JIIK 2009, Volume 1, Issue 3
Editor: Rafiq Islam, pp.

ISSN 1937-7983

© 2008 Nova Science Publishers, Inc.

INFRASTRUCTURE OF A CONTEXT ADAPTIVE AND PERVASIVE MULTIMODAL MULTIMEDIA COMPUTING SYSTEM

Manolo Dulva Hina*[*,1,2], Amar Ramdane-Cherif[3], Chakib Tadj[4] and Nicole Levy[5]

[1]Université du Québec, École de technologie supérieure, Montréal, Canada
[2]PRISM Laboratory, Université de Versailles-Saint-Quentin-en-Yvelines, France
[3]PRISM Laboratory, Université de Versailles-Saint-Quentin-en-Yvelines,
45 avenue des États-Unis, 78035 Versailles Cedex, France
[4]Université du Québec, École de technologie supérieure,
1100, rue Notre-Dame Ouest, Montréal, Québec H3C 1K3 Canada
[5]PRISM Laboratory, Université de Versailles-Saint-Quentin-en-Yvelines,
45 avenue des États-Unis, 78035 Versailles Cedex, France

ABSTRACT

The aim of pervasive multimodal multimedia computing is to realize anytime, anywhere computing using various modes of human-computer interaction. The current state-of-the-art pervasive systems and solutions, however, do not include applications that are related to pervasive multimodality. Also, the current multimodal interfaces were designed with pre-defined modes of human-machine interaction that were not chosen based on the given context of the user, of his environment and of his computing system. This paper addresses these weaknesses by proposing a pervasive multimodal multimedia computing system in which its modalities are chosen based on their suitability to the given interaction context. This same system chooses a multimodal (or unimodal) interface based on the given context, available media devices and user preferences. This paper discusses the challenges in designing the infrastructure of such computing system and illustrates how we addressed those challenges. This work is our contribution to the ongoing research that aims at realizing pervasive multimodality.

Keywords: multimodal multimedia computing; pervasive computing; human-computer interaction; context-aware system.

[*] Corresponding author: E-mail: manolo-dulva.hina.1@ens.etsmtl.ca

1. INTRODUCTION

Pervasive computing (also known as *ubiquitous computing*) aims at providing anytime, anywhere computing to a user working on a computing task using various software applications. This has been made possible because the infrastructure of pervasive computing (Satyanarayanan 2001) (Pahlavan and Krishnamurthy 2002), (Satyanarayanan 1990; Satyanarayanan 1996) does exist, an infrastructure that allows both wired and wireless computing and communications. *Pervasive multimodal multimedia computing*, on the other hand, aims to provide the infrastructure that would realize anytime, anywhere computing using various modes of human-computer interaction. Context awareness is an integral characteristic of pervasive computing systems. Context-awareness implies that the system is capable of adapting its operations to the most current context without explicit user intervention. Some context-aware systems have been developed to deliver pervasive healthcare, education, and communication, just to cite a few. Noticeably, however, there is something missing in the current state-of-the-art pervasive applications – one that would permit pervasive multimodal multimedia computing.

A *multimodal system*, in the context of human-computer interface, refers to the fusion of two (or more) input modes – such as speech, pen, gesture, gaze and head and body movements (Oviatt 2002). In contrast, a *unimodal recognition system* or *interface* involves only a single recognition-based technology, such as speech, pen and vision. Some of the current multimodal interfaces do fusion of speech and pen inputs, speech and lip movements, speech and manual gesturing, and gaze tracking and manual input. This implies, however, that the modes of human-machine interaction are already pre-defined from the very beginning. In most of these interfaces, there is an assumption that the setting is ideal (i.e. that there is barely a change in environment's context) and that the user is stationary. For mobile computing, a new conflicting requirement arises – that is, mobility requires that the computing terminal be light and small yet the system is required to deliver more advanced multimedia features. Such requirement suggests that keypads should possibly shrink or even vanish. To this end, the suitability of manual input modality also shrinks while the others – specifically, the vocal input modality – augments. Hence, the necessity for a wireless user interface (Ringland and Scahill 2003) in which speech is the mode for data input interaction.

Most of the multimodal interfaces are not suitable to mobile users. Most of their modes for data input are pre-defined from the beginning and not selected based on their suitability to the environment's context. The drawback of such non-adaptive system is that if a context parameter changes (e.g. the environment becomes noisy) then the effectiveness of the interface is compromised (e.g. speech, as mode for data input, is not effective in a noisy workplace). To this end, we believe that an ideal pervasive system for multimodal application must have a wide-range selection of multimodal interfaces (aside from the regular unimodal interfaces) and at any given time, one particular interface is selected based on its suitability to the context of the user, of his environment and of computing system (henceforth called the *interaction context*).

Modality, in the context of multimodal multimedia computing, refers to the mode of human-computer interaction for data input and output. Given the current state-of-the-art systems and solutions, we have noted that the infrastructure to realize pervasive multimodality is missing. Such infrastructure is important as it is meant to be the backbone that (i) *implements either stationary or mobile computing*, (ii) *allows the invocation of modalities based on context suitability and availability of supporting media devices*, and (iii) *appropriately selects a unimodal or multimodal interface based on the given interaction context*. This paper, therefore, is intended to present the design of such infrastructure, the challenges involved in the design and our proposed solutions to address these challenges.

Apart from this introductory section, the rest of this paper is structured as follows. Section 2 discusses the works related to pervasive computing, multimodal system and the shortcomings of the current state-of-the-art systems and solutions, and presents our idea of addressing them. In section 3, we list down software engineering challenges related to this work and how we address them, and in the process presents our contribution. In section 4, we explain the concepts of context and its relationship with modality and media devices. In section 5, we explain our proposed method for selecting appropriate modalities for the user interaction interface. Sample cases are cited in section 6 and the architectural framework of our proposed system is explained in section 7. Finally, we conclude this paper in section 8 and provide future works that we intend to do.

2. Related Work

Pervasive computing (aka ubiquitous computing) (Weiser 1991; Vasilakos and Pedrycz 2006) realizes anytime, anywhere computing; In doing so, a user's productivity increases as he can continue working on an interrupted computing task whenever and wherever he wishes. Context awareness, along with context management, heterogeneity, scalability, mobility, transparent user interaction, dependability and security are some software infrastructural issues for ubiquitous computing (Costa, Yamin et al. 2008). Several applications of pervasive computing have been developed and implemented, among them are one for pervasive *healthcare* (Varshney 2003), *education* (Garlan, Siewiorek et al. 2002) and *communication* (Vallejos, Desmet et al.), just to cite a few. Missing, however, in the current state-of-the-art pervasive applications is the one that is related to multimodal multimedia computing.

In human-computer interface, multimodality refers to the fusion of two (or more) modes for data input. Since Bolt's original "*Put that there*" concept demonstration (Bouhuys 1995), which processed speech and manual pointing during object manipulation, some significant achievements in multimodal interface have surfaced, such as the one that combines *speech and pen* (Oviatt 2000), *speech and gestures* (Oviatt and Cohen 2000), *speech and lips movements* (Rubin, Vatikiotis-Bateson et al. 1998), *gaze and speech* (Zhang, Imamiya et al. 2004), *speech and mouse* (Djenidi, Ramdane-Cherif et al. 2002), *interface for Internet* (Dong, Xiao et al. 2000), and *wireless user interface* (Ringland and Scahill 2003). But *why build multimodal interface*? It is because it supports more transparent, flexible, efficient and expressive means of human-computer interaction. Multimodal interfaces are expected to be easier to learn and use, and are expected to accommodate more adverse user conditions than in the past (Oviatt 2002). The drawback to the current state-of-the-art multimodal interfaces, however, is that they are all designed with pre-defined modes for data input and without consideration to the varying conditions in the user's workplace, such as a workplace that becomes noisy. Most of these existing systems are also meant for users who are in stationary locations, and hence would become ineffective the moment the user becomes mobile.

Given the limitations cited above, we then envision a pervasive multimodal multimedia computing system. This new computing paradigm's infrastructure is characterized by the following features: (1) it is adaptive to the given interaction context – that is, the modalities for data input and output between the user and the machine are chosen based on their suitability to the given context, (2) that the modalities of interaction are chosen because they can be supported by available media devices, (3) that the chosen user interface is selected based on its suitability to the given context, the availability of supporting media devices and of user's preferences, (4) that the infrastructure supports both stationary and mobile computing, and (5) that the infrastructure itself is autonomic – specifically, it is self-optimizing, self-adaptive, self-configurable, self-optimizing (Horn 2001; Kephart and Chess 2001; Salehie and Tahvildari 2005). Due to space constraints, however, the design of the system's infrastructure as presented in this paper demonstrates only its *self-adaptive* features.

This work is our contribution to the ongoing research in making anytime, anywhere computing using the most suitable form of human-computer interaction possible.

3. Requirements Analysis and Contribution

Here, we list down some software engineering challenges by posing specific technical challenges that need to be addressed. By answering these challenges, we do explain our novel contribution to the software engineering domain.

Our goal is to model a pervasive multimodal multimedia computing system. The design of such a system needs to address some key requirements cited below:

Requirement 1: Provide a generic representation of context. Provide a methodology that allows the incremental definition of context (i.e. add, delete, modify a context parameter).

Requirement 2: Provide the relationship between modality and context. Given that the application domain is multimodality, what parameters constitute the user, environment and system contexts? On what basis a specific modality is considered suitable to a context parameter and to the overall interaction context?

Requirement 3: Given a modality that is suitable to the given interaction context, provide a mechanism that chooses its supporting media devices. Then, given the modality and media devices selections, provide the mechanism that will determine the appropriate (unimodal or multimodal) user interface. What factors should be considered in the selection of a user interface?

The technical challenges are addressed by our proposed solutions given below:

Proposed solution to requirement 1: The term context, in this work, refers to *interaction context* (*IC*) which is the combined contexts of the user, his environment and his computing system. We provide a mathematical model that defines each parameter of interaction context and a virtual machine model for its implementation.

Proposed solution to requirement 2: The modalities for human-machine interaction are *manual*, *visual* and *vocal*, both to input and output data (details in next section). All context parameters are related to its domain of application, which in this case is multimodality. The relationship to consider is whether a specific modality is suitable to each *IC* parameter and if so, to what extent.

Proposed solution to requirement 3: We establish a relationship between modality and media devices. Given that any selected modality is deemed appropriate to the given context, it also follows that the selected media devices supporting the modality are also suitable to the given context. The interface of user-machine interaction is either unimodal or multimodal. When more than one interface is found suitable, then another factor to consider is the user's preference vis-à-vis user interface. Hence, we propose a priority ranking being assigned to the user's preference. The same priority ranking applies to the user's preferred modality and preferred media devices. The selection of user's interface, therefore, is based on appropriately selected modalities, available media devices and user's preferences.

4. Context, Multimodality and Media Devices

Here, we define context and provide its mathematical representation. We also illustrate how we can implement an incremental definition of context. Then, we derive the relationships that exist between context and multimodality and between multimodality and media devices.

4.1. Context Definition and Representation

In chronological order, the early definition of context includes that of (Schilit and Theimer 1994; Oviatt 2002) in which context means the answers to the questions "*Where are you?*", "*With whom are you?*", and "*Which resources are in proximity with you?*". Schilit defined context as the changes in the physical, user and computational environments. This idea is taken afterwards by Pascoe (Pascoe 1998) and later on by Dey (Dey, Salber et al. 1999). Brown considered context as "*the user's location, the identity of the people surrounding the user, as well as the time, the season, the temperature, etc.*" (Brown, Bovey et al. 1997). Ryan defined context as the environment, the identity and location of the user as well as the time involved (Ryan, Pascoe et al. 1997). Ward viewed context as the possible environment states of an application (Ward, Jones et al. 1997). In Pascoe's definition, he added the pertinence of the notion of state: "*Context is a subset of physical and conceptual states having an interest to a particular entity*". Dey specified the notion of an entity: "*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 application themselves*" (Dey 2001). This definition became the basis for Rey and Coutaz to coin the term interaction context: "*Interaction context is a combination of situations. Given a user U engaged in an activity A, then the interaction context at time t is the composition of situations between time t_0 and t in the conduct of A by U*" (Rey and Coutaz 2004).

We adopted the notion of "*interaction context*", but define it in the following manner: An interaction context, IC = {IC_1, IC_2,…, IC_{max}}, is a set of all possible interaction contexts. At any given time, a user has a specific interaction context *i* denoted as IC_i, $1 \leq i \leq max$, which is composed of variables that are present in the conduct of the user's activity. Each variable is a function of the application domain which, in this work, is multimodality. Formally, an IC is a tuple composed of a specific user context (UC), environment context (EC) and system context (SC). An instance of IC is given as:

$$\mathrm{IC_i} = \mathrm{UC_k} \otimes \mathrm{EC_l} \otimes \mathrm{SC_m} \qquad (1)$$

where $1 \leq k \leq max_k$, $1 \leq l \leq max_l$, and $1 \leq m \leq max_m$, and *max_k, max_l and max_m* = maximum number of possible user contexts, environment contexts and system contexts, respectively. The Cartesian product (symbol: ⊗) denotes that IC yields a specific combination of UC, EC and SC at any given time.

The user context UC itself is composed of parameters that describe the state of the user during the conduct of his activity. A specific user context k is a tuple composed of ($\mathrm{ICParam}_{k1}$, $\mathrm{ICParam}_{k2}$, … $\mathrm{ICParam}_{k\,max_k}$) and is given by:

$$\mathrm{UC_k} = \bigotimes_{x=1}^{max_k} \mathrm{ICParam_{kx}} \qquad (2)$$

where $\mathrm{ICParam}_{kx}$ = parameter of UC_k, k = the number of UC parameters, $k \in 1 .. max_k$. Using similar convention as that of UC, a specific instance of environment context EC_l and a specific instance of system context SC_m can be specified as follows:

$$\mathrm{EC_l} = \bigotimes_{y=1}^{max_l} \mathrm{ICParam_{ly}} \qquad (3)$$

$$\mathrm{SC_m} = \bigotimes_{z=1}^{max_m} \mathrm{ICParam_{mz}} \qquad (4)$$

4.2. Incremental Definition of Interaction Context

As stated, an instance of *IC* is composed of specific instances of *UC, EC*, and *SC*, which themselves are composed of one or more parameters. To realize the incremental definition of *IC*, each of these parameters is introduced into the system, one at a time.

In our work, a virtual machine is designed to add, modify or delete one context parameter at a time, making *IC* parameters a reflection of the system's dynamic needs. A *virtual machine* (VM) is software that creates a virtualized environment on computer platform so that the end user can operate the software. *Virtualization* is the process of presenting a group or subset of computing resources so that they can be accessed collectively in a more beneficially manner than their original configuration. In effect, a VM is an *abstract computer*; it accepts input, has algorithms and steps to solve the problem related to the input, and yields an output. The steps taken by the VM are its "*instructions set*" which is a collection of functions that the machine is capable of undertaking. A *layered VM* is a group of VM's wherein interaction takes place only between adjacent layers. *Layering* is a design choice to limit the propagation of errors within the concerned layer only during a modification of its functionality. Generally, in layered VM, the top layer refers to the interface that interacts with the end users while the bottom layer interacts with the hardware. Hence, Layer 0 is the bottom layer composed of sensors that generate some raw data representing the value needed by the topmost VM layer.

Figure 1 shows the functionality of such "*machine*". In general, the transfer of instruction command is top-down (steps 1 to 4). At Layer 0, the raw data corresponding to the *IC* parameters are collected for sampling purposes. The sampled data are then collated and interpreted, and the interpretation is forwarded to different layers bottom-up (steps 5 to 8).

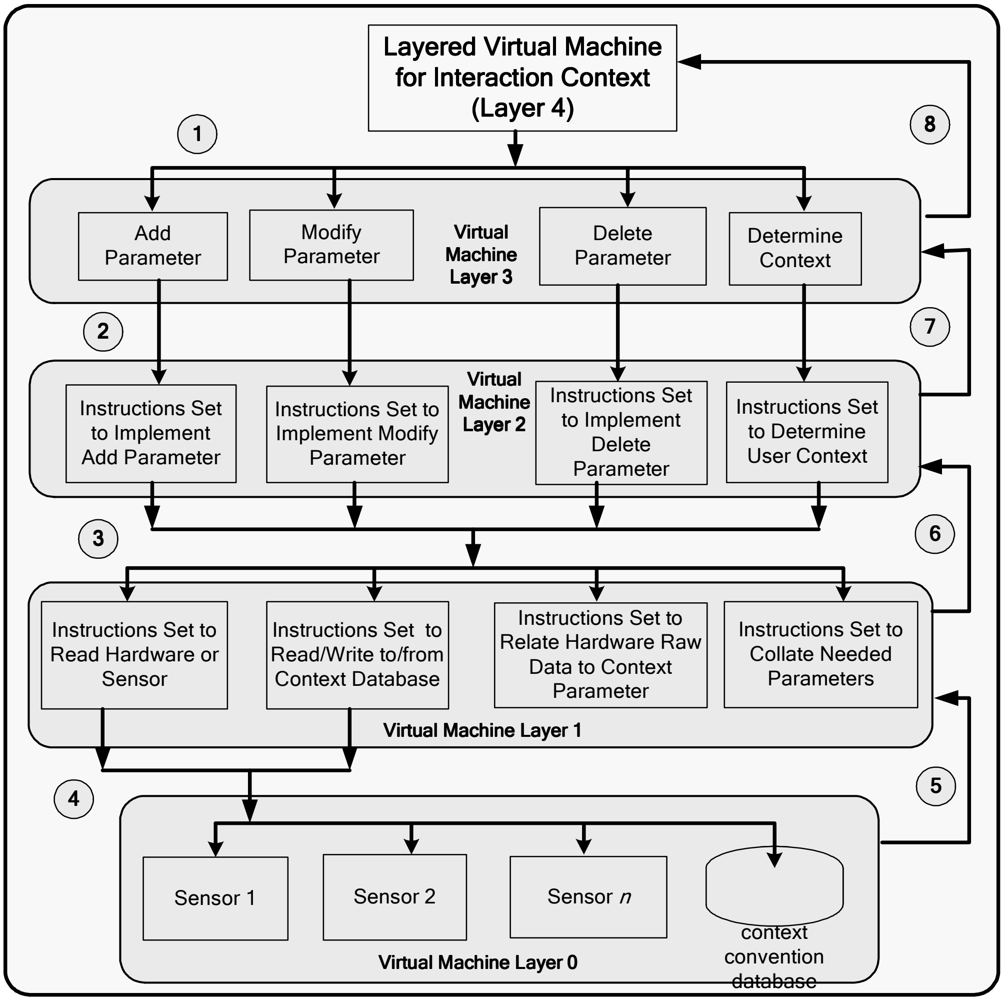
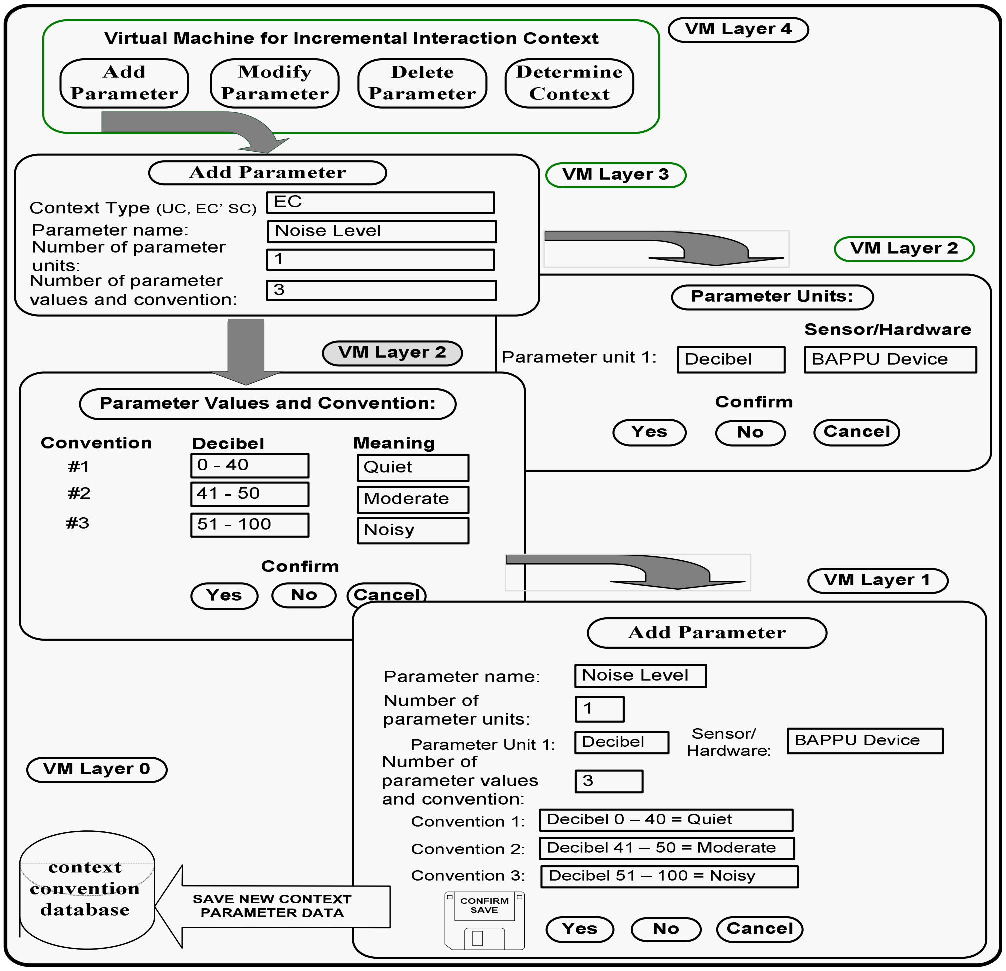

Layered Virtual Machine for Interaction Context (Layer 4)
1
8
Add Parameter
Modify Parameter
Virtual Machine Layer 3
Delete Parameter
Determine Context
2
7
Instructions Set to Implement Add Parameter
Instructions Set to Implement Modify Parameter
Virtual Machine Layer 2
Instructions Set to Implement Delete Parameter
Instructions Set to Determine User Context
3
6
Instructions Set to Read Hardware or Sensor
Instructions Set to Read/Write to/from Context Database
Instructions Set to Relate Hardware Raw Data to Context Parameter
Instructions Set to Collate Needed Parameters
Virtual Machine Layer 1
4
5
Sensor 1
Sensor 2
Sensor n
context convention database
Virtual Machine Layer 0

Figure 1. The design of a layered virtual machine for incremental interaction context.

The VM Layer 4 acts as the human-machine interface; its "*instruction set*" are the four functions found in Layer 3. The "*add parameter*", "*modify parameter*", and "*delete parameter*" are basic commands that manipulate the sensor-based context parameters while "*determine context*" yields the values of currently-defined parameters. VM Layer 2 is a "*library of functions*" that collectively supports Layer 3 instructions while Layer 1 is another "*library of functions*" that acts as a link between Layer 2 and Layer 0.

4.2.1. Adding a Context Parameter

Consider using VM to add a specimen context parameter: the "*noise level*". See the design of VM's user interface in Figure 2.



Context Type (UC, EC' SC)	EC
Parameter name:	Noise Level
Number of parameter units:	1
Number of parameter values and convention:	3

	Sensor/Hardware
Parameter unit 1: Decibel	BAPPU Device

Convention	Decibel	Meaning
#1	0 - 40	Quiet
#2	41 - 50	Moderate
#3	51 - 100	Noisy

Parameter name:	Noise Level
Number of parameter units:	1
Parameter Unit 1:	Decibel
Sensor/ Hardware:	BAPPU Device
Number of parameter values and convention:	3
Convention 1:	Decibel 0 – 40 = Quiet
Convention 2:	Decibel 41 – 50 = Moderate
Convention 3:	Decibel 51 – 100 = Noisy

Figure 2. The interactions among layers to add a new (specimen only) context parameter: "Noise Level".

As shown in the diagram, upon invoking the VM user interface (i.e. Layer 4), the user chooses the "*Add Parameter*" menu. A window opens up which transfers the execution control to Layer 3. Then data entry takes place. To realize adding a new context parameter, at least four data entry functions must exist, namely: (i) *getting context type of the parameter*, (ii) *getting name of the parameter*, (iii) *getting the parameter's number of units*, and (iv) *getting number of parameter values and conventions*. In Layer 3, the user inputs "*Noise level*" as parameter name, itself an EC parameter, "*1*" as parameter unit, and "*3*" as parameter values and conventions. When done, two new windows open up, one window at a time, that brings up the functionalities of Layer 2. For each parameter's unit, the VM receives input for the parameter's unit name and the sensor (or hardware) that supplies its raw data. As shown, the unit for “Noise level” is specified as “*decibel*” and the BAPPU noise measuring device (http://www.bappu.com/) (or any sensor for that matter that measures noise and supplies data to the computer) as the sensor supplying the data. When done, another Layer 2 window opens up for data entry of “*Parameter values and conventions*”. In the diagram, the user specifies the value (range of decibels) that he considered is equivalent to “*quiet*”, “*moderate*” and “*noisy*”. When done, a window for Layer 1 opens up to save the newly-added parameter information. This function interacts directly with the hardware (i.e. the context convention database).

4.2.2. Modifying and Deleting a Context Parameter

The VM layers interaction involved in “*Modify parameter*” is almost identical to that of “*Delete Parameter*” function. The only thing extra in the former is a procedure that allows user to select the context parameter that should be modified. Other than that, everything else is the same. The processes involved in “*Delete Parameter*” menu are shown in Figure 3.

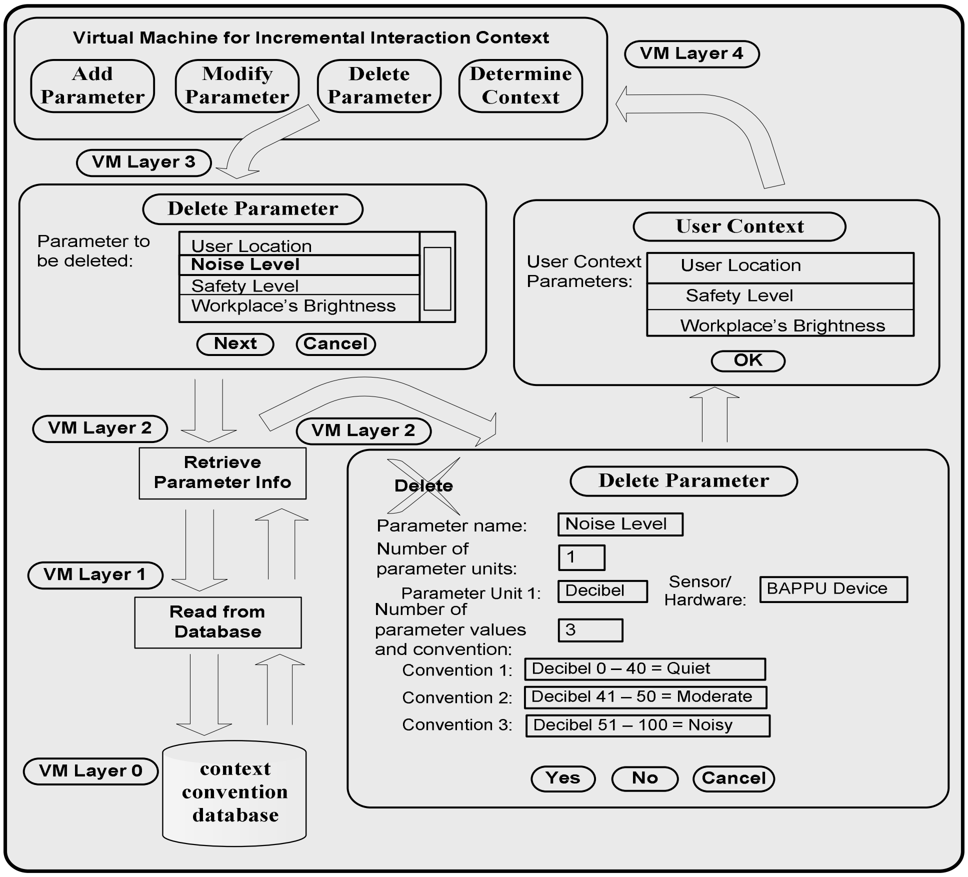


Figure 3. The VM layers interaction to realize “deleting a user context parameter”.

Upon menu selection, the execution control goes to Layer 3, demanding the user to specify the parameter for deletion (e.g. “*Noise level*” is chosen for deletion). Upon confirmation, the information about the parameter for deletion is extracted and read from database (transfer of control from Layer 2 to Layer 1 then to Layer 0). When the information for deletion is read, the control goes back to Layer 2 where such information is presented and a re-confirmation of its deletion is required. When parameter deletion is done, the control goes back to Layer 3 which presents the updated list of context parameters. An "*OK*" button click transfers the control back to Layer 4.

4.2.3. Capturing the User's Current Context

The interactions of VM layers to "*Determine Context*" are shown in Figure 4. This is simulated using three specimen context parameters, namely (i) *the user location*, (ii) *the safety level*, and (iii) *the workplace's brightness*. When the user opts for this menu, the VM execution control goes to Layer 3. The function "*get user context*" creates threads equal to the number of parameters. Hence, this process produces thread "*get parameter 1*", assigned to detect user location, thread "*get parameter 2*" assigned to get the user's safety level, and the thread "*get parameter 3*" for the user's workplace's brightness (i.e. light intensity). The concepts involved are identical for each thread. Consider the case of "*user location*". The thread passes control to Layer 1 where a function takes sample data from a sensor (e.g. global positioning system, GPS, at http://www.rayming.com), attached to the user computer's USB port or value transmitted to the computer via wireless communication. In the VM design, the user can specify the number of raw data that need to be sampled and in what frequency (*n* samples per *m* unit of time). These *n* samples are then collated, normalized and interpreted.

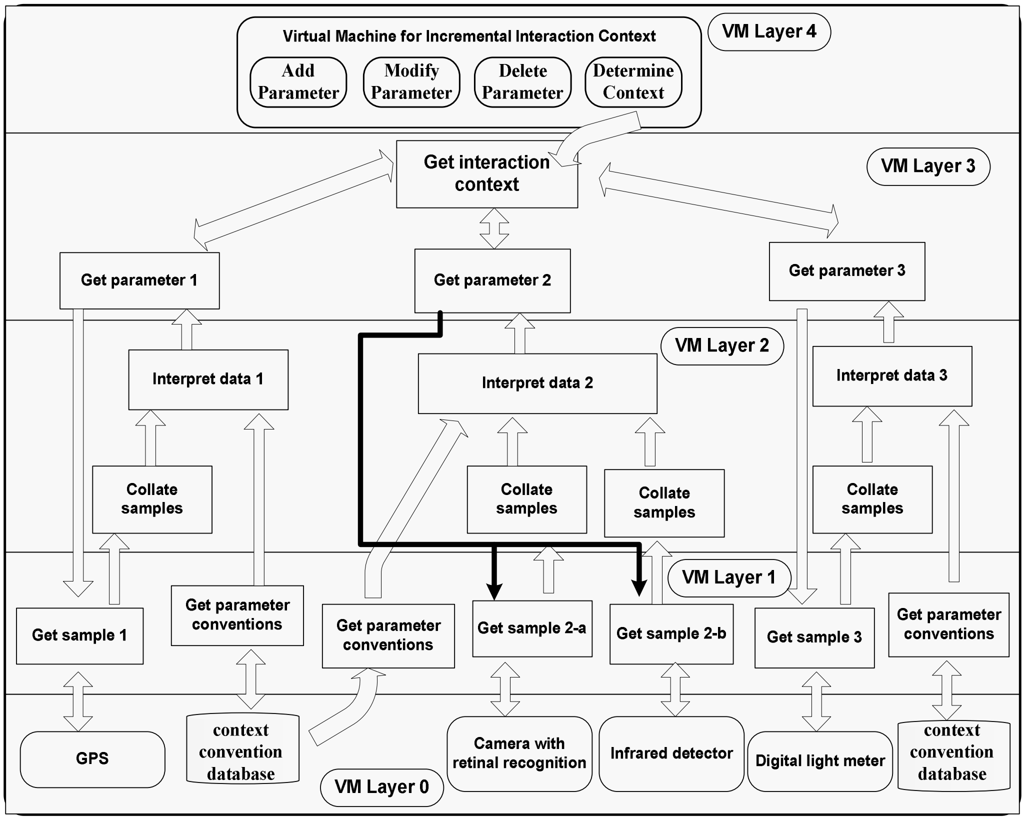
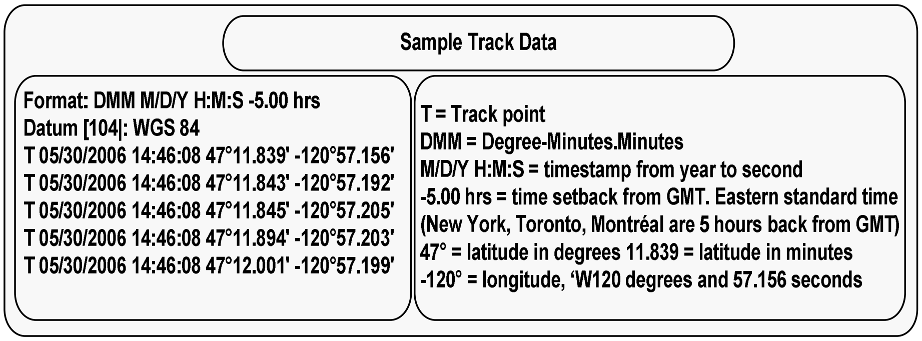


Figure 4. VM layers interaction in detecting the current interaction context.

For example, a specimen GPS data of 5 samples, taken 1 sample per minute, is shown in Figure 5. The data are then normalized (averaged), hence, the user's computer is located at 14°11' latitude and -120°57' longitude. Then, this value is interpreted using the convention values for user location parameter. Table 1 shows the format of the convention values of the specimen parameters. (Recall that the convention value of a parameter is created during the "*Add Parameter*" process.) Using Table 1-a, the interpretation identifies if the user (who uses the computer equipped with a GPS) is at home, at work or on the go.



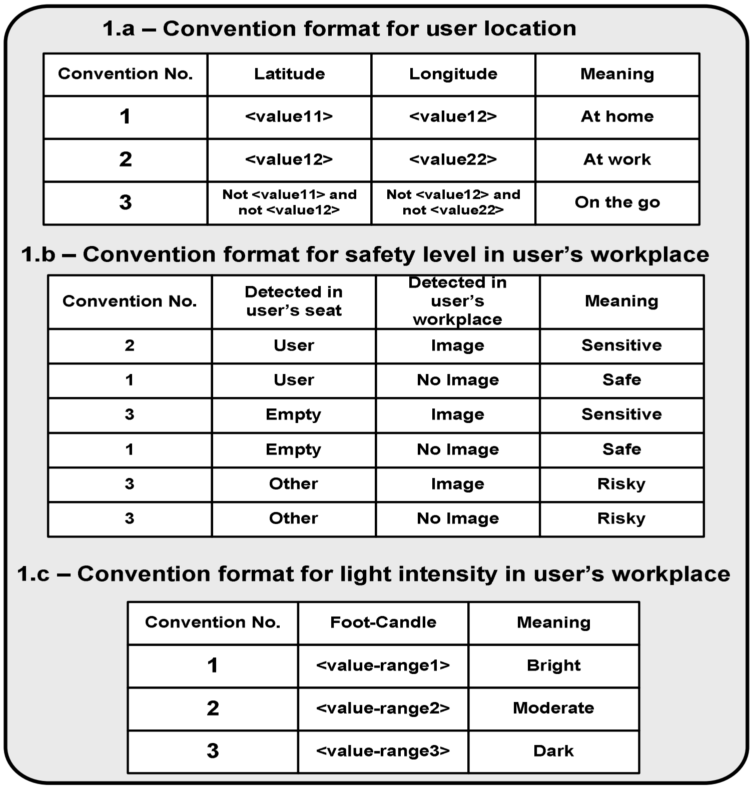
Sample Track Data

Format: DMM M/D/Y H:M:S -5.00 hrs
Datum [104|: WGS 84
T 05/30/2006 14:46:08 47°11.839' -120°57.156'
T 05/30/2006 14:46:08 47°11.843' -120°57.192'
T 05/30/2006 14:46:08 47°11.845' -120°57.205'
T 05/30/2006 14:46:08 47°11.894' -120°57.203'
T 05/30/2006 14:46:08 47°12.001' -120°57.199'

T = Track point
DMM = Degree-Minutes.Minutes
M/D/Y H:M:S = timestamp from year to second
-5.00 hrs = time setback from GMT. Eastern standard time (New York, Toronto, Montréal are 5 hours back from GMT)
47° = latitude in degrees 11.839 = latitude in minutes
-120° = longitude, 'W120 degrees and 57.156 seconds

Figure 5. Sample GPS data gathered from Garmin GPSIII+.

Table 1. Sample conventions of the specimen sensor-based context parameters



1.a – Convention format for user location

Convention No.	Latitude	Longitude	Meaning
1	<value11>	<value12>	At home
2	<value12>	<value22>	At work
3	Not <value11> and not <value12>	Not <value12> and not <value22>	On the go

1.b – Convention format for safety level in user's workplace

Convention No.	Detected in user's seat	Detected in user's workplace	Meaning
2	User	Image	Sensitive
1	User	No Image	Safe
3	Empty	Image	Sensitive
1	Empty	No Image	Safe
3	Other	Image	Risky
3	Other	No Image	Risky

1.c – Convention format for light intensity in user's workplace

Convention No.	Foot-Candle	Meaning
1	<value-range1>	Bright
2	<value-range2>	Moderate
3	<value-range3>	Dark

Specimen parameter 2 *(the workplace's safety level*) is a function of (i) the person sitting in front of the computer, and (ii) the presence of other people in the user's workplace. A camera with retinal recognition (http://www.informatik. uniaugsburg.de/~kimjongh/biometrics/retinal.pdf) may be used to identify the person sitting in the user's seat. The identification process would yield three values: (1) *User* – if the legitimate user is detected, (2) *Other* – if another person is detected, and (3) *Empty* – if no one is detected. Also, an infrared detector (http://www.globalsources.com/manufacturers /InfraredDetector.html) may be used to identify the presence of other person in front or in either side of the user. The identification process would yield two values: (1) *Image* – if at least one person is detected, and (2) *No Image* – if nobody is detected. (Note that the image and pattern recognition is not the subject of this work; hence, the detection process is not elucidated further in this paper.). The VM takes n = 5 samples, normalizes them and compares the result against the convention values in Table 1-b. The interpretation yields a result indicating if user's workplace is *safe*, *sensitive* or *risky*. This specimen parameter is useful for people working on sensitive data (e.g. bank manager) but can be irritating to a person working with teammates (e.g. students working on a project). Hence, this specimen parameter can be added or deleted on the user's discretion.

The third specimen parameter (i.e. *workplace's brightness*) detects the workplace's light intensity. Here, we can assume that a sensor measuring the light's intensity (http://www.gossen-photo.de/english/lichtmess_produkte.html) is attached to the computer's USB port. Its measurement unit, the *foot-candle*, is the number of "*lumens*" falling on a square foot of an inch; lumen is a unit of light used to rate the output of a bulb. For example, we may assume the following conventions in a user's workplace: (i) 0 – 9 foot candles = *dark*, (ii) 10 – 20 foot-candles = *moderate*, and (iii) 21 – 100 foot-candles = *bright*. The processes involved in sampling, collating and interpreting sensor data for parameter 3 is identical with the other 2 parameters mentioned above. Given the specimen parameters, when "*determine context*" is done, the output indicates (1) *if the user is at home, at work or on the go*, (2) *if user's workplace is safe, sensitive or risky*, and (3) *if the workplace's light intensity is bright, moderate or dark.*

4.3. Context Storage and Dissemination

In general, if a system is to obtain an accurate representation of the user's interaction context, then the system must be introduced to the most number of possible context parameters. As a context parameter is added to the system, the VM's context convention database forms a tree-like IC structure, as shown in generic format in Figure 6. Every new IC parameter is first classified as either UC or EC or SC parameter and is then appended as a branch of UC or EC or SC. Then, the conventions of the parameter are identified.

For the IC information to be propagated in a pervasive system, the data representation used is XML Schema which is based on XML (Ross and Lightman 2005). Figure 7(Left) illustrates the general XML format of a context parameter (i.e. name, units, source of raw data, and conventions) and Figure 7(Right) shows the various snapshots of windows involved in adding a parameter in the VM as implemented using Java programming language.

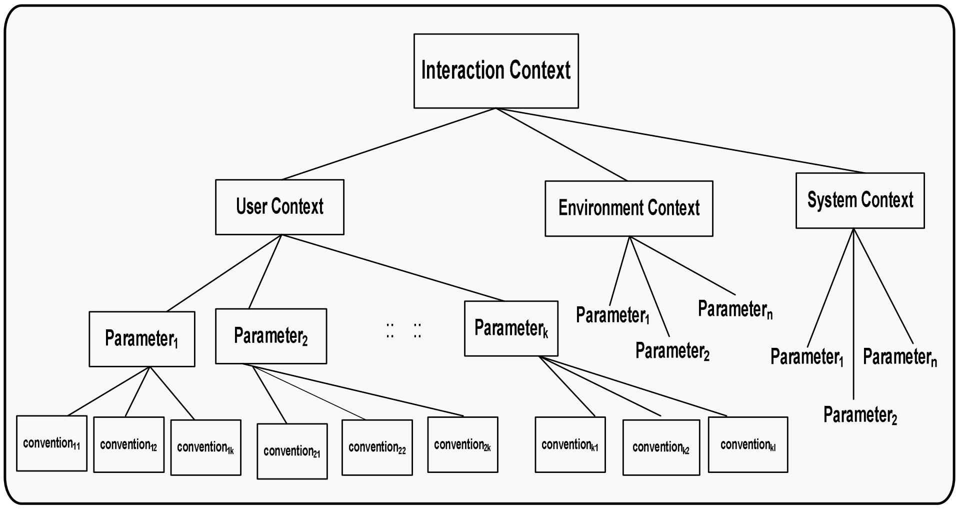
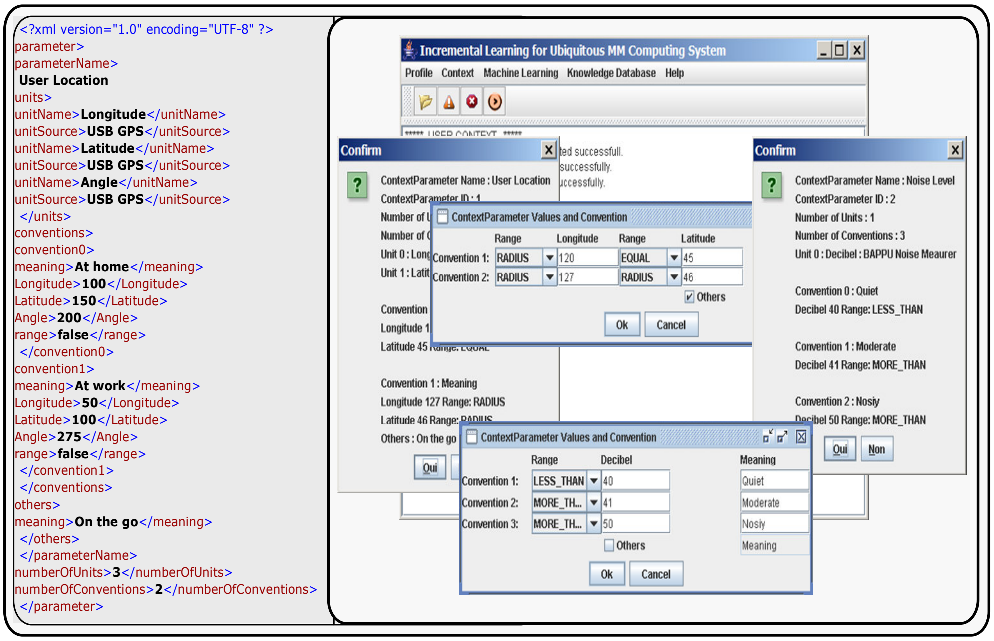


Figure 6. The structure of stored IC parameters.



```
<?xml version="1.0" encoding="UTF-8" ?>
parameter>
parameterName>
 User Location
units>
unitName>Longitude</unitName>
unitSource>USB GPS</unitSource>
unitName>Latitude</unitName>
unitSource>USB GPS</unitSource>
unitName>Angle</unitName>
unitSource>USB GPS</unitSource>
 </units>
conventions>
convention0>
meaning>At home</meaning>
Longitude>100</Longitude>
Latitude>150</Latitude>
Angle>200</Angle>
range>false</range>
 </convention0>
convention1>
meaning>At work</meaning>
Longitude>50</Longitude>
Latitude>100</Latitude>
Angle>275</Angle>
range>false</range>
 </convention1>
 </conventions>
others>
meaning>On the go</meaning>
 </others>
 </parameterName>
numberOfUnits>3</numberOfUnits>
numberOfConventions>2</numberOfConventions>
 </parameter>
```

Incremental Learning for Ubiquitous MM Computing System
Profile Context Machine Learning Knowledge Database Help
***** USER CONTEXT *****

ted successfull.
successfully.
uccessfully.

Confirm

ContextParameter Name : User Location
ContextParameter ID : 1
Number of U
Number of C
Unit 0 : Long
Unit 1 : Latit

Convention
Longitude 1
Latitude 45 Range: EQUAL

Convention 1 : Meaning
Longitude 127 Range: RADIUS
Latitude 46 Range: RADIUS
Others : On the go

Qui

ContextParameter Values and Convention

	Range	Longitude	Range	Latitude
Convention 1:	RADIUS	120	EQUAL	45
Convention 2:	RADIUS	127	RADIUS	46

☑ Others

Ok Cancel

Confirm

ContextParameter Name : Noise Level
ContextParameter ID : 2
Number of Units : 1
Number of Conventions : 3
Unit 0 : Decibel : BAPPU Noise Meaurer

Convention 0 : Quiet
Decibel 40 Range: LESS_THAN

Convention 1 : Moderate
Decibel 41 Range: MORE_THAN

Convention 2 : Nosiy
Decibel 50 Range: MORE_THAN

Qui Non

ContextParameter Values and Convention

	Range	Decibel	Meaning
Convention 1:	LESS_THAN	40	Quiet
Convention 2:	MORE_TH...	41	Moderate
Convention 3:	MORE_TH...	50	Nosiy
		☐ Others	Meaning

Ok Cancel

Figure 7. (Left) Sample context parameter in XML, (Right) snapshots of windows in add.

4.4. Measuring a Modality's Context Suitability

Multimodality refers to the selection of modalities based on its suitability to the given IC. Here, modality refers to the logical interaction structure (i.e. the mode for data input and output between a user and computer). A modality may only be realized if there is/are media devices that would support it. In this work, media refers to a set of physical interaction devices (plus some software supporting the physical devices). Using natural language processing as basis, we group modalities as follows: (i) Visual Input (VI_{in}), (ii) Vocal Input (VO_{in}), (iii) Manual/Tactile Input (M_{in}), (iv) Visual Output (VI_{out}), (v) Vocal Output (VO_{out}), and (vi) Manual/Tactile Output (M_{out}). Multimodality is possible if there is at least one modality for data input and at least one modality for data output.

Using Z language specification (Lightfoot 2001), let there be a set of input modalities and output modalities, as given by INPUTMODE ::= $VI_{in} \mid VO_{in} \mid M_{in}$ and OUTPUTMODE ::= $VI_{out} \mid VO_{out} \mid M_{out}$. Let the relationship multimodality be a set of pairs of input and output modalities, as denoted by multimodality: P (INPUTMODE $\otimes$ OUTPUTMODE) where P = power set which is the set of all subsets denotes power set and $\otimes$ = Cartesian product. At any given time, we can test if multimodality is possible by getting an instance of input and output modalities. Assume that x: P INPUTMODE, y: P OUTPUTMODE. Multimodality is possible if x and y forms a pair within the relationship multimodality and that neither x nor y is an empty set, that is, Possible((x,y)) $\Leftrightarrow$ (x,y) $\in$ multimodality $\wedge$ x $\neq \varnothing \wedge$ y $\neq \varnothing$.

Accordingly, media devices themselves are grouped as follows: (i) Visual Input Media (VIM), (ii) Visual Output Media (VOM), (iii) Oral Input Media (OIM), (iv) Hearing Output Media (HOM), (v) Touch Input Media (TIM) (vi) Manual Input Media (MIM), and (vii) Touch Output Media (TIM). The relationships that map modalities with media group and then the media group with media devices are shown in Figure 8.

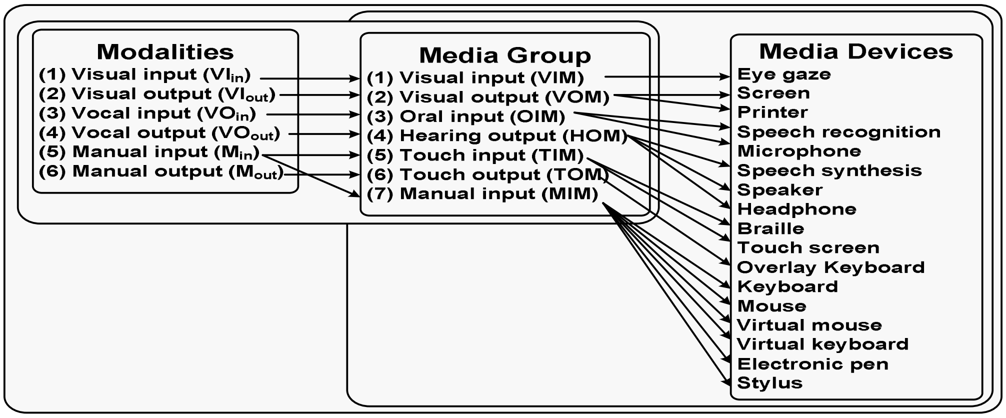


Figure 8. The relationship among modalities, media group and physical media devices.

To build a relationship between modalities and media devices, let there be a function g_1 that maps a modality to a media group, given by g_1: MODALITY → MEDIAGROUP. This is shown in Figure 8. There is often a case, however, when two or more media devices both belong to one media group. In such a case, devices selection is determined through their priority rankings. Hence, let there be another function g_2 that maps a media group to a media device and its priority rank, denoted by g_2: MEDIAGROUP → (MEDIADEVICE, $\mathbb{N}_1$) where $\mathbb{N}_1$ denotes an integer value greater than zero. Sample elements of these functions are:

g_1 = {(VI_{in},VIM), (VI_{out},VOM), (VO_{in},OIM), (VO_{out},HOM), (M_{in},TIM), (M_{in},MIM), (M_{out},TOM)}

g_2 = {(VIM, (eye gaze,1)), (VOM, (screen,1)), (VOM, (printer,1)), (OIM, (speech recognition,1)), (OIM, (microphone,1)), (HOM, (speech synthesis,1)), (HOM, (speaker,2)), (HOM, headphone,1)), etc.}

A modality's suitability to IC is equal to its collective suitability to the IC's individual parameters. Suitability measure is not binary, not just suitable or not suitable, because there are some cases wherein the extent of suitability lies in between. Hence, our suitability measures are *high*, *medium*, *low* and *inappropriate*. High suitability means that the modality in consideration is the preferred mode for computing; medium suitability means the modality is simply an alternative mode for computing, hence, its absence is not considered as an error but its presence means added convenience to the user. Low suitability means the modality's effectiveness is negligible and is the last recourse when everything else fails. Inappropriateness recommends that the modality should not be used at all.

If the collective IC is composed of n parameters, then the modality in consideration has n suitability scores. We then adopt the following conventions:

1. A modality's suitability to an IC parameter is one of the following: H (high), M (medium), L (low), and I (inappropriate). Mathematically, H = 1.00, M = 0.75, L = 0.50, and I = 0.
2. The modality's suitability score to an IC is given by:

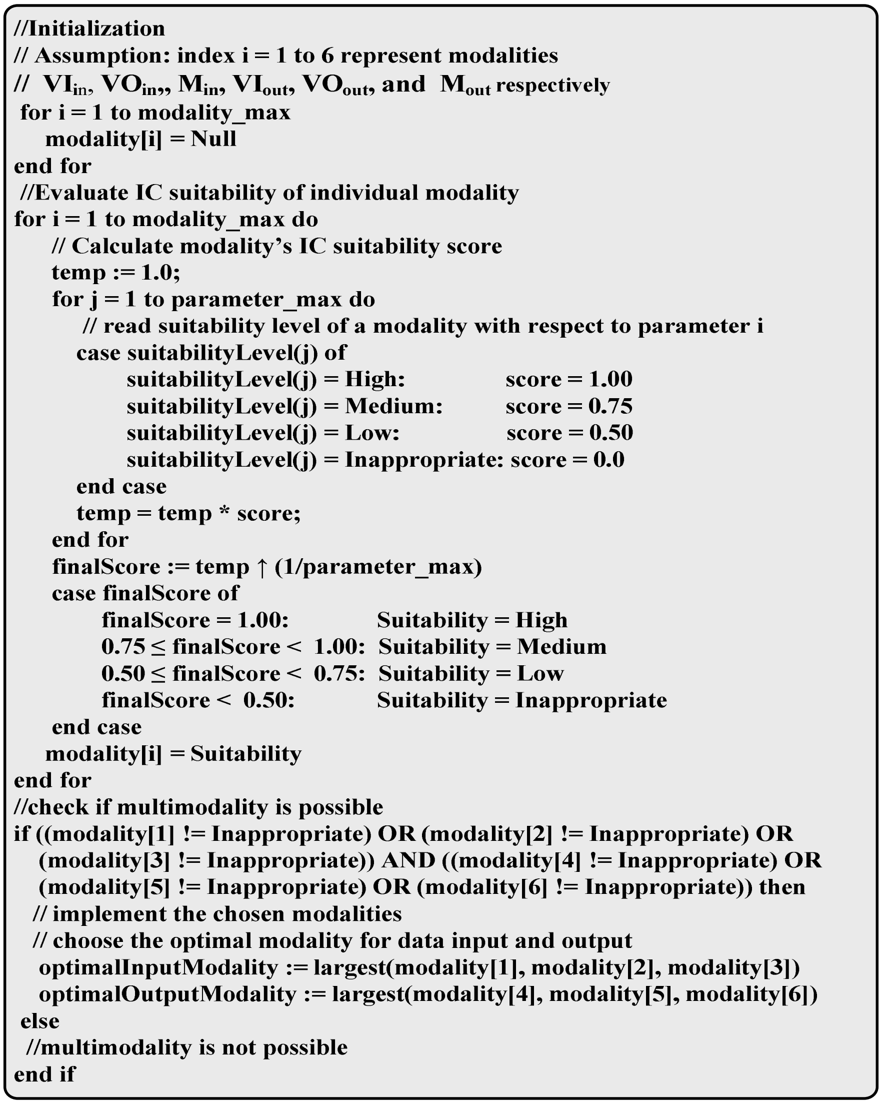
$$\text{SuitabilityScore}_{\text{modality}} = \sqrt[n]{\prod_{i=1}^{n} \text{context\_parameter}_{\,i}} \tag{5}$$

where i = parameter index and n = total number of parameters. Given the calculated value, a modality's IC suitability is given by:

$$\text{FinalSuitability}_{\text{modality}} = \left\langle \begin{array}{c} \text{H if SuitabilityScore}_{\text{modality}} = 1.00 \\ \text{M if } 0.75 \leq \text{SuitabilityScore}_{\text{modality}} < 1.00 \\ \text{L if } 0.50 \leq \text{SuitabilityScore}_{\text{modality}} < 0.75 \\ \text{I if SuitabilityScore}_{\text{modality}} < 0.50 \end{array} \right. \tag{6}$$

4.5. Selecting Context-Appropriate Modalities

Figure 9 shows the algorithm for determining the suitability of modalities to a given IC and if multimodality is possible (i.e. section 4.4). Checking that multimodality is possible is done by determining that not all of input modalities (i.e. specified by indexes 1, 2 and 3) are scored "inappropriate", and so does for output modalities (i.e. specified by indexes 4, 5 and 6). The optimal input modality is chosen from the group of input modalities and is one with the highest IC suitability score. The same principle applies to the selection of the optimal output modality. Subject to the availability of media devices, an optimal modality is ought to be implemented; all others are considered optional. In the absence of supporting media devices, an alternative modality is chosen and is one that has the next highest score.



```
//Initialization
// Assumption: index i = 1 to 6 represent modalities
//  $VI_{in}$, $VO_{in}$,, $M_{in}$, $VI_{out}$, $VO_{out}$, and  $M_{out}$ respectively
 for i = 1 to modality_max
    modality[i] = Null
end for
 //Evaluate IC suitability of individual modality
for i = 1 to modality_max do
     // Calculate modality's IC suitability score
     temp := 1.0;
     for j = 1 to parameter_max do
         // read suitability level of a modality with respect to parameter i
        case suitabilityLevel(j) of
               suitabilityLevel(j) = High:              score = 1.00
               suitabilityLevel(j) = Medium:            score = 0.75
               suitabilityLevel(j) = Low:               score = 0.50
               suitabilityLevel(j) = Inappropriate: score = 0.0
        end case
        temp = temp * score;
     end for
     finalScore := temp ↑ (1/parameter_max)
     case finalScore of
            finalScore = 1.00:            Suitability = High
            0.75 ≤ finalScore <  1.00:  Suitability = Medium
            0.50 ≤ finalScore <  0.75:  Suitability = Low
            finalScore <  0.50:            Suitability = Inappropriate
     end case
    modality[i] = Suitability
end for
//check if multimodality is possible
if ((modality[1] != Inappropriate) OR (modality[2] != Inappropriate) OR
   (modality[3] != Inappropriate)) AND ((modality[4] != Inappropriate) OR
   (modality[5] != Inappropriate) OR (modality[6] != Inappropriate)) then
  // implement the chosen modalities
  // choose the optimal modality for data input and output
   optimalInputModality := largest(modality[1], modality[2], modality[3])
   optimalOutputModality := largest(modality[4], modality[5], modality[6])
 else
  //multimodality is not possible
end if
```

Figure 9. Algorithm to determine a modality's suitability to IC and if multimodality possible.

Again, any alternative modality must be supported by available media devices. This process is repeated until the system is able to find a replacement modality that can be supported by currently available media devices.

4.6. Selecting Media Devices Supporting Modalities

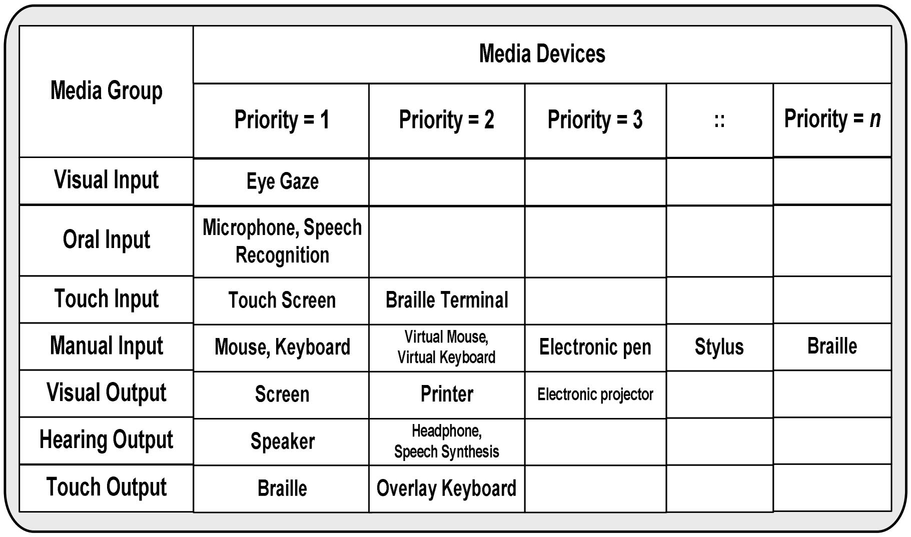
When multimodality is possible and optimal modalities have been chosen, then supporting media devices are checked for availability. Using function g_1, the media group that supports the chosen modality can be identified. Given that MODALITY = {VI_{in}, VO_{in}, M_{in}, VI_{out}, VO_{out}, M_{out}} and MEDIAGROUP = {VIM, OIM, MIM, TIM, VOM, HOM, TOM} and that g_1: MODALITY → MEDIAGROUP, then formally, using Z language, we can specify that for every p which is a selected modality, there corresponds a media group q, wherein neither p nor q is an empty set, such that the ordered pair (p, q) is a member of set g_1, that is $\forall$ p: MODALITY; $\exists$ q: MEDIAGROUP $\mid$ $x \neq \emptyset \wedge y \neq \emptyset \bullet (p, q) \in g_1$.

By using function g_2, the top-ranked media devices that belong to the specified media group can also be identified. Given function g_2, a media device d, priorities p_1 and p_2 of type $\mathbb{N}_1$, then the specification for finding the top-ranked device within the media group m is that the mapping between m and media device d with a priority ranking of d_1 does exist in function g_2 and that the numerical value of device's priority p_1 is less than p_2, (i.e. the lesser the numerical value, the higher is its priority ranking), that is, $\exists$ m: MEDIAGROUP; $\forall$ d: MEDIADEVICE; $\exists$ p_1: $\mathbb{N}_1$; $\forall$ p_2: $\mathbb{N}_1$ $\mid$ $d \bullet m \rightarrow (d, p_1) \in g_2 \wedge (p_1 < p_2)$.

Let there be a media devices priority table (MDPT) (see Table 2) which tabulates all media groups, and its set of media devices arranged by priority ranking. T = {T_1, T_2… T_{max_table}} is the set of MDPT's. The elements of table $T_n \in$ T, where n $\in$ 1 .. max_table, are similar to elements of function g_2. No two MDPT's are identical. To create a new table, at least one of its elements is different from all other tables that have already been defined. The priority ranking of a media device may be different in each MDPT. In general, it is possible that two or more different context scenarios may be assigned to one common MDPT.

When a new media device d_{new} is added or introduced to the system for the *first time*, the device is associated to a media group and is given a priority ranking r by the user. What happen to the rankings of other devices $d_{i,}$ ($i \in$ 1 .. $n_,$ and n = number of media devices) which are in the same media group as d_{new} in the MDPT? Two things may happen, depending on the user's selection. The first possibility is after having the new device's priority Priority(d_{new}) set to r then the priority of the other device i, ($1 \leq i \leq n$) denoted Priority(d_i), remains the same.

Table 2. A sample media devices priority table (MDPT)



Media Group	Media Devices				
	Priority = 1	Priority = 2	Priority = 3	::	Priority = *n*
Visual Input	Eye Gaze				
Oral Input	Microphone, Speech Recognition				
Touch Input	Touch Screen	Braille Terminal			
Manual Input	Mouse, Keyboard	Virtual Mouse, Virtual Keyboard	Electronic pen	Stylus	Braille
Visual Output	Screen	Printer	Electronic projector		
Hearing Output	Speaker	Headphone, Speech Synthesis			
Touch Output	Braille	Overlay Keyboard			

The second possibility is the priority rankings of all media devices (d_i) ranked r or lower are adjusted such that their new priority rankings are one lower than their previous rankings. Formally, this is specified as: $\forall$ i, $\exists$ r: $\mathbb{N}_1$; $\forall$ d_i, $\exists$ d_{new}: MEDIADEVICE | (Priority(d_{new}) = r $\wedge$ Priority(d_i) $\geq$ r) $\Rightarrow$ Priority(d_i)' = Priority(d_i) + 1.

5. Modalities in User Interaction Interface

Here, we wish to determine the selections of modality to be used in the user interaction interface, given that it is already known that multimodality is possible for implementation.

5.1. Media Groups and Media Devices

In general, the association between media group and media devices can be specified as:

$$\text{VIM} = \text{eye gaze} \vee \text{gesture interpreter} \tag{7}$$

$$\text{OIM} = \text{speech recognition} \wedge \text{microphone} \tag{8}$$

$$\text{MIM} = \text{keyboard} \vee \text{Braille} \vee \text{pen} \tag{9}$$

$$\text{TIM} = \text{mouse} \vee \text{virtual mouse} \vee \text{touch screen} \tag{10}$$

$$\text{VOM} = \text{terminal screen} \vee \text{printer} \tag{11}$$

$$\text{HOM} = \text{speech synthesis} \wedge (\text{speaker} \vee \text{headset}) \tag{12}$$

$$\text{TOM} = \text{tactile keyboard} \vee \text{Braille} \tag{13}$$

Note that the relationships cited above list down only limited number of commonly-used media devices. That said, these relationships can be modified easily and accordingly to include other media devices.

Given that INPUT_MEDIA_GROUP = {VIM, OIM, TIM, MIM}, then the power set (i.e. the set of all subsets) of this group is given by P(INPUT_MEDIA_GROUP) = {{VIM}, {OIM}, {TIM}, {MIM}, {VIM, OIM}, {VIM, TIM}, {VIM, MIM}, {VIM, OIM, TIM}, {VIM, OIM, MIM}, {VIM, TIM, MIM}, {VIM, OIM, TIM, MIM}, {OIM, TIM}, {OIM, MIM}, {OIM, TIM, MIM}, {TIM, MIM}, {}}. These results indicate that as far as human-machine interaction interface is concerned, there can be four types of user interface. Note that, by definition, an interface is a function of input modalities only. Hence, the possible types of human-machine interaction interfaces are:

- *unimodal interface* – media devices (and supporting software) belonging to VIM, OIM, TIM and MIM can be used, but there is no fusion of data generated by one media with the data generated by another media (ex. speech, pen, vision)
- *bimodal interface* – there are 6 possible combinations of fusion of data generated by two media devices – that of {VIM, OIM}, {VIM, TIM}, {VIM, MIM}, {OIM,

TIM}, {OIM, MIM}, and {TIM, MIM}. The current state-of-the-art multimodal interfaces fall in this category.

- *trimodal interface* – this interface allows the combination of data that are generated by three media devices into a new meaningful data; there are 4 possible selections, namely: {VIM, OIM, TIM}, {VIM, OIM, MIM}, {VIM, TIM, MIM}, and {OIM, TIM, MIM}
- *quadmodal interface* – this one would combine all types of input media altogether, {VIM, OIM, TIM, MIM}.

As far as research advancement (i.e. year 2008) is concerned, a user interface can only be unimodal or bimodal. There is no evidence that a trimodal interface, let alone a quadmodal one, exists, at least not yet.

5.2. Selection of User Interface Modalities

Given that a unimodal or bimodal user interface is possible, then the system, in consultation with the user, decides the most suitable user interface. We believe that the selection of user interface that suits the user should be based on (i) the modalities and media groups that suit the given context (ii) the availability of media devices (and their supporting software) that would support the chosen modalities, and (iii) the availability of the preferred interface system or middleware within the user's computing system, and (iv) the user's preference on these interface as given by their priority rankings.

In order to determine whether the system will implement a unimodal or multimodal interface, let there be a *human-machine interaction interface priority table* (HMIIPT). This table contains important information related to the user's preferences, such as: (i) the priority ranking of multimodal and unimodal interface, (ii) the priority ranking of modalities within the interface, and (iii) the priority ranking of media devices that support a modality. See Table 3 for a sample HMIIPT.

Suppose that the user prefers a unimodal interface over a multimodal (actually, bimodal) one. Using HMIIPT, the system then determines the ranking assigned to each of the input modalities. Then the priority ranking of media devices supporting a preferred modality is taken from the media devices priority table (see Table 1). For multimodal interface, the user is also consulted in the priority rankings of all modality combinations/fusion. In the same manner, the priority of media devices supporting the multimodal fusion is also indicated in HMIIPT.

The selection process for optimal user interface modality uses the following functions to determine the score of each user interface mode. We take the result yielding the highest score as the optimal user interface modality:

$$\begin{aligned}\text{User Interface Modality Score} &= \text{User Interface Priority} \\ &\quad \text{x Modality Priority} \times \text{Media Devices Priority}\end{aligned} \tag{14}$$

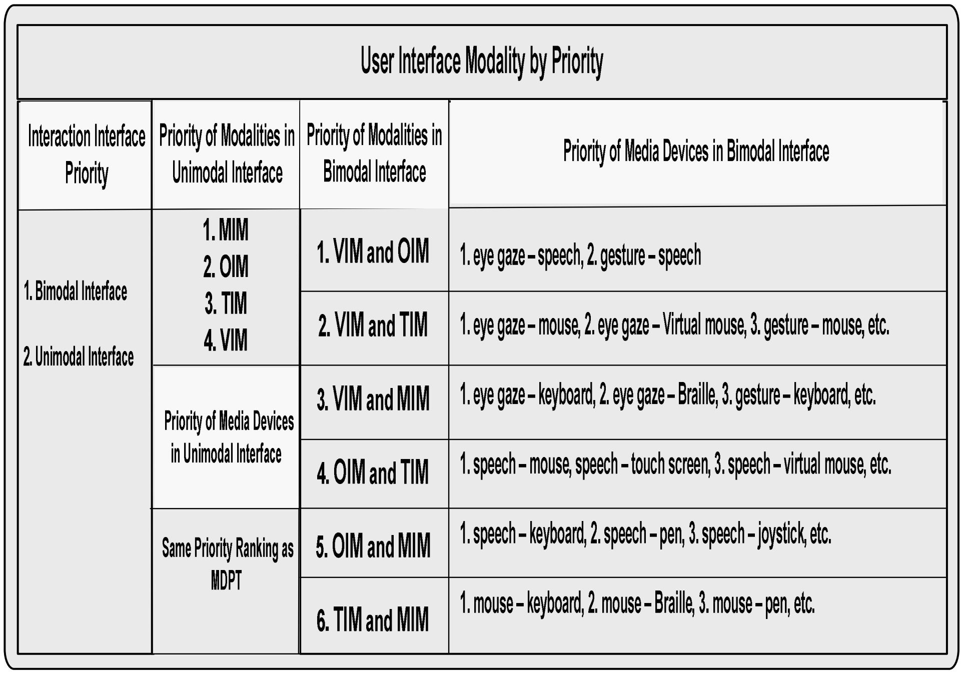
$$\text{User Interface Priority} = (m + 1 - p) / m \tag{15}$$

$$\text{Modality Priority} = (q + 1 - p) / q \tag{16}$$

$$\text{Media Device}_i \text{ Priority} = 1 - \sum_{1}^{i-1}(1/n) \tag{17}$$

such that for user interface priority, the variable m = number of types of user interface (i.e. unimodal, bimodal, etc. available in HMIIPT) and p = priority ranking as obtained from HMIIPT, $1 \leq p \leq m$. For modality's priority, the variable q = number of modality selections (i.e. available in HMIIPT) and p = priority ranking of the specified modality as obtained from HMIIPT, $1 \leq p \leq q$. For media devices priority, n = available media devices supporting the chosen modality and media group (see Equations (7) through (13)). Also, given the i^{th} device, where $1 \leq \mathrm{i} \leq \mathrm{n}$, then d_i = priority ranking obtained from MDPT and n = number of media devices supporting the same modality as the i^{th} device.

Table 3. A sample human-machine interaction interface priority table (HMIIPT)



User Interface Modality by Priority

Interaction Interface Priority	Priority of Modalities in Unimodal Interface	Priority of Modalities in Bimodal Interface	Priority of Media Devices in Bimodal Interface
1. Bimodal Interface 2. Unimodal Interface	1. MIM 2. OIM 3. TIM 4. VIM	1. VIM and OIM	1. eye gaze – speech, 2. gesture – speech
		2. VIM and TIM	1. eye gaze – mouse, 2. eye gaze – Virtual mouse, 3. gesture – mouse, etc.
	Priority of Media Devices in Unimodal Interface	3. VIM and MIM	1. eye gaze – keyboard, 2. eye gaze – Braille, 3. gesture – keyboard, etc.
		4. OIM and TIM	1. speech – mouse, speech – touch screen, 3. speech – virtual mouse, etc.
	Same Priority Ranking as MDPT	5. OIM and MIM	1. speech – keyboard, 2. speech – pen, 3. speech – joystick, etc.
		6. TIM and MIM	1. mouse – keyboard, 2. mouse – Braille, 3. mouse – pen, etc.

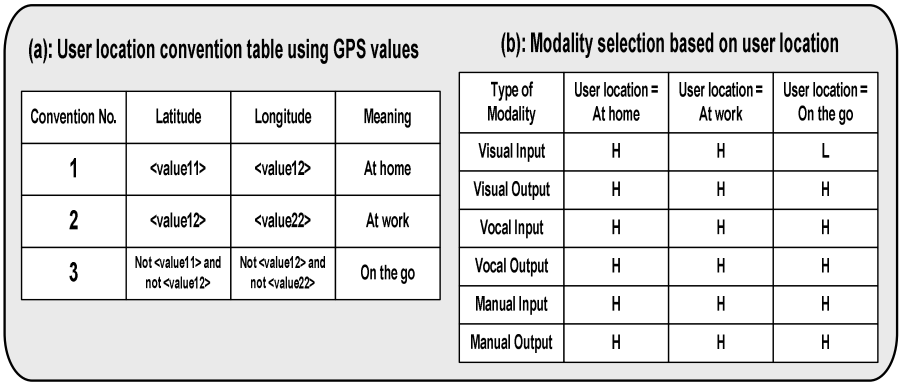
6. Sample Cases

Here, we simulate sample cases and accordingly apply the principles discussed in the previous sections.

6.1. Sample Interaction Context and Its Corresponding Modalities Selection

Suppose that we are given the following interaction context: (i) *user context*: user location = at home, user handicap = none, (ii) *environment context:* noise level = quiet, safety factor = safe, (iii) *system context:* computing device = PDA. Suppose that the context convention database contains the conventions and suitability score of different modalities as shown in Tables 4 to 8. What will be the optimal modality?

Table 4. User location conventions and suitability scores



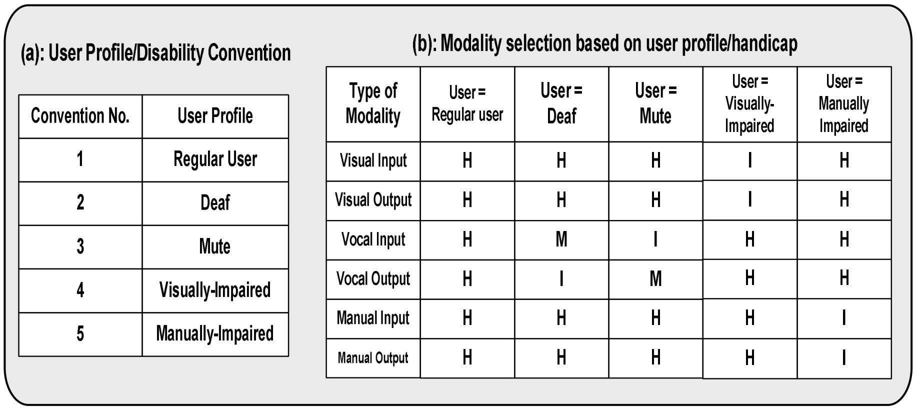
(a): User location convention table using GPS values

Convention No.	Latitude	Longitude	Meaning
1	<value11>	<value12>	At home
2	<value12>	<value22>	At work
3	Not <value11> and not <value12>	Not <value12> and not <value22>	On the go

(b): Modality selection based on user location

Type of Modality	User location = At home	User location = At work	User location = On the go
Visual Input	H	H	L
Visual Output	H	H	H
Vocal Input	H	H	H
Vocal Output	H	H	H
Manual Input	H	H	H
Manual Output	H	H	H

Table 5. User disability conventions and suitability scores



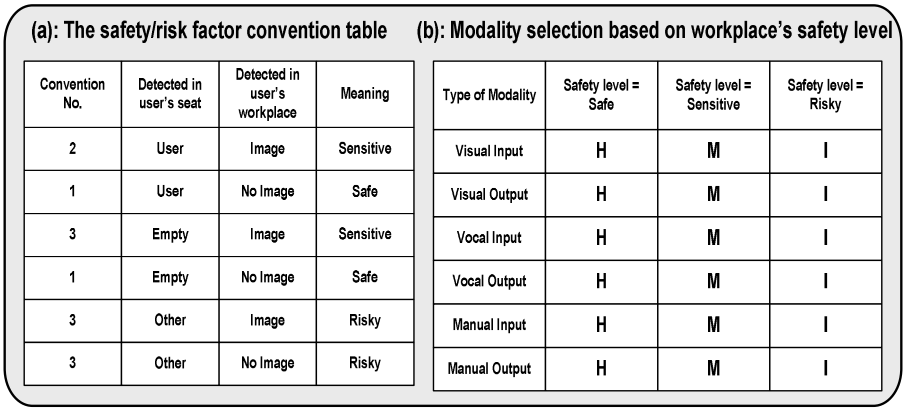
(a): User Profile/Disability Convention

Convention No.	User Profile
1	Regular User
2	Deaf
3	Mute
4	Visually-Impaired
5	Manually-Impaired

(b): Modality selection based on user profile/handicap

Type of Modality	User = Regular user	User = Deaf	User = Mute	User = Visually-Impaired	User = Manually Impaired
Visual Input	H	H	H	I	H
Visual Output	H	H	H	I	H
Vocal Input	H	M	I	H	H
Vocal Output	H	I	M	H	H
Manual Input	H	H	H	H	I
Manual Output	H	H	H	H	I

Table 6. Workplace safety conventions and suitability scores



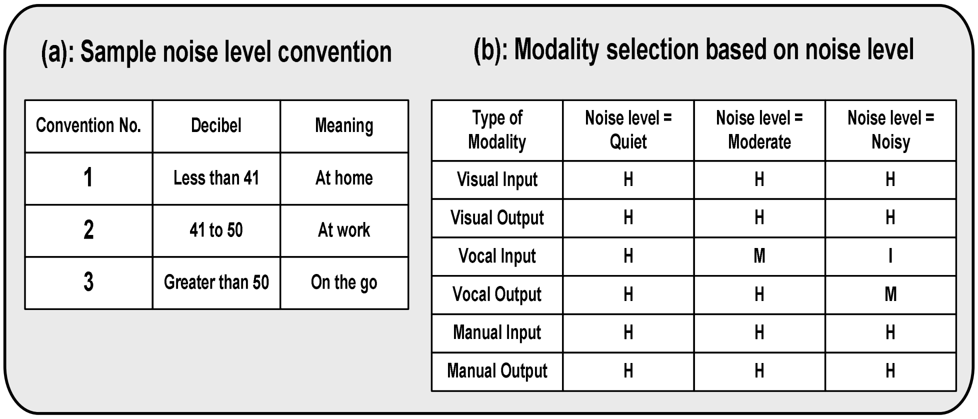
(a): The safety/risk factor convention table

Convention No.	Detected in user's seat	Detected in user's workplace	Meaning
2	User	Image	Sensitive
1	User	No Image	Safe
3	Empty	Image	Sensitive
1	Empty	No Image	Safe
3	Other	Image	Risky
3	Other	No Image	Risky

(b): Modality selection based on workplace's safety level

Type of Modality	Safety level = Safe	Safety level = Sensitive	Safety level = Risky
Visual Input	H	M	I
Visual Output	H	M	I
Vocal Input	H	M	I
Vocal Output	H	M	I
Manual Input	H	M	I
Manual Output	H	M	I

Table 7. Noise level conventions and suitability scores



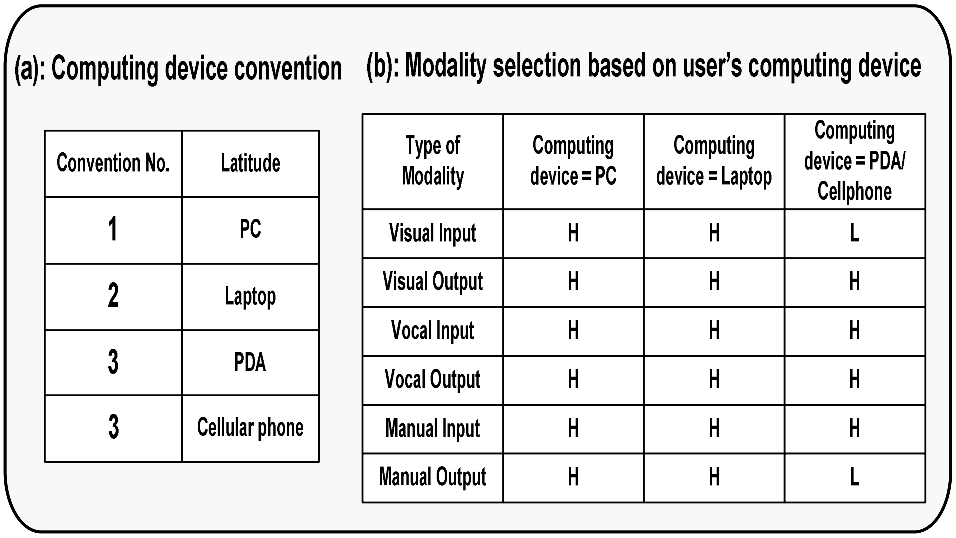
(a): Sample noise level convention

Convention No.	Decibel	Meaning
1	Less than 41	At home
2	41 to 50	At work
3	Greater than 50	On the go

(b): Modality selection based on noise level

Type of Modality	Noise level = Quiet	Noise level = Moderate	Noise level = Noisy
Visual Input	H	H	H
Visual Output	H	H	H
Vocal Input	H	M	I
Vocal Output	H	H	M
Manual Input	H	H	H
Manual Output	H	H	H

Table 8. Computing device conventions and suitability scores



(a): Computing device convention

Convention No.	Latitude
1	PC
2	Laptop
3	PDA
3	Cellular phone

(b): Modality selection based on user's computing device

Type of Modality	Computing device = PC	Computing device = Laptop	Computing device = PDA/ Cellphone
Visual Input	H	H	L
Visual Output	H	H	H
Vocal Input	H	H	H
Vocal Output	H	H	H
Manual Input	H	H	H
Manual Output	H	H	L

The given interaction context is IC = (c_1, c_2, c_3, c_4, c_5) = (1, 1, 1, 1, 3). The calculated final suitability scores of each type of modality are given below:

Visual Input= $[(H)(H)(H)(H)(L)]^{1/5}$ = $[(1)(1)(1)(1)(0.50)]^{1/5}$ = 0.87 = Medium suitability
Vocal Input = $[(H)(H)(H)(H)(H)]^{1/5}$ = $[(1)(1)(1)(1)(1)]^{1/5}$ = 1 = High suitability
Manual Input = $[(H)(H)(H)(H)(H)]^{1/5}$ = $[(1)(1)(1)(1)(1)]^{1/5}$ = 1 = High suitability
Visual Output = $[(H)(H)(H)(H)(H)]^{1/5}$ = $[(1)(1)(1)(1)(1)]^{1/5}$ = 1 = High suitability
Vocal Output = $[(H)(H)(H)(H)(H)]^{1/5}$ = $[(1)(1)(1)(1)(1)]^{1/5}$ = 1 = High suitability
ManualOutput=$[(H)(H)(H)(H)(L)]^{1/5}$=$[(1)(1)(1)(1)(0.50)]^{1/5}$= 0.87 = Medium suitability

Given this case, multimodality is possible (see section 4.4). The preferred *input modality* is either Vocal Input (VO_{in}) or Manual Input (M_{in}). The preferred *output modality* is Visual Output (VI_{out}) or Vocal Output (VO_{out}). All non-optimal modalities are considered *optional*. Using Figure 8 as visual reference, the media groups that suit the given interaction context are OIM, MIM, VOM, and HOM.

6.2. Sample Media Devices and User Interface Selection

Consider that the same user has the following media devices: OIM: speech recognition system, microphone, MIM: mouse, keyboard and electronic pen, VOM: screen and keyboard, and VOM: speech synthesis, speaker. Question: what is the most suitable human-computer interaction interface for the user?

To answer this question, we need to know the user preferences concerning media devices and user interface. Assuming that the data in the specimen MDPT and HMIIPT apply, then:

i. Bimodal Interface Priority = (2 + 1 – 1)/2 = 1
ii. Modality Priority: OIM and MIM = (6 + 1 - 5)/6 = 0.33
iii. Media Devices Priority: speech-keyboard = 1, speech-pen = 2/3
iv. Unimodal Interface Priority = (2+1-2)/2 = 0.5
v. Modality Priority: MIM = 1, OIM = 0.75
vi. Media Devices Priority: OIM: speech = 1 MIM: keyboard = 1, pen = 3/5

The calculation for bimodal interface follows:

(i) speech-keyboard = 1*0.33*1 = 0.33
(ii) speech-pen = 1* 0.33*0.67 = 0.22

For, unimodal interface, the result is:

MIM: (i) keyboard = 0.5 * 0.75 * 1 = 0.375

(ii) pen = 0.5 * 0.75 * 0.6 = 0.225, and
OIM: speech = 0.5 * 0.75 * 1 = 0.375.

In this case, the system determines that the optimal user interface is a unimodal one in which the optimal input device/modality is the keyboard and speech.

7. Our Multimodal Multimedia Computing System

7.1. Architectural Framework

Our proposed system is conceived for two purposes: (1) to contribute to multimodal multimedia computing research and (2) to further advance self-adaptive computing system. To achieve the first goal, we develop the model that relates modality with user context, and associate media devices to support the implementation of the chosen modality. In the second goal, we advocate the propagation of knowledge, acquired through training, into the user's computing environment so that such knowledge can be used for system adaptation based on user's requirements and system's restrictions. The major components of our multimodal multimedia computing system are shown in Figure 10. The functionality of each component is given below:

- *The Task Manager Agent (TMA)* – manages user's profile, task and pertinent data and their deployment from a server to the user's computing device, and vice versa.
- *The Context Manager Agent (CMA)* – detects user context from sensors and user profile, and selects the modality and media apt for the context.
- *The History and Knowledge-based Agent (HKA)* – responsible for ML training and knowledge acquisition.
- *The Layered Virtual Machine for Interaction Context (LVMIC)* – detects sensor-based context and allows the incremental definition of context parameters.
- *The Environmental Manager Agent (EMA)* – detects available and functional media devices in the user's environment.

In the diagram, the user (Manolo) can work at home, logs out, and still continue working on the same task at anytime and any place. Due to user's mobility, the variation in user's context and available resources is compensated by a corresponding variation in modality and media devices, and user interface selections.

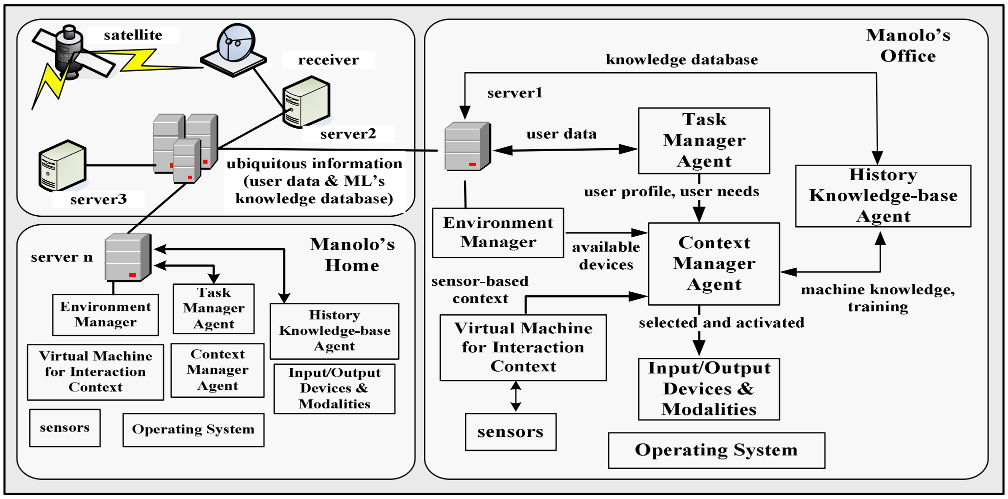


Figure 10. The architecture of a context-aware ubiquitous multimodal computing system.

7.2. Ubiquity of System Knowledge and Experience

The HKA is the component that incorporates incremental learning to the system. Its knowledge, for now, is concentrated on the system's ability to configure modalities, media devices and user interface based on the given context.

In the beginning, we assume that our system would have no knowledge whatsoever. Its initial knowledge is related to context parameter and its conventions, obtained incrementally as every relevant context parameter gets added using our VM. The system will acquire extra knowledge when HKA interacts with CMA (see Figure 11).

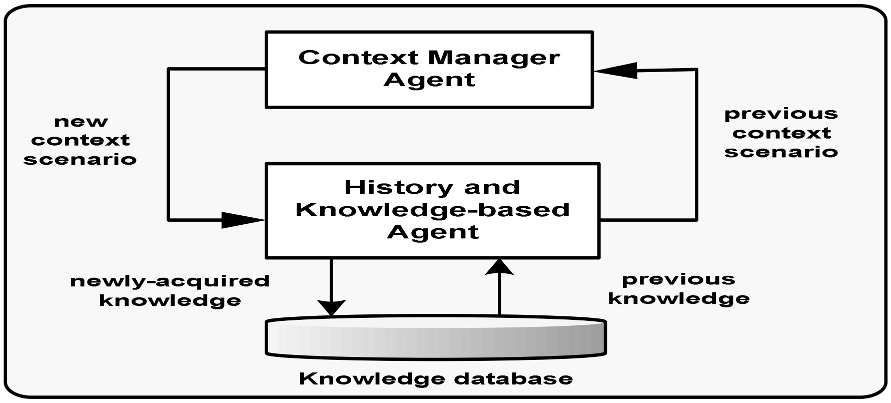


Figure 11. The History and Knowledge–based Agent at work.

Through CMA, the HKA obtains information containing the user's pre-condition scenario (i.e. the instance of interaction context), and accordingly determines the corresponding post-condition scenario (i.e. the selected modalities, media devices, and user interface). Each unique instance of interaction context (pre and post conditions) forms an entry in the knowledge database.

The system adds newly-acquired knowledge onto the database. Whenever a situation arrives that the system needs to do some decision or calculation based on the given instance of IC, the system first consults the database for any previous knowledge. If an exact match exists, then the system would simply implement the applicable set-up or post-condition scenario. If no match is found, then the system would have to do all the calculations as this is a new case. Afterwards, the result of the calculations becomes a newly acquired knowledge that is appended onto the database. Over time, the system would have enough knowledge to deal with almost all conceivable IC situations. Ideally, when the system could react automatically for almost every conceivable computing condition with minimum or no human intervention, we then say that the machine is "*intelligent*".

To implement a pervasive system, the system's knowledge database needs to be transportable from one computing environment to another. A model of migration of user's task and of machine's knowledge is already demonstrated in our previous work, as applied to visually-impaired users (Awde, Hina et al. 2007). We intend to apply the same main principle into this system.

8. Conclusion and Future Works

In this paper, we noted that present-day applications of pervasive system do not include that of pervasive multimodality. The current state-of-the-art multimodal interfaces themselves are not apt for pervasive application since they are conceived and developed with pre-defined modalities from the very beginning. Indeed, there is a need for a pervasive multimodal multimedia computing system that would serve both stationary and mobile users, chooses modalities (i.e. mode of human-computer interaction to input and output data) that are appropriate to the given interaction context (i.e. user context + environment context + system context), chooses media devices to support the selected modality and chooses the optimal unimodal/multimodal interface based on user's preferences. This paper enumerates some software engineering challenges in designing the system's infrastructure and explains some details on how those challenges are addressed. We then show the architectural framework of our system and explain briefly our vision on the system's incremental knowledge acquisition.

Our future works include the system's dynamic configuration of its applications whenever computing resources become scarce. Also, more knowledge acquisition and machine learning algorithms need to be developed to make our system exhibits more autonomic computing system features (i.e. self-optimization, self-protection and self-healing).

Acknowledgement

This research work has been made possible through the funding provided by the *Natural Sciences and Engineering Research Council of Canada* (NSERC), the *Décanat des études* of École de technologie supérieure and *La bourse de cotutelle* (Programme Frontenac) provided by the governments of Quebec and France.

REFERENCES

Awde, A., Hina, M. D., Bellik, Y., Ramdane-Cherif, A. and Tadj, C. (2007). Task Migration in a Pervasive Multimodal Multimedia Computing System for Visually-Impaired Users. *GPC 2007 - 2nd International Conference on Grid and Pervasive Computing*, Paris, France, LNCS 4459, Springer Verlag.

Bouhuys, A. (1995). "Induction of depressed and elated mood by music influences – the Perception of Facial Emotional Expressions in Healthy Subjects." *Journal of Affective Disorder*s 33: 278 - 282.

Brown, P. J., Bovey, J. D., and Chen, X. (1997). Context-Aware Applications: From the Laboratory to the Marketplace. *IEEE Personal Communications*. 4: 58 - 64.

Costa, C. A. d., Yamin, A. C., and Geyer, C.F.R. (2008). Toward a General Infrastructure for Ubiquitous Computing. IEEE Pervasive Computing, *IEEE Computer Society*. 7: 64 - 73.

Dey, A. K. (2001). "Understanding and Using Context " *Springer Personal and Ubiquitous Computing* 5(1): 4 - 7.

Dey, A. K., Salber, D., Masayasu, F., and Abowd, G.D. (1999). An Architecture to Support Context-Aware Applications. *GVU Technical Report* GIT-GVU-99-23.

Djenidi, H., Ramdane-Cherif, A., Tadj, C. and Levy, N. (2002). Dynamic Multi-agent Architecture for Multimedia Multimodal Dialogs. *IEEE Workshop on Knowledge Media Networking*.

Dong, S. H., Xiao, B., and Wang. G. P. (2000). "Multimodal user interface for internet." *Jisuanji Xuebao/Chinese Journal of Computers* 23(12): 1270-1275.

Garlan, D., Siewiorek, D., Smailagic, A., and Steenkiste, P. (2002). "Project Aura: Towards Distraction-Free Pervasive Computing." *IEEE Pervasive Computing, Special Issue on Integrated Pervasive Computing Environments* 21(2): 22 - 31.

Horn, P. (2001). Autonomic Computing: *IBM's Perspective on the State of Information Technology*, IBM Research.

Kephart, J. O. and Chess, D. M., (2001). *The Vision of Autonomic Computing*, IBM Thomas J. Watson Research Centre.

Lightfoot, D. (2001). *Formal Specification Using Z*, McMillan Press.

Oviatt, S. (2002). *Multimodal Interfaces. Handbook of Human-Computer Interaction*. J. Jacko and A. Sears. New Jersey, USA, Lawrence Erbaum.

Oviatt, S. L. (2000). "Designing the user interface for multimodal speech and gesture applications: State-of-the-art systems and research directions." *Human Computer Interaction* 15(4): 263 - 322.

Oviatt, S. L. and Cohen, P. R. (2000). "Multimodal Interfaces that Process What Comes Naturally." *Communications of the ACM* 43(3): 45 - 53.

Pahlavan, K. and Krishnamurthy, P., (2002). *Principles of Wireless Networks*.

Pascoe, J. (1998). *Generic Contextual Capabilities to Wearable Computers*. 2nd International Symposium on Wearable Computers.

Rey, G. and Coutaz, J. (2004). The Contextor Infrastructure for Context-Aware Computing. 18th European Conference on Object-Oriented Programming (ECOOP 04), *Workshop on Component-Oriented Approach to Context-Aware Systems*, Oslo, Norway.

Ringland, S. P. A. and Scahill, F. J. (2003). "Multimodality - The future of the wireless user interface." *BT Technology Journal* 21(3): 181-191.

Ross, D. A. and Lightman, A. (2005). Talking Braille: A Wireless Ubiquitous Computing Network for Orientation and Wayfinding *Proceedings of the 7th International ACM SIGACCESS Conference on Computers and Accessibility Baltimore*, MD, USA ACM Press.

Rubin, P., Vatikiotis-Bateson, E., and Benoit, C. (1998). "Audio-Visual Speech Processing." *Speech Communications* 26(1-2): 1998.

Ryan, N., Pascoe, J., and Morse, D. (1997). Enhanced Reality Fieldwork: the Context-Aware Archeological Assistant. *Exxon Computer Applications in Archeology*.

Salehie, M. and Tahvildari, L. (2005). Autonomic Computing: Emerging Trends and Open Problems. *2005 Workshop on Design and Evolution of Autonomic Application Software*, St. Louis, Missouri, USA, ACM Press.

Satyanarayanan, M. (1990). "Scalable, secure, and highly available distributed file access." *Computer* 23(5): 9-18.

Satyanarayanan, M. (1996). "Mobile information access." *IEEE Personal Communications* 3(1): 26-33.

Satyanarayanan, M. (2001). "Pervasive Computing: Vision and Challenges." *IEEE Personal Communications* 8(4): 10-17.

Schilit, B. and Theimer, M. (1994). "Disseminating Active Map Information to Mobile Host." *IEEE Network* 8(5): 22 - 32.

Vallejos, J., Desmet, B., (2007). "Pervasive Communication: The Need for Distributed Context Adaptations" ECOOP 2007 *Workshop on Object Technology for Ambient Intelligence and Pervasive Systems*, Berlin, Germany.

Varshney, U. (2003). "Pervasive Healthcare." *Computer* 36(12): 138 - 140.

Vasilakos, A. V. and Pedrycz, W. (2006). *Ambient Intelligence, Wireless Networking, Ubiquitous Computing*. USA, ArtecHouse.

Ward, A., Jones, A., and Hooper, A. (1997). "A New Location Technique for the Active Office." *IEEE Personal Communications*: 42 - 47.

Weiser, M. (1991). "The computer for the twenty-first century." *Scientific American* 265(3): 94 - 104.

Zhang, Q., Imamiya, A., Go, K. and Mao, X. (2004). *A gaze and speech multimodal interface*, Hachioji, Japan, Institute of Electrical and Electronics Engineers Inc., Piscataway, USA.

WEBSITES

Noise Measuring Device, ELK Company, at http://www.bappu.com/

Rayming Corp. USB GPS Receiver, at http://www.rayming.com

HTG Advance System's retinal recognition by a camera, at http:// www.informatik.uniaugsburg.de/~kimjongh/biometrics/retinal.pdf

Infrared Detector, at http://www.globalsources.com/manufacturers/ InfraredDetector.html

USB Digital Light Meter", at http://www.gossen-photo.de/english/ lichtmess_produkte.html