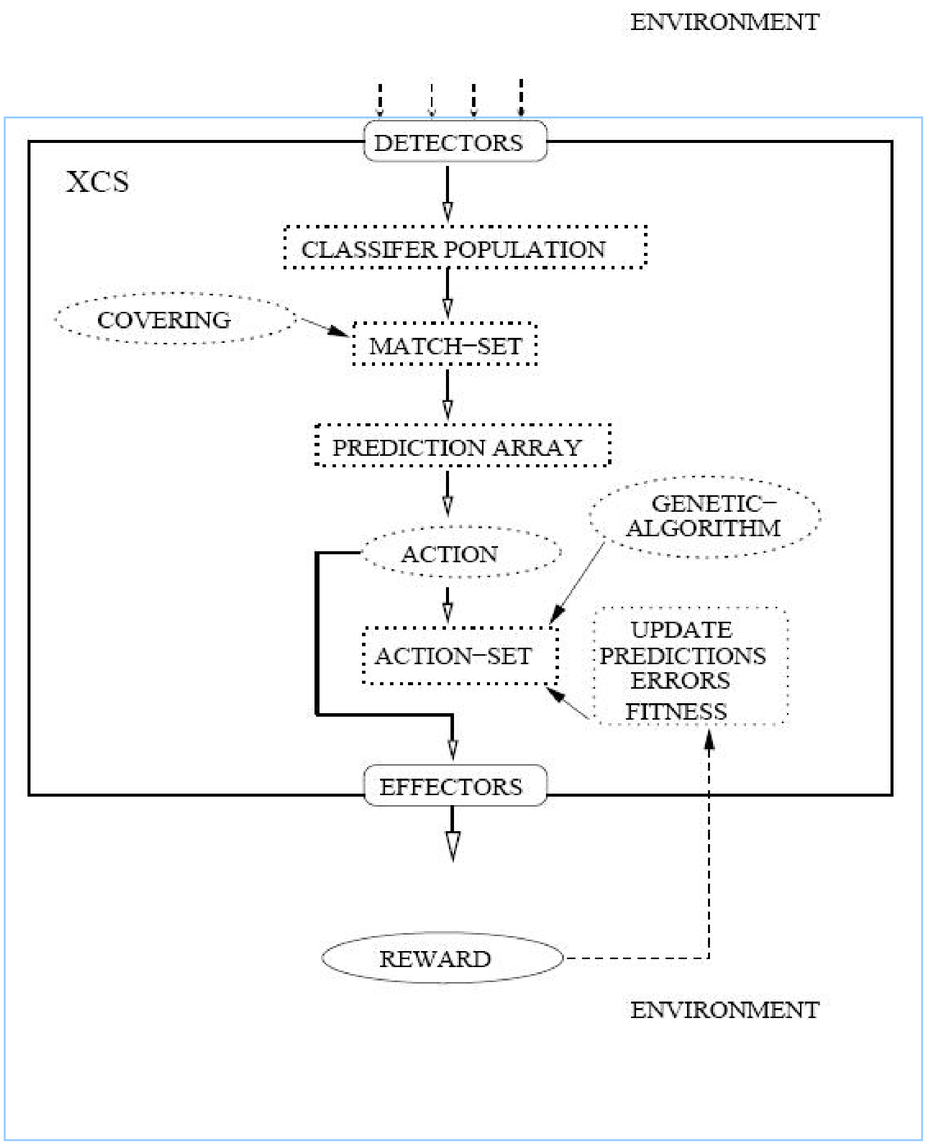
  -Tom Mitchell, *Machine Learning*

- **Techniques**: Decision Trees, Learning Classifier Systems

XCS Overview



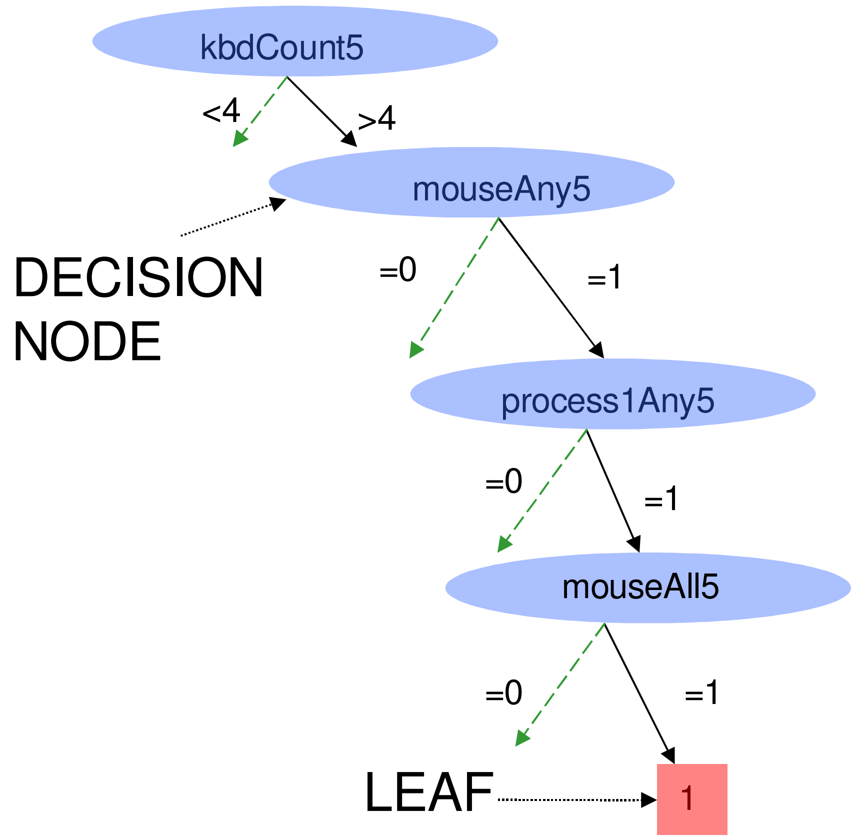
XCS



- **Genetics Based Machine Learning (GBML) technique**
- **Working**
  - Sample a user-context data exemplar
  - Create matching-set and action-set
  - Choose an action, receive feedback, update fitness

Decision Tree

- Tree structure
- Nodes, leaf
- For classifying a case, you traverse the tree until you reach a leaf
- Example decision-rule:
  - If kbdCount > 4
    - And mouseAny5 = 1
      - And process1Any5 = 1
        - And mouseAll5 = 1
        - **Then classify the reminder type as 1**



Results-1
Three different users

Two-class problem

Learning Algorithm	Zero-R	J48	XCS	XCS', N	XCS vs J48	XCS' vs J48
Data-Set						
Contex-Data-1	0.5677	0.7428	0.7460	0.7406, 5059	Equal	Equal
Contex-Data-2	0.5988	0.7885	0.9800	0.9300, 9678	Better	Better
Contex-Data-3	1.0000	1.0000	1.0000	1.0000, 4722	Equal	Equal
Contex-Data-allusers	0.6115	0.8685	0.8791	0.8788, 7699	Equal	Equal

- J48 and XCS achieve a minimum accuracy of **74** percent across all the user-context data sets (Except in case of ContextData-3)
- J48 and XCS outperfrom Zero-R(baseline)

Results-1…contd
Three different users

Four-class problem

Learning Algorithm Data-Set	Zero-R	J48	XCS	XCS', N	XCS vs J48	XCS' vs J48
Context-Data-1	0.5677	0.7000	1.0000	1.0000, 16289	Better	Better
Context-Data-2	0.5988	0.7267	1.0000	1.0000, 9433	Better	Better
Context-Data-3	0.9913	1.0000	0.9118	0.9118, 5071	Worse	Worse
Context-Data-allusers	0.3884	0.8236	0.8761	0.8236, 8493	Better	Better

- J48 and XCS achieve a minimum accuracy of **70** percent across all the user-context data sets (Except in case of ContextData-3)
- J48 and XCS outperfrom Zero-R (baseline)
- XCS outperforms J48 on the four-class problem

Results-2…contd

Two-class problem, J48

Data-Set	Performance with External Context	Performance without External Context
Context-Data-1	74.3100	71.75
Context-Data-2	81.0700	77.68
Context-Data-3	100.0000	100
Context-Data-all-users	86.4100	83.38

Removing external context degrades J48's performance

Results-2

Four-class problem, J48

Data-Set	Performance with External Context	Performance without External Context
Context-Data-1	67.4300	65.99
Context-Data-2	72.0300	71.18
Context-Data-3	99.1400	99.14
Context-Data-all-users	80.8100	78.15

Removing external context degrades J48's performance

Results-3…contd

Two-class problem, XCS

Data-Set	Performance with External Context	Performance without External Context
Context-Data-1	0.7406	0.2254
Context-Data-2	0.9300	0.8000
Context-Data-3	1.0000	1.0000
Context-Data-all-users	0.8788	0.4536

Removing external context degrades XCS's performance

Results-3

Four-class problem, XCS

Data-Set	Performance with External Context	Performance without External Context
Context-Data-1	1.0000	0.7745
Context-Data-2	1.0000	1.0000
Context-Data-3	0.9118	0.9557
Context-Data-all-users	0.8236	0.5460

Removing external context degrades XCS's performance

New Results
User Study Overview

- 10 users
- 4 separate sessions each lasting 45 mins.
- Read short/long articles
- Answer questions after reading the articles
- Hints (reminders) generated for users while reading
- User provides feedback specifying her preferred hint-type

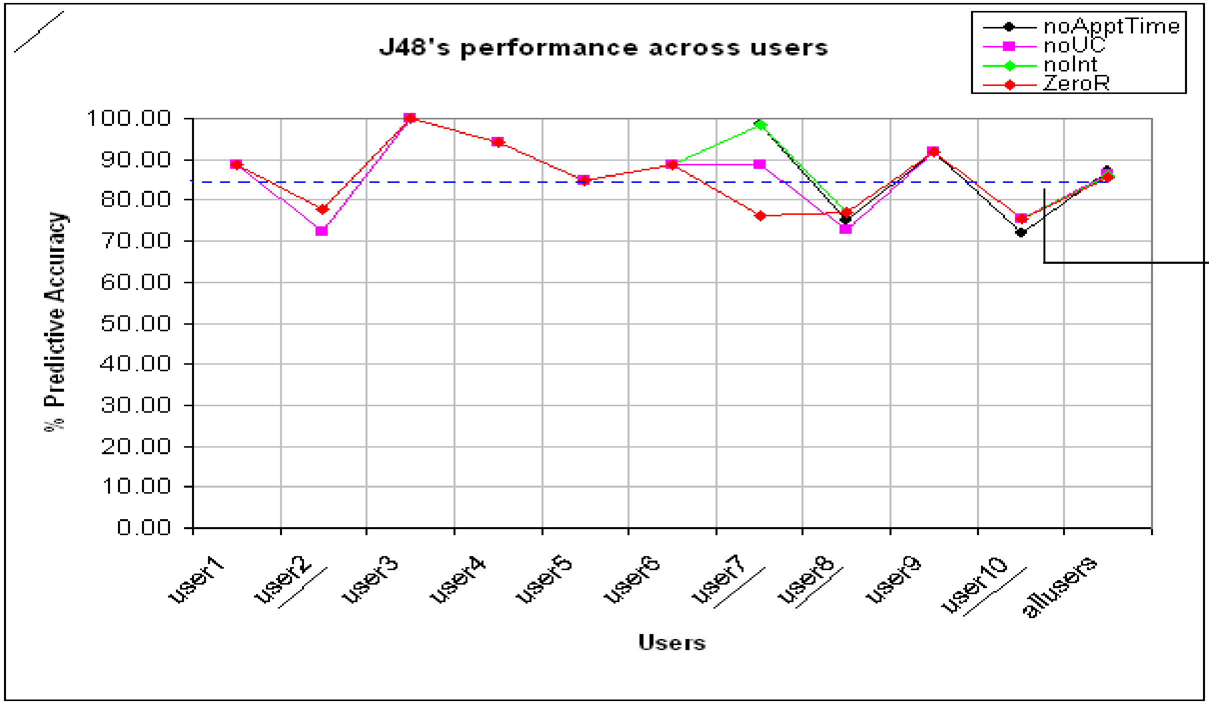
New Results

Experimental Design

Session	Task	Reminder Type* Order	Treatment
1	**Short Article**	**0,2, 3, 1**	**Talk, No-Music**
		3, 1, 2, 0	**Music, No-Talk**
		1, 0, 2, 3	**No-Music, Talk**
		2, 1, 0, 3	**Music, Talk**
2	**Long Article**	**3, 1, 0, 2**	**Talk, No-Music**
		1, 2, 0, 3	**Music, No-Talk**
		1, 3, 2, 0	**No-Music, Talk**
		3, 0, 2, 1	**Music, Talk**
3	**Short Article**	**1, 3, 0, 2**	**Talk, No-Music**
		2, 3, 1, 0	**Music, No-Talk**
		2, 0, 3, 1	**No-Music, Talk**
		3, 2, 0, 1	**Music, Talk**
4	**Long Article**	**0,3, 2, 1**	**Talk, No-Music**
		1, 0, 2, 3	**Music, No-Talk**
		1, 0, 2, 3	**No-Music, Talk**
		3, 0, 1, 2	**Music, Talk**

***Reminder Types: 0 =None; 1 = Visual; 2 = Voice; 3 = Both**

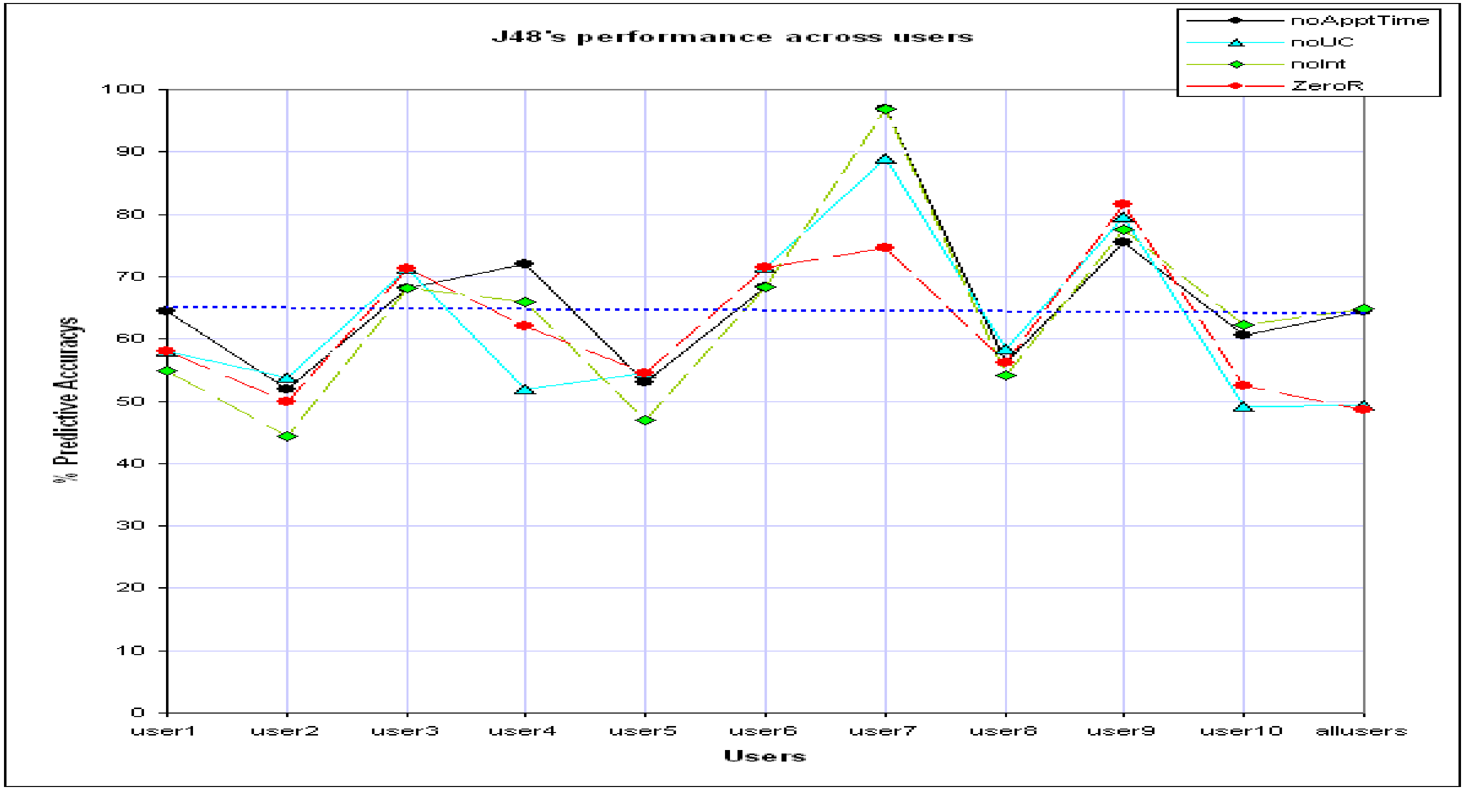
New Results
Two-Class Problem



Average Performance

Better than chance results (50)
Six users (1,3,4,5,6,9) better than average performance
Four users (2,7,8,10) below average performance

New Results
Four-class Problem



Better than chance results (50)
Six users (1,3,4,6,7,9) better than average performance
Four users (2,5,8,10) below average performance

Short-term study, 10 users…contd

Prediction accuracy on the four-class problem

Learning Algorithm	Original Data	No User Context	No External Context
Zero-R	48.62	48.62	48.62
One-R	63.23	48.62	63.23
J 48	62.71	50.00	62.54
XCS	88.35	3126	62.51

Removing user-context degrade the performance of One-R, J48 and XCS
Removing External user-context degrades the performance of J48 and XCS

Short-term study, 10 users…contd

Prediction accuracy on the two-class problem

Learning Algorithm	Original Data	No User Context	No External Context
Zero-R	85.56	85.56	85.56
One-R	86.42	86.42	86.42
J 48	87.11	86.42	86.59
XCS	85.91	71.13	74.39

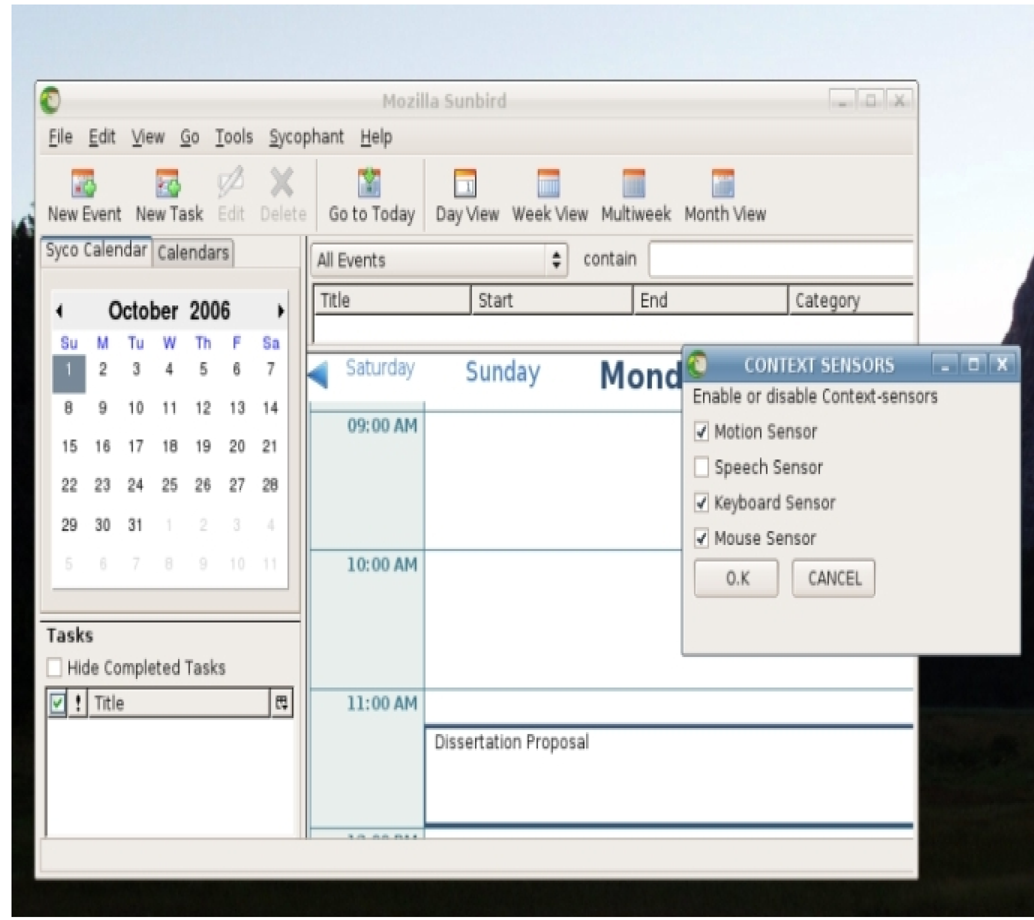
Removing external user-context or user-context degrade the performance of J48 and XCS

New Results Summary

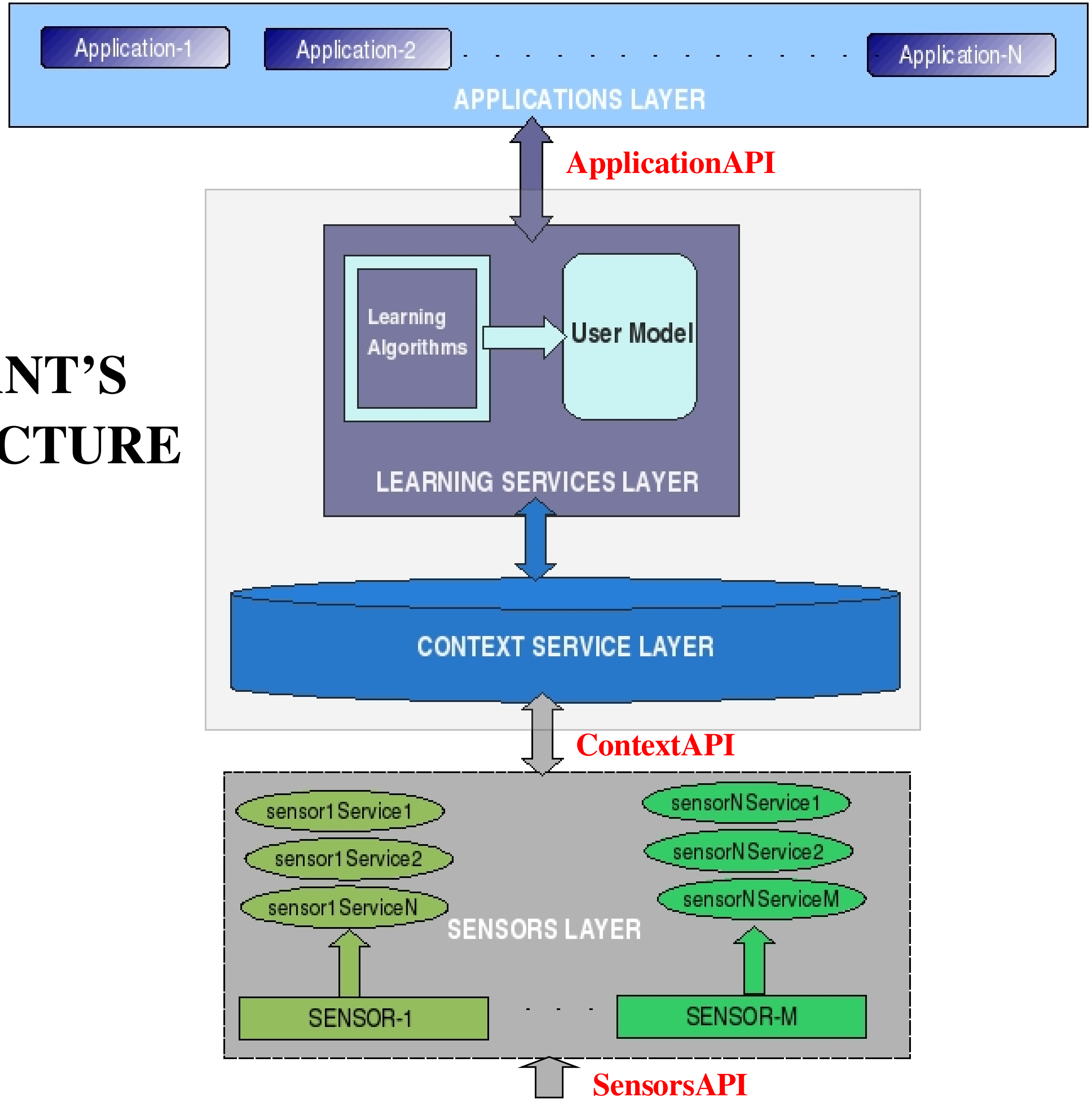
- Learn preferences for individual users
- Predictive accuracy for reminder-types better than chance
- Learning preferences (four-classes) is harder than learning to interrupt (two-classes)
- User-context important for learning user-preferences
- Below average performance for four users: user2,user7 user8, user10
- Classifier System (XCS) is a promising approach for learning user preferences

Proposed Work

- Incorporate Sunbird (a calendaring application) into the Sycophant framework
- Collect short-term usage data
- Collect long-term usage data
- Experiment with other machine learning approaches
- Enable one more desktop application (xmms) to use our Sycophant framework

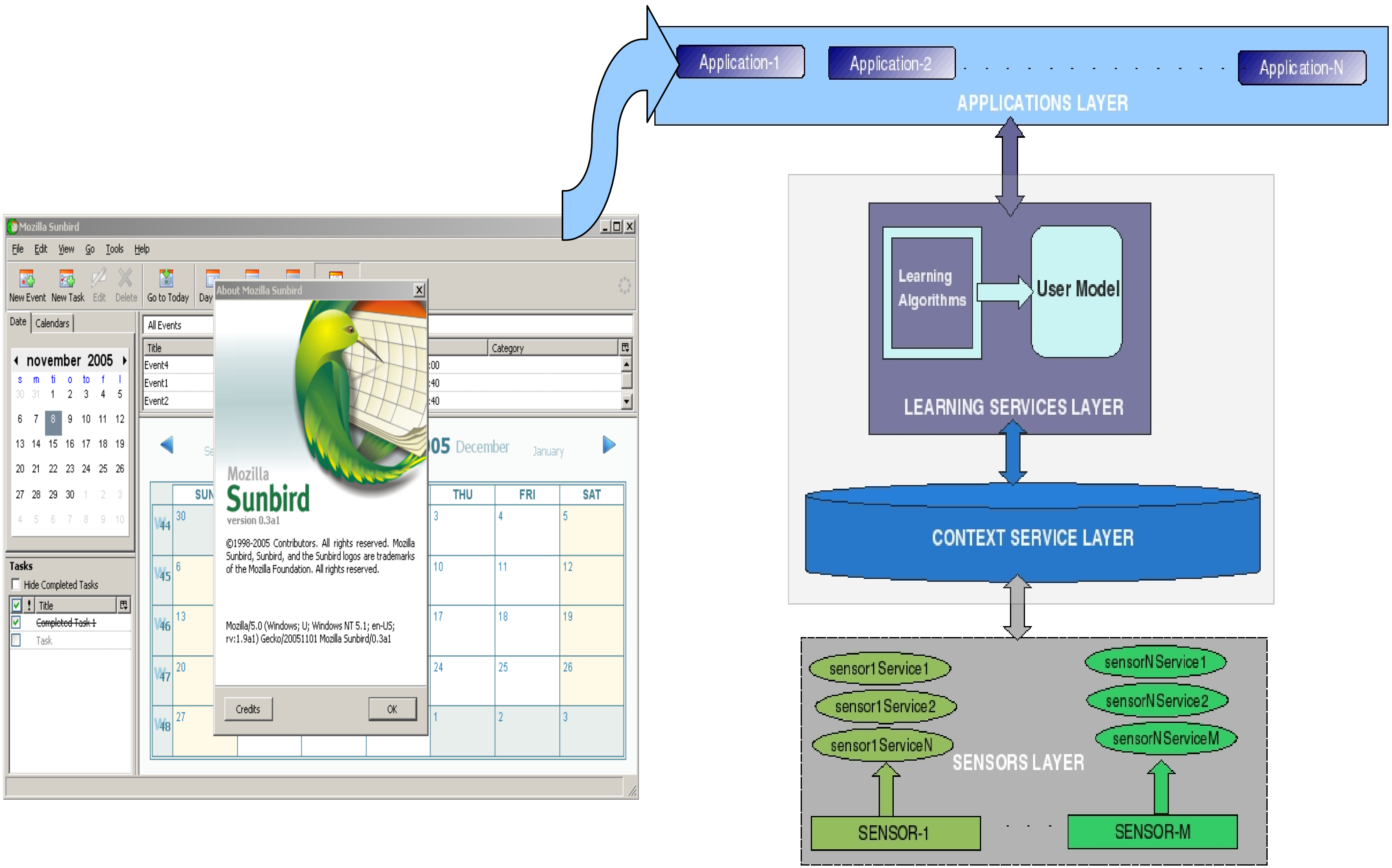


SYCOPHANT'S ARCHITECTURE



Anil Shankar

User-Context Aware Calendaring



Application Programming Interface (API)

1. Set up SENSORS

```
ms = Sensors('Motion Sensor')
ms.setLogFile('MotionLog.txt')
ms.setTickValue(15) // log data every 15 seconds into a sensor file
ms.addSensorService('Any15')
ms.logSensorData('Any15')
```

2. Set up FEATURE EXTRACTOR

```
fe = FeatureExtractor()
sfDict = {ms: 'Any15'}
UserContextData = fe.getUserContextData(sfDict)
```

API in action contd…

3. Enable a Calendar application to use the API

```
appapi = ApplicationAPI()
appapi.setAppParamToPredict('reminderType')
String[] appParams = {'user-id', 'appointment time', 'appointment text'}
appapi.setAppUserContextFeatures(appParams)
appParamVals = appapi.getAppUserContextData()
learner = 'xcs'
currUserModel = appapi.getUserModel(learner,UserContextData, appParamVals)
predictedReminder = appapi.predictParameter(currUserModel)
appapi.updateUserModel(currUserModel)
```

User-Data Collection

- Design user-studies
- Collect short-term usage data
- Long-term data collection
- Disperse the software; collect data from a diverse set of users

Milestones

- Fall 2006
  - Context-enable Sunbird, distribute the user-context aware software bundle, start short-term data gathering
- Spring 2007
  - Gather and analyze preliminary results. Submit to conferences and a journal, start long-term data gathering
- Summer 2007
  - Refactor and refine the framework, experiment with additional machine learning approaches, Integrate XMMS – media player with the framework
- Fall 2007
  - Distribute the framework through the Open-Source community, Gather short-term usage data from XMMS, start long-term data collection
- Spring 2008
  - Publish results, Write proposal, defend Ph.D.

Proposed Enhancements

Learning Services Layer:

- Incorporate additional machine learning algorithms - Support Vector Machines and reinforcement learners

Applications Layer:

- Test the framework with at least one more application, say xmms (open-source music player for Linux)

Intended Contributions

HCI - Application Personalization

- A novel machine learning framework to enable applications learn user preferences based on user-context gathered from internal and external sensors
- Personalization to individual users
- Personalization across groups of users

Intended Contributions

Computer Security

- learning user patterns to enhance current intrusion detection methodologies

Robotics:

- stationary robot model of a desktop-PC
- provide more insights into human-robot interactions (social robotics)

Publications

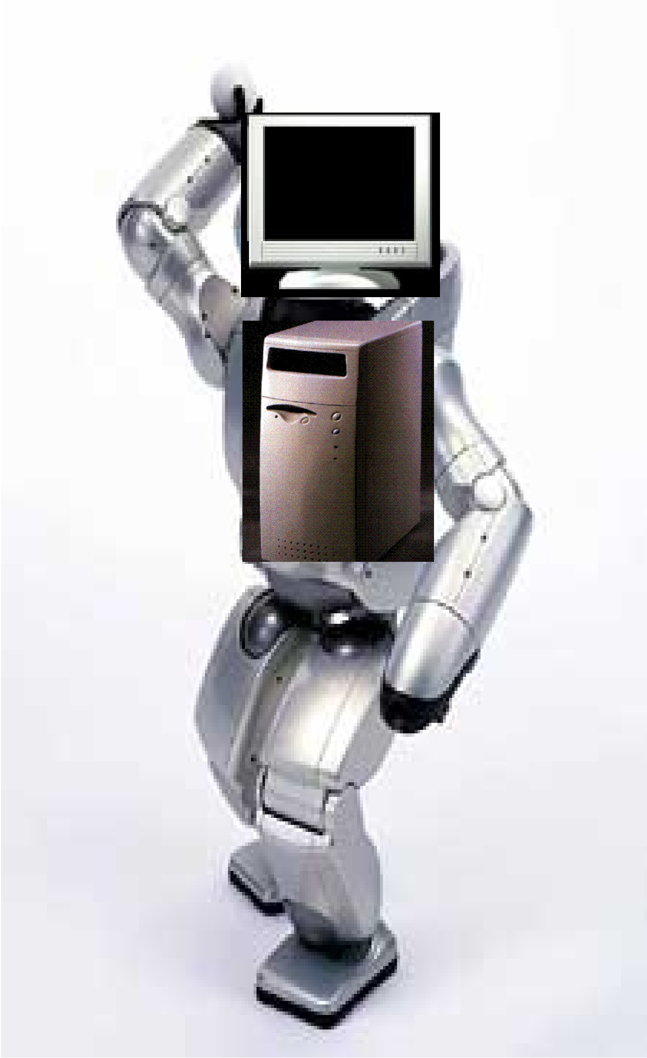
1. Sushil J. Louis and Anil Shankar.
   Context learning can improve user interaction.
   In Proceedings of the 2004 IEEE International Conference on Information Reuse and Integration, IRI - 2004, November 8-10, 2004, las Vegas Hilton, Las Vegas, NV USA , pages 115-120, 2004.

2. Anil Shankar and Sushil J. Louis.
   Learning classifier systems for user context learning.
   In 2005 IEEE Congress on Evolutionary Computation,September 2-5 2005, Edinburgh, UK , 2005.

3. Anil Shankar and Sushil J. Louis.
   Better personalization using learning classifier systems.
   In Proceedings of the 2005 Indian International Conference on Artificial Intelligence, December 20-22 2005, Poona, INDIA , 2005.

4. A. Shankar. *Simple user-context for better application personalization*. Master's thesis, University of Nevada, Reno, NV.,2006.

Comments/Questions?



Sycophant's Application Programming Interface (API)

SENSORS API

Description: Manages sensors, sensor services and sensor data

Class Sensor:

Sensor(String SensorName):

Description - Creates a new sensor with SensorName

Returns - boolean indicating the status of the operation

def setLogFile(File SensorLogFile):

Description: Associates a log file for a sensor

Returns: boolean indicating whether a file was successfully created

def delete():

Description: Deletes the sensor

Returns: boolean indicating the status of the operation

def setTickValue(int tickSeconds):

Description: Sets the time interval during which sensor data is repeatedly logged to the sensor file

Returns: a boolean status of the operation

SENSORS API contd…

def addSensorService(String SensorServiceName):

Description: Adds a new SensorService object to the sensor

Returns: a boolean indicating the status of the operation

def deleteSensorService(SensorService service):

Description: deletes a sensor service type associated with the sensor

Returns: a boolean indicating the status of the operation

def logSensorData(String serviceName):

Description: logs the service data value specified by serviceName to the sensor log file

Returns: a boolean indicating the status of the operation

def getSensorData(SensorService service):

Description: fetches the sensor services data

Returns: a column of data formatted according to the specified service

SENSORSERVICES API

Description: Creates and manages a sensor's service and its data

def SensorService(String serviceFormat):

Description: Creates a sensor service with the data formatting type specified by serviceFormat

Returns: boolean status of the operation

def getServiceData(String serviceFormat):

Description:Accesses the sensor's log file and formats the data specified by the serviceFormat

Returns: a column of sensor data formatted according to the specified serviceFormat

FEATURE EXTRACTOR API

Description: extracts sensor service features for use by the Application API

def getUserContextData(HashMap SensorServicesDict):

- **Description: Gets user-context data columns for the sensors and their services specified in the SensorServicesDict**
- **Returns: Columns of user-context data with formatted features of sensor services**

APPLICATION API

Description: Provides user-context features access for a machine learner

def setAppParamToPredict(String paramName):

Description:Sets which application parameter needs to be predicted

Returns: a boolean indicating the status of the operation

def setAppUserContextFeatures(String[] appParamNames):

Description:Sets which application specific features need to be logged

Returns: a boolean indicating the status of the operation

def getAppUserContextData():

Description:Provides current values of the application-specific user-context features

Returns: String containing the values of the application-specific user-context features

APPLICATION API contd…

def getUserModel(String learner, HashMap SensorServicesDict, String[] appFeatures):

Description: Executes a learner and returns a user model built using the application-specific user-context features and the user-context features chosen from sensor-service dictionary

Returns: a user-model data structure

def updateUserModel(UserModel userModelName):

Description:Updates an existing user model

Returns: a boolean indicating the status of the operation

def predictParameter(UserModel um):

Description: uses the current user model to predict the application's appParamToPredict

Returns: predicted value of the application parameter to be predicted

Results from short-term study

DataSet	ZeroR	OneR	OneR_NoExt
user1	0.5806	0.6290 (motionCount5)	0.5645 (mouseCount5)
user2	0.5000	0.6111 (mouseCount5)	0.5925
user3	0.7121	0.6515 (no preferences!, 1 def)	0.7121
user4	0.6200	0.56 (mouseCount5)	0.6200
user5	0.5454	0.5303 (motionCount5)	0.5303
user6	0.7142	0.6824	0.7142 (kbdCount)
user7	0.7460	0.9523 (motionAny1)	0.9206
user8	0.5625	0.5414	0.5625
user9	0.8163	0.7751 (keybdCount5)	0.8163
user10	0.5245	0.5245 (motionCount5)	0.4262
allUsers	0.4862	0.6323 (user-id)	0.6323

Rules learned for different users

User	Reminder types predicted by OneR
user1	Visual and Voice Reminder
user2	Visual and Voice Reminder
user3	Visual Reminder
user4	Visual and Voice Reminder
user5	Voice Reminder
user6	Visual Reminder
user7	Visual Reminder
user8	Visual and Voice Reminder
user9	Visual Reminder
user10	Visual Reminder

Intended Contributions - 2

Learning Cognitive User Models

- Psychological theories of behavior to learn cognitive user models
- Psychological theories and computational cognitive models can mutually inform each other
- Designing user-studies to incorporate human factors and human psychology in our computationally generated user-models