Do-it-Yourself Smart Experiences

ITEA 2 project 08005

SotA report Smart Space DIY application creation and interaction design

D4.1

Editor:

Juan R. Velasco and Mario Vega – Universidad de Alcalá

Contributors:

Zhenzhen Zhao – Institut TELECOM SudParis (ITS)

Dries De Roeck & Christof van Nimwegen – Centre for User Experience Research, K.U.Leuven (CUO)

A. Outtagarts & B. Boidart – Alcatel-Lucent Bell Labs France.

Yan Tang – VUB STARLab

Security: Public

Version: 3.0

Date: April 6, 2010

Number of pages: 128

History

Version	Date	Person, Partner	Comment
0.1.0	13.10.2009	Mario Vega – Universidad de Alcalá	Creation of the document
0.1.1	19.10.2009	Johan Plomp - VTT	Various comments and additions to the TOC
0.1.2	19.10.2009	Mario Vega - UAH	Added interest on topics Comes from 0.1.1 VTT
0.1.3	30.10.2009	Juncal Alonso-ESI-Tecnalia	Added interest on topics
0.1.4	9.11.2009	Sergio Rodriguez – Robotiker-Tecnalia	Added interest on topics
0.1.5	16.11.2009	Zhenzhen ZHAO – Institut Telecom	Add interest & comment /reorganize TOC
0.1.6	18.11.2009	Dries De Roeck – CUO	Added interest
0.1.7	18.11.2009	Bertrand Boidart - ALBLF	Additions to the TOC
0.1.8	19.11.2009	Johan Plomp - VTT	Added interest
0.2.0	20.11.2009	Mario Vega – Universidad de Alcalá	Reorganize document
0.2.1	26.11.2009	Leire Bastida,Marisa Escalante, Juncal Alonso-ESI-Tecnalia	Added content
0.2.2	01.12.2009	Sergio Rodriguez – Robotiker-Tecnalia	Added content
0.2.4	04.12.2009	Bertrand Boidart – ALBLF	Added content
0.2.5	08.12.2009	Abdelkader Outtagarts - ALBLF	Added content (in 3.4.2)
0.3.0	10.12.2009	Mario Vega Barbas – Universidad de Alcalá	Convergence to 0.2.x documents & added content
0.3.1	18.12.2009	Sergio Rodriguez – Robotiker-Tecnalia	Added content
0.3.2	18.12.2009	Karen Torben Nielsen, An Jacobs, Wendy Van den Broeck, Jo Pierson	Added content
0.3.3	28.12.2009	J. F. Rodelgo, ATOS	Added content
0.3.4	06.01.2010	Yan Tang, VUB STARLab	Added content
0.3.5	10.1.2010	J. Kela, Finwe	Added content
0.3.6	12.1.2010	Minh Phan, Neotiq	Added content

0.4.1	**14.1.2010**	**A. Outtagarts & B. Boidart – ALBLF.**	**Added content and update of 3.4.2**
0.5	**15.1.2010**	**Mario Vega – UAH**	**Added content and merge**
0.5.1	**15.1.2010**	**Johan Plomp - VTT**	**Added content**
0.5.2	**15.1.2010**	**Mario Vega – UAH**	**Added content**
0.5.3	**17.1.2010**	**Dries De Roeck – KULCUO**	**Added content, TOC update**
0.6	**18.1.2010**	**Mario Vega – UAH**	**Edited final viersion**
2.1	**05.04.2010**	**Stijn Mostinckx – VUB SOFT**	**Added short description of AmbientTalk**
3.0	**06.04.2010**	**Mario Vega – UAH**	**Edited final viersion**

The DiYSE Project

The Do-it-Yourself Smart Experiences project (DiYSE) aims at enabling ordinary people to easily create, setup and control applications in their smart living environments as well as in the public Internet-of-Things space, allowing them to leverage aware services and smart objects for obtaining highly personalised, social, interactive, flowing experiences at home and in the city.

www.dyse.org

The three State of the Art Deliverables.

The DiYSE project has three State of the Art (SOTA) Documents covering the different tools, techniques, methods and environments that may be used to provide a DiYSE platform. These documents present the same pool of elements from different points of view. Due to this SOTA partition, it will be needed to link some of the sections from one of the documents to other sections on some of the other two documents. This is really important in Section 3 of the D1.1 and D4.1 that will try to present the same topics from the requirements and the interfaces point of view: web technologies, mobile technologies, platforms, devices, etc…

In WP1 (Use cases and requirements), deliverable D1.1 will focus on which requirements will be covered by web technologies, mobile technologies, system platforms and toolsets, i.e. how the users will access the smart experiences by using these systems.

This document will present the SOTA of current applications, systems platforms and business models relating to DiYSE. This includes Ambient Experience applications, features of toolsets and the business models and ecosystems that are working at this moment in similar proposals.

The document also includes a PEST (Political, Economical, Social, Technological) analysis. It is important to know why people are motivated to produce and share services, devices, etc.

In WP2 (Interaction with the environment), deliverable D2.1 will focus on the State of the art of:

- electronic devices that can retrieve data from the users' environment and produce physical outputs,
- algorithms to extract information from them (such as identification or location) and the functionalities those devices can provide in DiYSE,
- networking technologies to interconnect them.

In particular, D2.1 will put a special emphasis on the following kinds of devices:

- existing ready-made devices available in the market,
- networks of tiny battery-powered programmable wireless sensors,
- open hardware platforms used by DiY hobbyists.

In WP4 (Interactive Experience Creation), deliverable D4.1 analyses similar elements to D1.1, but from the user/developer point of view: how the users will use the elements of the DiYSE ecosystem.

The review of existing application creation approaches will identify technologies that may be supportive for the envisioned creation of applications and services in smart spaces. It is expected that lessons learnt from methods empowering users in the world wide web to contribute content or even applications to communities may provide a good base. Also the issue of actually do-it-yourself versus do-it-together (or have the community do it for you) and crowdsourcing will be addressed. The document reviews how to create interactive experiences.

Abstract of this deliverable

SoA of user interface models, devices and toolsets, that may be used to provide rich and interactive experiences from a do-it-yourself point of view. In this way, the document analyses the SoA on interfaces for service creation tools, and discuss about the kind of user that may create new services by using these tools. On the other hand, the document reviews the different kind of interfaces that may be used at this moment to use services in different devices, both fixed and mobile.

TABLE OF CONTENTS

1 Introduction

As technology advances and becomes more pervasive, the DiY (Do-it-Yourself) paradigm that emerged on the furniture & home decoration market in the 70's is now experiencing a second birth in the digital realm. Continuing from the prosumer paradigm, where people are allowed not only to surf a network obtaining content and information, but also (co-)create such elements themselves, the user-centred participation is increasing beyond the Web 2.0 as we know it.

The ITEA2 DiY Smart Experiences (DiYSE) project focus its effort to enable people to take control of their ever smarter surroundings, both at home and outdoor, evolving towards mass creativity in an open Internet-of-Things world.

It is clear that there is a lot of technologies already available allowing people to create and combine connected devices and services, but it is notable that most of the people involved must have a deep technical background. However when thinking about DiY, it is the 'everyday user' that should be enabled and involved in an intuitive and simple way. The interaction design between human and environment is a key factor in two different ways:

On one hand, services and devices have to be created, so it is important to determine which kind of development environment will be used. This document covers different kind of environment, and discuss about the skills needed to become a service/device creator.

On the other hand, created services should be used by non-experienced users. This document covers the user interfaces that are available at this moment in different devices that may be used to access these services.

The structure of the document is as follows: Section 2 analyses several kind of interfaces, from classic and tactile GUIs to multimodal interfaces (biometric or voice commanded). This section covers multidevice interaction and augmented reality too. Section 3 is related to D1.1 section 3 but from the user/developer point of view: how the users will use the elements of the DiYSE ecosystem (mobile devices and hardware/software platforms). Section 4 talks about modelling from several points of view: interfaces, users, devices and tasks. Section 5 is related interface creation: tools and experiences, mainly in adaptive interfaces. Section 6 presents how cooperative communities are working at this moment to provide new software and services, with special focus on those aspects that may be adopted into a do-it-yourself environment and methodology. The final section will provide a starting point for discussion on these topics.

2 User interface modes (kind of interface)

2.1 Classic GUI and WIMP interfaces

A GUI, or Graphical User Interface, is what at the moment of writing is 'the' way most people use to use computer systems. The GUI allows people to interact with a computer system withouth having to type in complex commands with a keyboard. First versions of the GUI were developed by Xerox PARC, the creators of the legendary 'sketchpad' appliation by Ivan Sutherland.

When research related to the GUI continued, the "WIMP" paradigmn emerged. WIMP, windows – icons – menu's – pointing device, is the paradigmn to which most mainstream computing applications are designed. Typical to a WIMP style interface are conventionalised icons or representations of reality, which have grown into standards over the years [185] .

Looking towards the future, a lot of research is being done into post-WIMP interfaces. This is happening because WIMP/GUI interfaces are often argued as being not 'natural' since they make too much abstraction of the real world [186] . Therefore it is important to look towards emerging interface modalities like NUI (natural user interfaces) and XUI (organic user interfaces) to bridge this gap between the real world and the digital world [187] .

2.2 Mobile interfaces

Mobile interface is the way that a mobil device permits to users interact with features and functionalities that it offers. The basic mobile interface are explained next:

- ***Touch screen***: Screen that permits to insert data information and orders through a direct touch over the screen [99] .
- ***Multitouch screen***: That interface represents an evolution of touch screens. Multi-touch allows the user to interact with the device by placing two or more fingers directly onto the surface of the screen [101] . The Multi-touch screen is a concept that is registered by Apple Inc [100]
- Keyboard ***QWERTY or a tipical pc keyboard adapted like mobile interface***
- ***Video Cam***
- ***Audio***
- ***Memory Slots***
- Other ***bottons*** such as direct access and trackball
- Communication interface such as Wi-Fi or Bluetooth

2.3 Multimodal

2.3.1 Biometric

In this chapter, we are exploring the different kind of user interfaces existing at the present time. The interfaces are one of the main parts of a DiY system. In this deliverable, we are dealing with Smart Spaces, so in this case not only is important

the user interface, but it is also important recognize the user. This is because if the Smart Space can recognize an individual, it is possible to change the user interface to provide a better experience to the user. In this situation, the biometrics is the first step in the user interaction process.

Biometrics is automated methods of recognizing a person, and is based on a **physiological** or **behavioural** characteristic [116] . Those characteristics must satisfy several requirements, which are:

- **Universality**: All the people, who are going to use the biometric interaction system, must have that biometric characteristic.
- **Distinctiveness**: The proposed characteristic must be different between two persons, so it is possible to differentiate those persons with the characteristic.
- **Permanence**: Over a period of time, the proposed characteristic must be sufficiently invariant.
- **Collectability**: It must be possible to measure the biometric characteristic.

In a real biometric system, those requirements are not enough. This is because there are other issues to take into account. Some of these issues are [117] :

- **Performance:** This issue refers to the achievable recognition accuracy and speed, resources required to achieve the desired recognition accuracy and speed, as well as the operational and environmental factors that affect the accuracy and speed.
- **Acceptability:** In the real applications, the culture constrains determine if one biometric characteristic is acceptable by the people or not. So it is important to consider the region where the biometric system is going to operate.
- **Circumvention:** which reflects how easily the system can be fooled using fraudulent.

The biometrics is used to identify a person. For this reason, the architecture of a biometric system has two modules. The first one is the enrolment module, which is responsible of the acquisition and store the user information. This module is the one that gives the base information in the future access of the user. The second module is the identification module. It is responsible of detect if a user is an accepted user or not.

The previous architecture is common to all biometric systems. Those systems also have three common physical components that are:

- **Sensor**: This is the device, which captures the biometric characteristics. The sensors must be adequate to each characteristic that is going to be measured.
- **Repository**: It is the database where the biometric information of each user is stored. It must be protected in a secure physical area, and must be encrypted and digitally signed.
- **Algorithms**: These are one of the most important components of the system. This is because the algorithms have to extract the biometric characteristics from the sensor information, and also, they are used to compare the users and finally determine the identification of the user.

As the first definition says, the biometrics is based on physiological and behavioural characteristics of the human beings. As it is possible to see, there are a lot of characteristics, but not all of them meet the previous mentioned requirements. So, with the different kind of biometric characteristics, is possible to define the following types of biometric modalities separated in physiological modalities and behavioural modalities:

- Physiological
  - Iris
  - Fingerprint
  - Hand geometry
  - Face
  - Voice
  - Retina
  - DNA
  - Odor
  - Ear Shape
- Behavioral
  - Signature
  - Keystroke
  - Voice
  - Gait

All of the previous modalities meet the biometric requirements, but not all those types are frequently used. In the following sections we are going to describe the different modalities of biometrics, separated in *common biometric modalities* and *other biometric modalities* [118] , which can be used as the previous step of the user interaction task [117] .

2.3.1.1 Common biometric modalities

2.3.1.1.1 Face

Since the 1960s, machine vision researchers have been developing automated methods for recognizing individuals via their facial characteristics. Despite the volumes of research, there are no agreed-upon methods for automated face recognition as there are for fingerprints.

Face recognition is a non-intrusive method, and facial images are probably the most common biometric characteristic used by humans to make a personal recognition. Multiple approaches have been developed for several years using low-resolution 2D images. Recent work in high-resolution 2D and 3D shows the potential to greatly improve face recognition accuracy. In all the cases, the system has to extract appropriate characteristics from the facial image in order to perform identification.

There are two major approaches to automated face recognition [120] :

- **Analytic**: In this case, geometrical models are used in order to identify faces. More specifically, discrete local (geometrical) features are extracted for retrieving and identifying faces. The location of the facial features is selected where are less susceptible to alteration such as eye sockets, cheekbones, sides of mouth. The position of these features, with respect to one another, determines the overall location of the face. Standard statistical pattern recognition techniques such as Hidden Markov Models may be applied on these measurements. Other approaches include active contour models (Snakes), wavelets, and knowledge- or rule-based techniques such as facial action coding system (FACS).

- **Holistic**: These methods are not based in face specific characteristics. These methods use “templates” or mathematical models to perform a global recognition of the face, not only local features. So in this case, a feature vector is used to represent the entire face. This approach includes Principal Component Analysis (eigenfaces), Artificial Neural Networks, linear discriminants and optical flow.

These systems also have difficulty in recognizing a face from images captured from two drastically different views and under different illumination conditions. In addition, it is questionable whether the face itself, without any contextual information, is a sufficient basis for recognizing a person from a large number of identities with an extremely high level of confidence[117] .

2.3.1.1.2 Fingerprint

Fingerprints have an uneven surface of ridges and valleys that form a unique pattern for each individual, and its formed during the first seven months of fetal development.

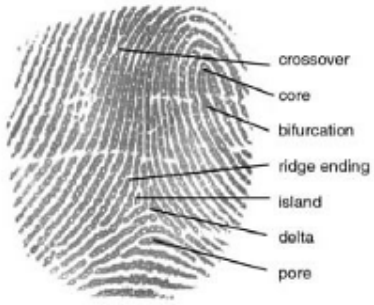

Figure 1. Fingerprint [115]

The fingerprint modality uses local features of the fingerprint to identify persons. Those features are called “minutiae”, and the characteristics that are search are: type, orientation, spatial frequency, curvature and position.

Nowadays, this modality is one of the most used (a lot of laptops have a sensor to identify the user) because the sensors that it requires are the cheaper ones. However, one problem with the current fingerprint recognition systems is that they require a large amount of computational resources, especially when operating in the

identification mode. Finally, fingerprints of a small fraction of the population may be unsuitable for automatic identification because of genetic factors, aging, environmental, or occupational reasons (e.g., manual workers may have a large number of cuts and bruises on their fingerprints that keep changing).

2.3.1.1.3 Iris

The iris is the colored part of an individual's eye. The iris modality is based on visible features inside the iris, i.e. rings, furrows, freckles or the corona. The iris is essentially formed between 8 months of age and 2 years, and remains stable through life, so it meets the stability characteristic. The complex iris texture carries very distinctive information useful for personal recognition (each iris has 266 unique spots vs. 13-60 for other biometrics).

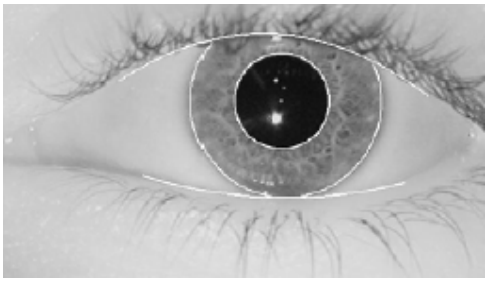

Figure 2. Iris recognition [118]

Each iris is distinctive and, like fingerprints, even the irises of identical twins are different. It is extremely difficult to surgically tamper the texture of the iris. Further, it is rather easy to detect artificial irises (e.g., designer contact lenses).

To obtain a good image of the iris, identification systems typically illuminate the iris with near infrared light, which can be observed by most cameras yet is not detectable by, nor can it cause injury to, humans. A common mistake is that iris recognition shines a laser on the eye to "scan" it.

2.3.1.1.4 Hand geometry

In this modality several morphological parameters of the hand are studied. Those parameters can be: width, height, finger length or distance between fingers. The technique is very simple, relatively easy to use, and inexpensive.

Figure 3. Hand geometry recognition process [118]

Although the basic hand geometric of one individual seems to be consistent through the time, some environmental factors (weather or user skin) can change this geometry, and it also may not be invariant during the growth period of children. In

addition, the hand shape and dimensions are not suitable for individual identification, because the differences between persons are very small.

2.3.1.2 Other biometric modalities

2.3.1.2.1 Gait

The aim of this modality is to identify an individual using a single camera located far from the user, detecting the way the user walks. As in face recognition, this technique is one that humans intuitively use to recognize someone.

Gait is a complex spatio-temporal biometrics, and it is based in the behaviour of the user. This behaviour may not remain invariant over the time (due to fluctuations in the body weight or other bio-physical problems).

Gait is easily simulated and it is not very distinctive, but is sufficiently discriminatory to allow verification in some low-security applications like a Smart Space.

2.3.1.2.2 Voice

The voice recognition is quite different from the speech recognition. While the speech recognition identifies the words that the user is “telling”, the voice recognition identifies the user that is talking. The voice is a combination of physiological and behavioural biometrics. This is because the voice is influenced by the physical structure of an individual’s vocal tract and the behavioural characteristics of this individual. These physiological characteristics of human speech are invariant for an individual, but the behavioural part of the speech of a person changes over time due to age, medical conditions, emotional state, etc.

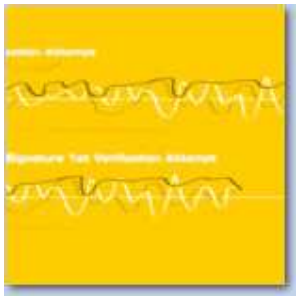

Figure 4. Voice signature [115]

The voice modality is one of the most popular recognition systems, but it is less accurate due to the dependence of the text used in the verification and in the environmental noise. Because of that, it may not be appropriate for large-scale identification, but in a smart environment application can be perfectly used.

2.3.1.2.3 Signature dynamics

This technique is based in the dynamics of a signature, not in what the signature looks like. When the user is signing, it measures the speed and pressure that are characteristic of the individual. Although signatures require contact with the writing instrument and an effort on the part of the user.

This method is based in the behaviour of the user, so it is potentially falsificable. It also changes over a period of time influenced by physical and emotional conditions.

Because of that, this modality can't be used in an application with high security requirements.

2.3.1.2.4 Keystroke

This biometric modality proposes to identify a person using the keystroke dynamics. This method supposes those dynamics characteristics in a person, so it is possible to identify it. The problem of this technique is that it is based in the behaviour of the user, so it is potentially emulable, and it is also easily monitored.

2.3.1.2.5 Retina

The retina is a light sensitive tissue lining the inner surface of the eye, composed by multiple blood vessels. The structure of the blood vessels in the retina is supposed to be a characteristic of each individual and each eye. It is the most secure modality, because it is complicated to change it or replicate that structure. The main problem of this technique is that it not only involves cooperation of the subject, but it requires a conscious effort on the part of the user.

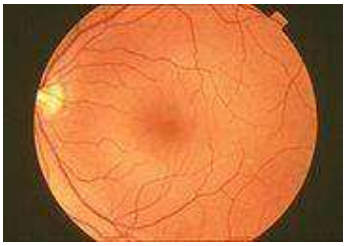

Figure 5. Retina image [119]

2.3.1.2.6 Thermogram

This modality is based on the measure of how heat is radiated by a human body. This radiation pattern is characteristic of an individual and it is possible to be captured using an infrared camera. The advantage of this technique is that it is not an invasive modality because it doesn't require contact, but the image acquisition is challenging in uncontrolled environments, where heat-emanating surfaces are present.

2.3.1.2.7 DNA (Deoxyribo Nucleic Acid)

There are no doubts that this modality can identify definitely a person. It is the most powerful modality in the biometrics field, but it is the most difficult one to implement. This is because nowadays it is impossible to develop a DNA based system that works in real-time, and convenient for the users. There is also a problem related to the easiness of steal a piece of DNA (for example from the user hair).

Odor: Each object exudes a characteristical odor, and particularly, the odor emitted by a human body is distinctive to a particular individual. So it is possible to identify persons using its odor, but it is not clear if the invariance in the body odor could be detected despite deodorant smells, and varying chemical composition of the surrounding environment.

2.3.2 Gesture interaction

The gestures are the most natural action to interact between human beings. Because of this fact, the gesture interaction enables humans to interact with the computers more naturally than using classical WIMP interfaces.

The gestures can broadly be classified in the following types [18] :

1. **Hand and arm gestures:** those are the most general gestures in all cultures. Examples of this gesture interaction can be: recognition of hand poses, sign languages, and entertainment applications (allowing children to play and interact in virtual environments).
2. **Head and face gestures:** some examples of those gestures are: nodding or shaking of head; direction of eye gaze; raising the eyebrows; opening the mouth to speak; winking; flaring the nostrils; and express emotions.
3. **Body gestures:** this gesture interaction involves the full body motion. Some examples of the usage of this interaction are: tracking movements of two people interacting outdoors; analyzing movements of a dancer for matching music and graphics; and recognizing human gaits for medical rehabilitation and athletic training.

The technologies used in the gesture interaction try to determine what kind of movements the humans are doing and interpret them in order to perform some action. In general, the gesture interaction systems can be divided into two main components:

1. **Features extraction**: This is the first step to perform in a gesture interaction system. This task target is to obtain the information from the human performed gesture and extract characteristics about the position of the body and body's parts.
2. **Gesture recognition:** After the parameters are computed and the features are extracted, the system must decide if the user is performing a meaningful gesture. In order to know if the gestures are valid, they need to be classified and interpreted. This interpretation generally is based on model and based on some grammar rules that reflect the internal syntax of gestural commands. The grammar may also encode the interaction of gestures with other communication modes such as speech, gaze, or facial expressions.

Some of the gesture interaction techniques have another component which is the **tracking** of the features [20] . After the features extraction, the position and possibly other attributes of the body must be tracked from frame to frame. This is done to distinguish a moving part of the body from the background and other moving objects, and to extract motion information for recognition of dynamic gestures.

In the following sections it is described the both of the main gesture interaction components.

2.3.2.1 Features Extraction

The first task whch needs a gesture interaction system is the features extraction. The aim of this component is to obtain information from the human body, like the position,

configuration (angles and rotations), and movement (velocities). With that is possible to extract characteristics which can infer the gesture made by the human.

In order to determine all these aspects, the human body needs to be sensed. This can be done either by using sensing devices attached to the user or using computer vision techniques. The sensing devices may be magnetic field trackers, instrumented (data) gloves, and body suits. Each sensing technology varies along several dimensions, including accuracy, resolution, latency, range of motion, user comfort, and cost [18] . Glove-based gestural interfaces typically require the user to wear a device and carry a load of cables connecting the device to a computer. This hinders the ease and naturalness of the user's interaction with the computer.

Figure 6. Glove based gestural interface Source: Vrealities

In the other hand, vision-based techniques can solve those problems, but those need to deal with other problems related to occlusion of parts of the user's body or illumination variances.

As a comparative example of those two techniques of features detection is hand movement detection. While tracking devices can detect fast and subtle movements, even if the user is moving, a vision-based system will at best get a general sense of the type of motion. Again, vision-based devices can handle properties such as texture and colour for analyzing a gesture, while tracking devices cannot.

In the following sections, it is described those two areas of research, giving special attention to the vision-based techniques, because with those techniques is possible to perform the most natural human computer gesture interaction.

2.3.2.1.1 Sensors based features extraction

These systems use sensors that are attached to the human body in order to get information about its movements. The sensors that are more commonly used are the **accelerometers**. These sensors turn acceleration (either lineal, angular or both) into an output signal that can be processed.

Huiyu Zhou revises the commercial systems based upon accelerometers, in the market there is a wide variety of such accelerators [38] . The price of these

equipments ranges from 1,200€ for an accelerometer with its reception equipment, to 56,000€ for a complete body monitorization equipment.

Figure 7. Accelerometer based gestural globe Source: AnthroTronix

One of the disadvantages of this method is that it is necessary to place complex equipment upon the person who is going to be monitored. Its advantages are that it does not suffer from occlusions, unlike the rest of the methods, and it always provides the position of all the points of interest of the human body.

2.3.2.1.2 Vision-based features extraction

As the introduction show, in human-human communication gestures are often performed using a variety of body parts (e.g., arms, eyebrows, legs, entire body, etc.). However, most researchers in computer vision use the term gesture interaction to refer exclusively to hand/arm gestures [19] . Because of that, the following sections, only concern to hand/arm gestures and body gestures also.

There is a first subdivision inside the vision-based features extraction techniques. Inside this division exist the techniques which use markers and those without markers.

The first approach to the problem of extracting the human body features are the systems based upon the use of markers. These small visual markers are placed upon the body that is going to be the object of the follow-up in order to capture the data.

Figure 8. Marker (red and green) based gestural interface Source: MIT Media Lab

As the human body is a highly articulate structure, these markers are constantly coming in and out from the occlusion areas. This is the reason for what the use of these small markers is more easy and fast to detect the features of the body. There are numerous commercial equipments, such as VICON, CODA, ReActor2, ELITE Biomech… Some of these devices are able to provide a modelling of the human body even in 1 ms.

While markers can be accurate, they place restrictions on clothing and require calibration, so they are not desirable in many applications.

Another approach to the problem is based upon the use of one or several cameras but without the markers in the human body. In these systems, the human gesture must be inferred from the images obtained through these cameras.

This approach makes it possible to establish a model of the human body, making up for the information loss provoked by the occlusion of several characteristics of the human body itself. This technique is still being developed and it still must face several unsolved problems. Besides, it must be said that the processing that a non marker-based vision system must carry out is more costly (in time) than the processing carried out by a marker or accelerometer-based system, but most of the existing systems in this field operate in real time.

As well as, there are no commercial systems based upon this follow-up technology. We will focus, mainly, upon this type of systems, because are the most natural interaction Systems between humans and computers (or smart devices).

Inside this field, there are two main approaches to extract features from the human body. Those approaches are differentiated by whether the system is based on an abstract model of the body or on knowledge of the appearance of the body in the image [20] :

- **Model based approach:** This approach uses previous knowledge about the structure of the human body and their cinematic and dynamic model. Those models are matched to the results of the pre-processing to determine the state of the tracked part. The model can be more or less elaborate, from a 3D model with 27 degrees of freedom (DOF) to a simple contour model of the hand. In addition to the model, how features in the image corresponding to the real body are produced, is required. This measurement model is needed in order to determine the state of the body model from the appearance of the body in the image. Continuously fitting the model to the body in the video frames is a process of tracking the complete state of the hand not just its position. This process is consequently called state based tracking. If the model contains a sufficient number of internal degrees of freedom, recognition of static gestures can be reduced to inspection of the state.

- **Appearance based approach:** The feature extraction is based on a representation learned from a large number of training images. As no explicit model exists, the degrees of freedom will not have to be specifically modelled. When only the appearance is known, differentiating between gestures is not as straight forward as with the model based approach, so this features

extraction will involve some statistical classifier. One example of this approach is the use of information about the colour, texture or the movement of the object; without trying to approach it to a previously established model.

It also, there exists other classification which divides the feature extraction in an another two approaches:

- **2D approach**: this research field is focused on the detection and extraction of features of the body in one plane. Such approach tries to obtain a follow-up of the human movement in a single plain, without extracting a model in three dimensions of the movement.
- **3D approach**: this field is focused on the extraction of features in the three coordinates.

In the following, it is going to describe more specifically the previous areas (and some research in those areas), starting by the model-based approach, continuing with the appearance based approach and finally describing the 3D approach.

2.3.2.1.2.1 Model based approach

In order to make up for the occlusions among the different parts of the human body, we can use our a priori knowledge about the human body, its proportions, its form, the turning capacity of the different joints… This is, more or less, the same behaviour that the human brain, because it can interpret images in movement using a previously learnt model, deducing the characteristics that might be hidden behind the image.

Wren et Al present a system (the pFinder) that models the human body by means of blobs [21] . Blob is a term that refers to a modelling that is not based upon defined contours, instead, each element (each blob) is defined by a Gaussian probability distribution g(x,y,U,V,W |average, covariance).

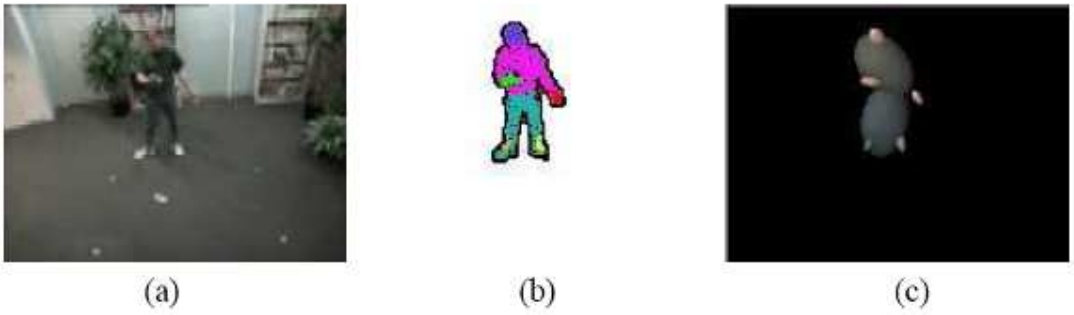

Figure 9. System presented by Wren et Al, based upon colour forms (Blobs) and statistic models.

The system recognizes the background of the image in order to extract the human body information. This background recognition and extraction it isn't trivial because the light changes, the occlusion of part of the background with objects (or with the human body). In order to avoid these problems, the Wren et Al. model each pixel in the background by means of a Gaussian distribution, using colour data. Through a distance check they obtain the points of the surface that have been hidden by a

person or by an object and those that belong to the background are applied an adjustment of the statistical model so that they are not affected by small illumination variations or changes in the background, calculating their new average and variance.

As it is said, the human body is modelled by blobs. In each frame, the system evaluates the data of the previous frame and it tries to obtain the new average measures and the covariance matrix for each of the blobs that make up the human body, which results in a set of superposed blobs in the region occupied by the human body.

In order to model the human body it is necessary to locate the head and the hands using a shape analysis. Once they are located, a blob is created upon them; and another two blobs are created, and corresponds with the human clothes.

Other authors, such as Niyogi and Andelson [22] propose to use contours (called *snakes*) to extract the human silhouettes. Those snakes can be deformed in time and space, so it can be adapted to a human body. The next figure shows how the snakes can model the gait of a human:

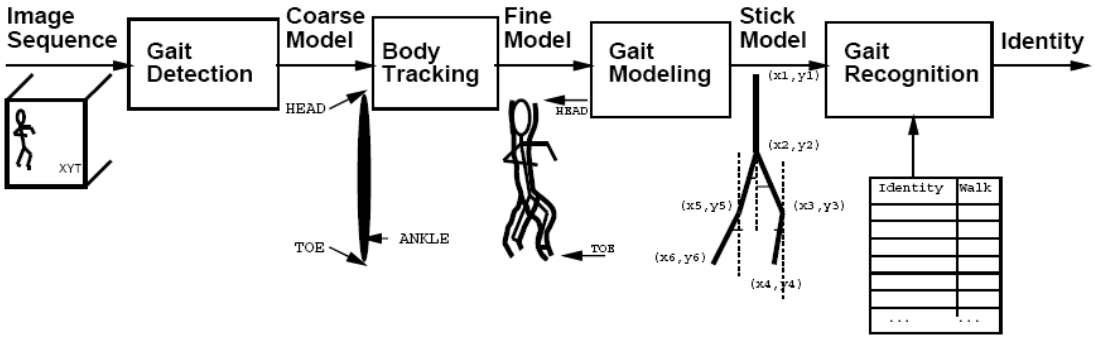

Figure 10. Modelling the walking movement proposed by Niyogi and Adelson.

The apply of snakes is possible when the system detects the presence of a human. For this purpose, the authors based the detection on the figures created by a person when he is walking. This phenomenon can be observed in the next image:

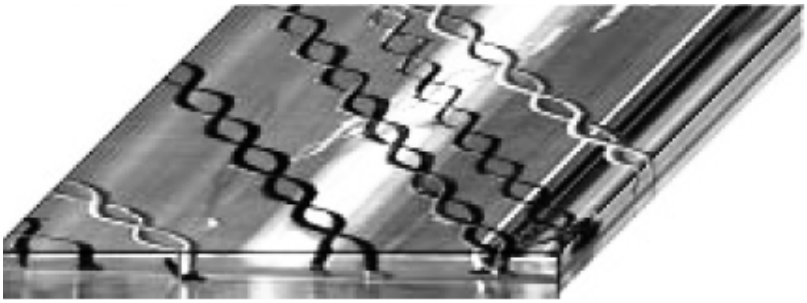

Figure 11. Trace left by several persons walking for some time

Another approach model based upon examples is the one proposed by A. Mohan et al. [24] , which uses hierarchic classifiers based upon SVM (Support Vector Machines).

The system is structured with four distinct example-based detectors that are trained to separately find the four components of the human body: the head, legs, left arm, and right arm. After ensuring that these components are present in the proper geometric configuration, a second example-based classifier combines the results of the component detectors to classify a pattern as either a “person” or a “nonperson”.

Figure 12. Body detector proposed by A. Mohan et al.

The algorithm is also more robust than the full-body person detection method in that it is capable of locating partially occluded views of people and people whose body parts have little contrast with the background.

Other authors propose to use neural networks in order to model out persons. Thus, they locate a human figure using deformable contours and then they extract a feature vector which is similar to a radial signature for several angles. By means of this feature vector and also using numerous training models it is possible to train a neuronal network that will be able to differentiate human silhouettes from those silhouettes that are not human and to obtain the position of the person.

A simpler approach, based upon the extraction of the contours and on their subsequent skeletisation, is the proposal that models out the points that are more distant from the centre of the silhouette.

2.3.2.1.2.2 Appareance based approach

The appearance-based approaches have used methods such as the Gaussian modelling, colour segmentation, Kalman filter, movement detection according to differences, active contours and analysis of the data from different cameras.

Many of these are used as initial segmentation methods in other types of approaches that are more advanced, or as low-level processing methods that offer additional information or support in the segmentation of a human body.

As the previous paragraph says, the segmentation of a human body in movement has been approached in many different ways. Thus, several authors model the background using Gaussian models, others study the variation of an image in space and time, another approach is based upon the study of the optical flow of stereoscopic images and then it tries to model the position of the human body using elements with the same velocity field. Methods based in temporal derivative, contour filters, can also be used in combination with other methods in order to facilitate the segmentation of the contours

In [26] proposes an stochastic human segmentation based on a static camera. In that work the authors created two models for each person. First one is based upon the colour of the person, and the second one is a presence model, based upon the probability that a person may appear in a particular place in the image. With this system, the authors were able to detect of multiple persons using a single camera.

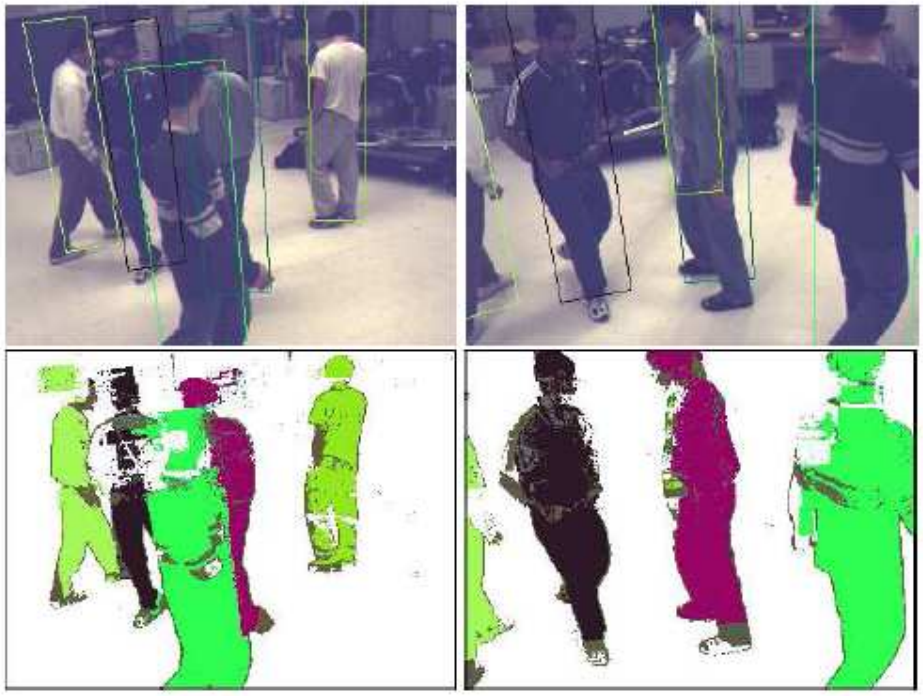

Figure 13. Segmentation of several persons proposed in [26]

Other authors propose the use of skin colour detection and segmentation. With this, it is possible to detect the position of the hands or head only using these models. One of the problems of colour segmentation is its variation due to the illumination changes, both in time and space (due to the movement of the person towards darker areas).

There is a lot of research in order to solve these problems. For example, [23] propose the use of SVM in order to adapt the colour model to the real scenario. Initially, given a gesture video sequence, a generic skin model is applied to the first couple of frames to automatically collect the training data. Then, an SVM classifier

based on active learning is used to identify the skin pixels. Finally, the results are improved by incorporating region segmentation.

Other appearance-based approaches are proposed. For example, Chen et al. [25] propose a new approach to the hand gesture recognition with the combination of statistical and syntactic analyses. The fundamental idea is to divide the recognition problem into two levels according to the hierarchical property of hand gestures.

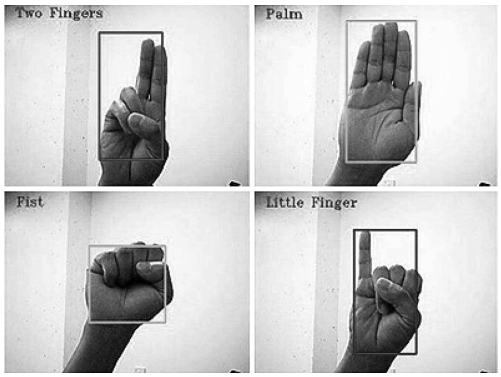

Figure 14. Hand detector proposed in [CHEN]

The lower level of the approach implements the posture detection with a statistical method based on Haar-like features and the AdaBoost learning algorithm. With this method, a group of hand postures can be detected in real time with high recognition accuracy.

Finally, in [27] propose a fast method for hand positioning and tracking. It combines KLT features and a learned foreground colour distribution to facilitate 2D position tracking from a monocular view.

Figure 15. Hand tracker proposed in [27]

The tracker's benefits lie in its speed, its robustness against background noise, and its ability to track objects that undergo arbitrary rotations and vast and rapid deformations.

2.3.2.1.2.3 3D approach

The different ways to model out a human body in three dimensions are based upon the following models:

- **Cinematic models**: these models take into account not only the joints, but also their position and velocity as well as the limitations of the angles that they can form.
- **Dynamic models**: evolution of the previous models that take into account the forces and the movements exerted upon the body in order to predict any possible changes in their condition.
- **Volumetric models:** These models are based upon the modelling of the human body using sets of ellipsoids or sets of ellipsoidal cylinders
- **Stick Figures**: Representation of the human model based upon an articulate skeleton.

N. Jojic presents a system that makes it possible to detect the position of a 3D human model using two stereoscopic cameras through a disparity map [29] .

Figure 16. 3D human model by [29]

This map gives us an idea of the dispersion of the pixels of similar features in the two stereoscopic images. This way, it is possible to calculate the depth coordinate (Z) for each zone of the image. Using Gaussian models formed by the coordinates X,Y,Z and using information about the colour of each part of the body, they can foresee the configuration of the three dimensional model that has formed the image. This way, they prevent occlusions from happening, as this 3D Gaussian model would associate them to the hidden parts of the image.

H. Sidenbladh et Al, propose a method to obtain a three-dimensional model of the human body using a monocular image [30] .

Figure 17. 3D human arm tracking in [30]

Thus, they propose a three-dimensional model that is projected by a camera model (pinhole) upon the silhouette that is going to be modelled and then, using an appearance model based upon the response of the member that is going to be modelled to several filters, finds the maximum probability model that adjusts to the aforementioned silhouette.

A cinematic approach based upon the skeleton is the one by C. Theobalt et Al. [31] . Their system proposes the use of several synchronized cameras that make it possible to extract a silhouette of the person. This silhouette is initially segmented by using the "Generalized Voronoi Diagram decomposition", which divides the silhouette into different regions. Once the system performs this division, it calculates the colour model, which will change from frame to frame, for the areas of hands and feet. Besides, with the information obtained from the silhouettes of different cameras, it provides a model based upon voxels (3D pixels), which is the frame in which the human model can be placed. Then, an offline module adjusts the model.

A similar approach is used by I. Mikic et Al [32] who use voxels too, in order to define the frame that contains the model of the body. The difference lies in the fact that they use a more complete modelling of the human body, based upon ellipsoids. Besides, the calculation of the longitudes of the model (size of arms, legs,…) is carried out automatically in the initial frames.

Apart from the information of the silhouette of the image, C. Theobalt [33] puts forward an additional approach using the optical flow on top of the silhouettes in order to carry out an analysis of the voxels.

2.3.2.2 Gesture Recognition

The recognition of the gesture is the final stage of a gesture interaction system. There are two kinds of gestures that can be detected: a) static gesture (pose) and b) dynamic gesture.

A **static gesture** is produced when the user adopts a certain pose. In this scenario, the gesture recognition can be address by template matching, standard pattern recognition techniques or neural networks.

When the user doesn't adopt a certain pose, but it moves a part of it's body, a **dynamic gesture** is done. In this case, the detection of the gesture involves the use

of other kind of tool such as Hidden Markov Models (HMM), Particle Filtering and Finite State Machines (FSM).

In [34] a HMM is proposed for various types of hand gesture recognition. A gesture spotting is proposed for the pre-processing stage of the gesture recognition. The gesture spotting algorithm divides the trajectory into real and meaningless segments. To construct a feature database, this approach uses a combined and weighted location, angle and velocity feature codes, and employs a *k*-means clustering algorithm for the HMM codebook. In the pre-processing stage it also uses two different procedures for hand localization and hand tracking. The hand location procedure detects hand candidate regions on the basis of skin-colour and motion. The hand tracking algorithm finds the centroids of the moving hand regions, connects them, and produces a hand trajectory.

In [35] a gesture recognition based on HMM is proposed.

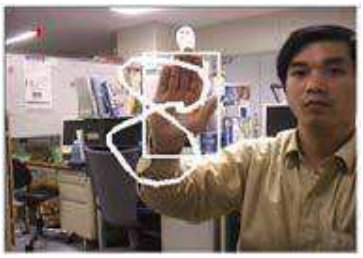

Figure 18. Example of a gesture detection [35]

The system consists of four modules: a real time hand tracking and extraction, feature extraction, hidden Markov model (HMM) training, and gesture recognition. First, the system applies a real-time hand tracking and extraction algorithm to trace the moving hand and extract the hand region. Then, it uses the Fourier descriptor (FD) to characterize spatial features and the motion analysis to characterize the temporal features. The spatial and temporal features of the input image sequence are combined as the feature vector. After having extracted the feature vectors, the system applies HMMs to recognize the input gesture. The gesture to be recognized is separately scored against different HMMs and the model with the highest score indicates the corresponding gesture.

The authors of [36] model the gesture as an ordered sequence of states in a spatio-temporal configuration space using a FSM approach. They propose the detection of a limited set of dynamic hand gestures. This involves extracting the temporal signature of the hand motion from the performed gesture. The concept of motion energy is used to estimate the dominant motion from an image sequence. To achieve the desired result, they introduce the concept of modelling the dynamic hand gesture using a finite state machine. The temporal signature is subsequently analyzed by the finite state machine to interpret automatically the performed gesture.

In [37] the authors propose a state based approach to gesture learning and recognition.

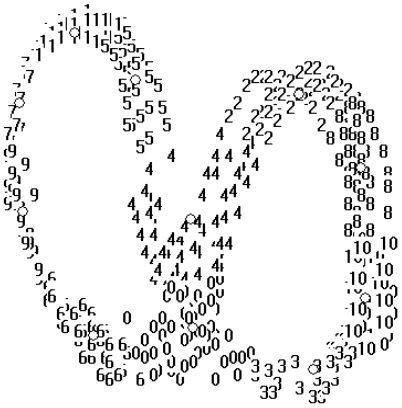

Figure 19. States of "infinity-figure" gesture [37]

Using Spatial Clustering and Temporal alignment, each gesture is defined to be an ordered sequence of states in spatial-temporal space. The 2D image positions of the centres of the head and both hands of the user are used as features; these are located by a colour based tracking method. From training data of a given gesture, the system first learns the spatial information and then groups the data into segments that are automatically aligned temporally. The temporal information is further integrated to build a Finite State Machine (FSM) recognizer. Each gesture has a FSM corresponding to it.

2.3.3 Mobile device gesture recognition

The source of innovations in mobile device user interfaces lies in combinations of input and output technologies that match the user's needs. In the mobile context, movement sensing - and haptic feedback as its counterpart - offers a new dimension to multimodal interactions. There are use cases where traditional interaction modalities are insufficient, for example when the device is placed in a pocket or a holster or if the user is wearing gloves. In these situations, the user cannot press or see buttons to interact with the device. Instead, small motion gestures can be used as a limited, but convenient, control modality. The movement of the device can be captured with a 3-axis accelerometer, and the resulting acceleration signal can be used to detect the movement patterns for controlling the device.

One of the main questions in the application of movement-based interface is how to distinguish gesture movements the user performs from those movements that are produced by various other activities while carrying and using the device. Reliability can be argued to be the most important challenge in developing a mobile device gesture interface.

In acceleration sensor-based gesture recognition, gestures are detected either from continuous stream or from discrete segments of sensor data. In detection from discrete segments, gesture start and end are marked explicitly, e.g. with a button, instead of a continuous flow of device movements. From the usability perspective, interaction without explicit marking is preferred in general, since it requires less attention from the user. However, continuous data streaming and execution of the gesture detection algorithm requires continuous data processing, which normally consumes battery.

The development in digital acceleration sensor technology enables the integration of programmable interrupt-based solutions that can operate with low current consumption. Such sensors generate interrupts when acceleration on a spatial axis is over or below a set threshold level. Hence movement detection algorithms, initiated by an exceeded threshold, can be implemented as event-based instead of continuously processing a stream of data. The processing load at the mobile device side is similarly reduced since the operating system is woken up less frequently. This development opens up new possibilities for practical application of the technology in mass products such as mobile phones.

Specifically, acceleration sensors have been applied in user-trainable and pre-trained machine learning-based gesture recognition systems [151] [152] . There are a lot of studies in the literature on gesture recognition from discrete segments [153] [154] . Free-form gesture recognition yet has a limitation, it requires an explicit marking of the gesture, e.g. with a button or a still part at the beginning and/or end of the gesture. In addition, gesture duration has to be longer to increase the recognition accuracy. Hence the interaction requires more user effort, and gesturing can be socially obtrusive. However, despite the possible obtrusiveness when applied in public places, free-form gestures also have a wide range of potential uses in other settings, such as games, home electronics control, etc., where the social acceptance does not limit the use of the modality. The social aspect, distinctively important in the mobile usage context, has been addressed, e.g., in [155] [156] [157] . It can be extrapolated based on the literature that, when performed with a mobile device such as a phone, smaller gestures are considered socially more acceptable than large [155] . Examples of possibly useful small-scale gestures include shaking the device, e.g. [158] , and swinging it from side to side [159] . However, both of these interaction methods can be considered quite noticeable regardless of scale. Shaking also raises a question, how many repetitions of the shake movement are required until a shake is recognised.

Simple accelerometer-based tilting control has been discussed in the literature in many studies during the years, e.g. [160] , but also recently, e.g. combining tilt and vibrotactile feedback [161] , scrolling, switching between landscape and portrait display orientations [162] . Tilting is another potentially unobtrusive, and very simple to implement, movement-based interaction modality to be applied in carefully selected use cases in mobile computing.

A minimalist extreme in hand gestures is tapping the mobile device, first introduced in [155] . Tapping requires only a small scale of device movement and can be performed by finger or palm. The technological benefit is that tapping can be relatively straightforwardly captured with a 3D accelerometer, since the resulting movement pattern has a distinguishable sharp spike form. The detection problem can be narrowed down by applying a small, predefined fixed set of movement patterns: tap events. The unique usability benefit of the tap interaction is that it is discreet and can be used if the mobile device is located in a pocket or a backpack, since explicit marking is not needed. Furthermore, the user is not required to hold the device or see the keyboard to interact. A good example of a use case where tapping is useful can be found from the Nokia 5500 phone [163] : when a text message arrives, the user has thirty seconds to tap the phone twice and the message will be read for the

user aloud. It is useful when the phone is in pocket or on belt, or the user is wearing gloves; the message can be read eyes-free without first taking the phone into the hand and opening the keypad lock. Furthermore, tapping can be used as an additional modality. For instance, phone music player commands such as play next or previous song can be controlled by tapping to either side of the phone, which is convenient when the device is worn on a belt or in a pocket. Again, the user does not have to take the phone, open keypad lock, and press a button to perform the control action.

2.3.4 Voice interaction

The voice interaction is the modality of human-computer interaction where the user gives commands to the computer using it's own voice. In this modality, an automated speech recognition is needed in order to detect the user command (or commands).

Specifically, Automatic Speech Recognition (ASR) is a part of Artificial Intelligence whose aim is to facilitate spoken communication between human beings and computers. Speech recognition is a technology that makes it possible to transform, automatically, an utterance taken from natural speech into a text that may correspond with it as faithfully as possible. It makes it possible for the system that is provided with it to "understand" or interpret the contents of an utterance regardless of the voice of the speaker.

The problem that the ASR system puts forward is that it is necessary to bring together a set of information from different sources of knowledge (acoustics, phonetics, lexicon, syntax, semantics and pragmatics), with unavoidable ambiguities, uncertainties and errors in order to obtain an acceptable interpretation of the acoustic message received.

The most basic classification that can be carried out with regards to speech recognition systems is the following one:

- **Discrete recognition systems.** In the case of these systems it is necessary for users to make a pause between one word and the next. This way, it is easier for the system to recognise the words.
- **Continuous recognition systems**. In which users can speak in a completely natural way without having to worry about introducing pauses. There are other divisions in order to classify these systems. We may distinguish between those systems that require a previous training in order to adapt to the voice of the user and those systems that are independent from the user; or systems that can only recognise commands and fixed phrases in grammar-based systems.

In the Speech Recognition area, there are several ways to recognize the person, and in the following sections we are going to describe the most successful techniques.

2.3.4.1 Comparing Templates or Patterns by using a programming technique called Dynamic Time Warping (DTW)

It consists in comparing the pattern that must be recognised (initially) with a series of templates or patterns that represent the units that must be recognised. The template is just a set of acoustic features that have been organised in time (sequence of

parameter vector or indexes of a centroid or codebook library), and the comparison of patterns includes a temporal, non-linear alignment as well as the measure of the distance. This technique, which is used to solve continual speech recognition problems, isolated and even independent of the speaker, to a certain extent, is known as DTW (Dynamic Time Warping).

2.3.4.2 Hidden Models of Markov (HMM)

The stochastic modelling of the speech signal solves the problem of the template alignment technique, providing the best results so far, both in the case of isolated speech recognition and in the case of continuous speech recognition and as far as the independence of the speaker is concerned.

This type of approach to the problem lies behind the philosophy of pattern comparison but it differs from it in the form in which the patterns are obtained, in the type of pattern, in the measurement of the distance and in the way the temporal alignment is used by applying the latter. At present, we use an non-linear alignment algorhytm (Dynamic Programming) known as Viterbi algorhytm, which is able to “align” the input vector sequence or the indexes of a codebook with the set of stochastic patterns (HMM) represented by the words in a dictionary, in the form of the probability of that sequence to be observed (generated) by the different Hidden Models of Markov.

2.3.4.3 Neural Networks (NN)

Neural networks are parallel information processing structures, formed by numerous simple nodes connected with each other through weights and grouped in different layers, from which the input and the output layer must be distinguished. Given their intrinsically non-linear nature, their classification capacity and, mainly, their capacity to learn a particular task using observation-objective pairs without carrying out any type of supposition about the underlying model, they have become one of the most attractive tools in order to solve the problem of speech recognition.

Nowadays, the results attained can be compared to those obtained through other classic methods such as the HMM. Yet, they have different problems or nuisances, such as: neither the layer structure, nor the necessary number of nodes for each problem are known a priori; it takes too much time to train them is on occasions and there is a possibility for them to remain "stuck" in local minimum rates of the cost functions used during the training of the network.

Besides, the speech signal requires methods with bidimensional processing capacity, in space and time; and the neural networks on themselves are only provided with space processing capacity. This makes it necessary to combine the Dynamic Programming techniques and the HMM with these networks, which makes it possible to model the time variable, making it possible to carry out not only very adequate classifications of the input that enters the network, but also the segmentation of the input signal. Yet, other solutions incorporating some type of memory to the networks have been proposed, but this makes it difficult, to a great extent, to carry out an analysis of these networks given their non-linear character.

2.3.5 Gaze

Gaze is defined as the direction to which the eyes are pointing in space and it is used as indicator of user attention and also as a user interaction system. In eye tracking systems, computer vision techniques are used to find the eyes in the image and then determine the gaze.

The eye trackers are divided in two main groups: wearable and non-wearable systems. The wearable systems, are often more intrusive than the non-wearables, but the accuracy is higher than those. On the other side, non-wearable eye trackers (or remote trackers) had restrictions related to the users' movements. It also has problems with the speed of head motion, so there is a risk of frequent track losses. Moreover a remote eye tracker is not suitable (in general) for non-desktop situations such as example interaction in "ubiquitous computing" scenarios.

There are several issues in developing eye-tracking systems: intrusiveness, speed, robustness, and accuracy. The type of hardware and algorithms necessary, however, depend highly on the level of analysis desired and the application of this analysis.

There is a lot of research in this field. One of the research lines is focused on improving the accuracy of the eye tracking systems in non-wearable trackers. For example, Wang et al. propose using a high-resolution image to improve overall accuracy [121] . Other line of research is in the automotive area. One example of research in this area is the one developed by [122] . In that work, the authors proposed to monitor driver visual attention using a single, non-wearable camera, which was placed on a car's dashboard. Other research lines are centered in the user interaction with a system. In this case, we can mention the use of gaze as a pointer [124] and the use of gaze in affective interfaces, using the eye information for infer it's cognitive state [123] .

2.4 Multi-device interaction

While the development of interfaces is most often still based on the one *device - one user* idea, our typical environment is already providing an abundance of devices with associated interfaces. Developers of mobile services were the first to notice this. Typical services were adapted from web-sites providing similar services. Taking the constraints of mobile devices into account – screen size, lack of proper keyboard, no mouse, restricted computational power and power consumption issues – was not as easy as just downscaling the graphical interface. Adding to that the complexity of managing a dynamic service provided to the web and a plethora of mobile devices, each with their own peculiarities, made the task sheer impossible. Current environments add even an extra dimension by providing "affordances" – devices and components in the environment that offer interaction capabilities.

We can distinguish three ways to define multi-device interaction, which partly overlap:

1. The interaction with a service through a variety of clients (terminal devices), each with different user interface capabilities. Interaction typically takes place through one device at the time. The main challenges are related to catering for

this variety of client capabilities (including modalities) and properly implementing session transfer between devices.

2. The interaction with a service through multiple devices simultaneously, using the interaction capabilities of provided by the devices best suitable for the task at hand. The devices are typically operated by one user, who uses for example the graphical display of one device, while inputting text via another. The main challenges involved in this concept are the selection of the most suitable interaction devices around the user, the presentation of a comprehensive UI through these devices using their interaction capabilities, and splitting the interaction and event flows between the devices involved.
3. Interaction with a service by one or multiple users using (possibly, but not necessarily similar) multiple personal devices to interact with the service simultaneously. A typical example of such interaction occurs in gaming, or any other situation where people are cooperating watching a common screen or view. A special case arises, when one person uses several devices simultaneously to accomplish a task (e.g. exchange data). The main challenges here are related to managing multiple users and their interaction (e.g. multiple cursors).

To capture all of these three viewpoints, we might define multi-device interaction as:

Multi-device interaction is the concept where one or more users utilise the interaction capabilities of a variety of devices (simultaneously or consecutively) to improve the use experience of an interactive service.

A common challenge to all three views is that the supporting architecture must be able to cope with a variety of modalities, abstract the user interface, communicate interaction capabilities and cater for ad-hoc splitting and adaptation of the UI for the devices at hand.

Ad 1) When HTML was first conceived, it was intended as a flexible means to present textual information augmented with links, pictures, etc. Scaling to the actual screen or window size was taken care of by the viewing client browser. In search for more attractive and better controllable views, designers started to use enhancements like frames and features like tables in different ways then originally intended. Also different technologies running via plug-ins, e.g. Java or Flash, were used for this purpose. While producing more attractive web pages and UI's, the scalability originally offered by HTML was lost. For most computer applications, all having approximately a same sized screen, this sacrifice seems justified, but it backfired at the advent of mobile technologies. When internet services were provided on these mobile devices, with limited computational and graphical abilities and slow data connections, the typical web page would not scale down anymore. Special HTML-style solutions were defined for the mobile devices, like cHTML and WML (WAP), basically simplified HTML, but achieved only limited success. It proved to be particularly difficult for the service providers to maintain traditional web services and mobile services – often different versions for different devices – and provide

consistent content for all of the services. Also session transfer from one device to another was cumbersome.

Solving the issue of managing web and mobile services was approached in two different ways; 1) Automatic conversion of the original web services for viewing in mobile devices and 2) Modelling services on an abstract level and generating presentations for various clients from this common source. The first approach yielded usually very poor usability, as navigating through frames and tables required knowledge of the web page structure and embedded media scaled poorly or not at all (e.g. clickable images). The second did not catch on very well either, in spite of efforts by Paterno on task modelling, RedWhale (XIML), the UsiXML and UIML consortia and various tool builders (see also chapter 4). Eventually it seems that mobile phones have matured to view the web pages in their original form, just panning the view over the page to accommodate for the limited screen size. The Opera browser is a good example of this effort to adjust web pages for mobile browsing.

In addition to catering for mobile devices, some effort was also put in extending web browsing to voice interfaces. VoiceXML allowed for the easy definition of voice driven services. Additionally various browsers added “read aloud” or voice control to their browsers. This was a great help for disabled users, but only of limited use for the average use. Phone-based services were limited to implementations of ticket reservation of e-mail access and were not used as an alternative to web browsing.

Ad 2) This definition is closely related to some aspects of multimodal interaction. Whereas multimodality is referring to the use of various modalities, either to be understood as *human senses and faculties* or as *I/O channels* of a computer, multi-device interaction refers to the use of various *devices* for this purpose. Clearly these devices implement such I/O channels and people use their senses and faculties to perform the interaction.

Historically multi-device interaction can be traced back to the early days of personal computing, when the WIMP (Windows, Icons, Menus, Pointers) paradigm developed. The computer consisted (consists) of a computing part, a display and a mouse. The latter two are dedicated I/O devices and we could thus speak about multi-device interaction. Although in the current understanding of multi-device interaction we emphasise the ad-hoc nature of the coupling of the devices and the wider use of the device’s UI capabilities beyond their initial (device-specific) purpose, we can utilise much from the pioneering work in this early stage. For example the use of an UIMS (User Interface Management System) was proposed already in the 1980’s, it catered for a variety of user interface technologies. The advent of WIMP has since limited UI devices to the well-known mice-like devices and displays.

An essential part of combining devices’ UI capabilities is advertising these abilities for inclusion in the multi-device interaction. Various standards have arisen, mostly tackling the connection of UI devices to a computer (USB-HID) or the description of graphical rendering capabilities for hand-held devices (CC/PP – UAProf).

Combining the devices to cooperate to form the user interface used for interaction can either be done automatically or by user intervention. Automatic combination requires intelligence to decide which devices to involve depending on user preferences, context information, device capabilities, etc. Manual combination needs action from the user, either by selecting the devices from a presented list or by means of physical browsing techniques (see next section).

When the devices provide capabilities beyond the usual WIMP paradigm, multimodal challenges need to be faced, like synchronising multimodal input. An example is the intuitive combination of pointing actions with speech, like first done in the Put That There prototype by Bolt et al.

Ad 3) Multi-user environments raise new challenges altogether. The problem of several people working on a task mediated by computers has been worked on for a long time, for example in CSCW – Computer Supported Cooperative Work. The research is often focussing on support for local and remote cooperation involving e.g. shared displays, digital whiteboards, video conferencing, telepresence, instant messaging, wikis, blogs, etc. More recent phenomena involve “communities” in this process like in crowd sourcing, open source development and social media. This research provides input to our multi-device research through its evaluation of the use of common displays, telepresence and multiple users working on the same documents.

Another background for this type of multi-device interaction is found from gaming. Multi-player games using a joint screen or a virtual world for interaction abound both at home and on the internet. The “devices” used for these games are often game controllers or the WIMP devices connected to the PC, and the games have been specifically designed for these controllers. The challenge for our multi-device paradigm is to implement similar control behaviour using the UIs of devices originally intended for other purposes. In case of gaming, reaction speed might be a serious challenge.

The special case, where one user uses several devices to accomplish a task, technically doesn’t differ much from the multi-user case. A typical task is to transfer data from one device to another. The user will assume the role of provider and consumer alternately, seen by the system as two distinct users.

The prototypes to be developed in the DIYSE project will often need to address issues related to multi-device interaction, even if they are not always recognised as such. The challenge of the DIYSE project is to find generalised means to facilitate multi-device interaction, such as:

- Basic ad-hoc connectivity between the devices
- User interface resource capability description, publishing and management
- Interaction modelling
- User and context modelling and sensing
- Device coupling
- Smart adaptation, input fusion and output mapping algorithms

- Multi-user support

2.5 Tactile interaction and augmented reality

2.5.1 Tactile interaction

In a world which is becoming more pervasive, tactile interacation and interfaces will play an important role. Tactile interaction can be interpreted in many ways, but the major commonality between all interpreations is the fact that people get some kind of tactile feedback from a system. This can be very ‘passive’ feedback, like a certain surface texture on an object or can be ‘active’ when a system gives a certain type of haptic feedback through vibriation elements. [188]

Using tactile or tangible interactions, creating the link between the digital world and the real world gets a whole new meaning. [189] A lot of user interface development is moving away from the traditional GUI and WIMP paradigms in order to search for ways to incorporate computing systems in people’s lives in a more “humane” or “natural way”.

2.5.2 Augmented reality (UAH)

The Augmented Reality (AR) is a term for a live direct or indirect view of a physical real-world environment. An augmented reality system generates a composite view for the user [85] . It is a combination of the real scene viewed by the user and a virtual scene generated by the computer that augments the scene with additional information. In this way, with the help of advanced AR technology [80] , the information about the surrounding real world of the users become interactive and digitally usable. The future of AR is related to the technology evolution and its developed costs.

Some authors think this kind of reality is a part of a virtuality continuum, in other words, the mixture of classes of objects presented in any particular display situation, where real environments are shown at one end of the continuum, and virtual environments at the opposite extremum [79] .

Figure 20. Simplified representation of a "virtuality continuum" [79]

A typical augmented reality system is made up the following components:

- Monitor display where it shows the result of real world and virtual information.
- Camera lens that takes the real information.
- Software to process real data and transform them into augmented reality.
- Markers such as GPS, paper symbols or other technologies that put the system in the correct position.

2.5.2.1 Application domains

Augmented Reality offers countless new chances for interaction that are present in many and various domains, such as architecture, education, art, medicine, design, robotics, military training, virtual communities or entertainment.

2.5.2.1.1 Entertainment

The entertainment domain brings together a lot of final users, so it is understandable that all video-game companies are interesting in AR since it can provide many new possibilities for the way play. For example, "Can You See Me Now?" [68] By Blast Theory is an on-line video game of chase through the streets where players start in random locations of a city, carrying a laptop and are connected to a GPS receiver. ARQuake Project [70] is other video-game example that moves the original Quake to outside. And a lot of initiatives in this sense are available on the www [69] .

2.5.2.1.2 Support to complex tasks

Complex tasks can be simplified by addition of virtual information on the visual field. Hence, some tasks such as repair and assembly, surgery, etc, can take advantage of the possibilities that AR offers. An clearly example of this application can be see in the BMW laboratories, where the mechanic employs RA devices on car repairs [71] .

2.5.2.1.3 Mixed technology

AR is easy to combine with other technologies such as QR codes that are being apply on publicity environments [72] [74] , and it is integrated with a lot of system platforms like software applications [73] [75] .

2.5.2.1.4 Future applications

The future of AR is related to the technology evolution and its developed costs. In this way, AR can be evoluted to holographic displays [76] , holodecks [77] , or virtual GPS [78] .

2.5.2.2 Toolset and communities

- ARToolKit is a Cross-platform Library for the creation of augmented reality applications [81] .

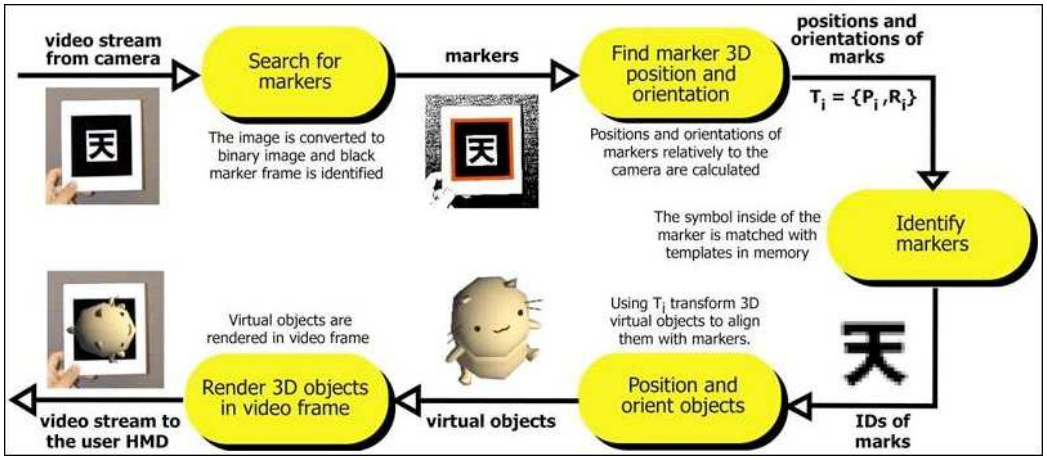

Figure 21. The ARToolKit tracking works (extracted from [82])

- ATOMIC Authoring Tool is a Cross-platform Authoring Tool software, for Augmented Reality Applications, which is a Front end for the ARToolKit library. Was developed for non-programmers, to create small and simple, Augmented Reality applications, released under the GNU GPL License [83] .

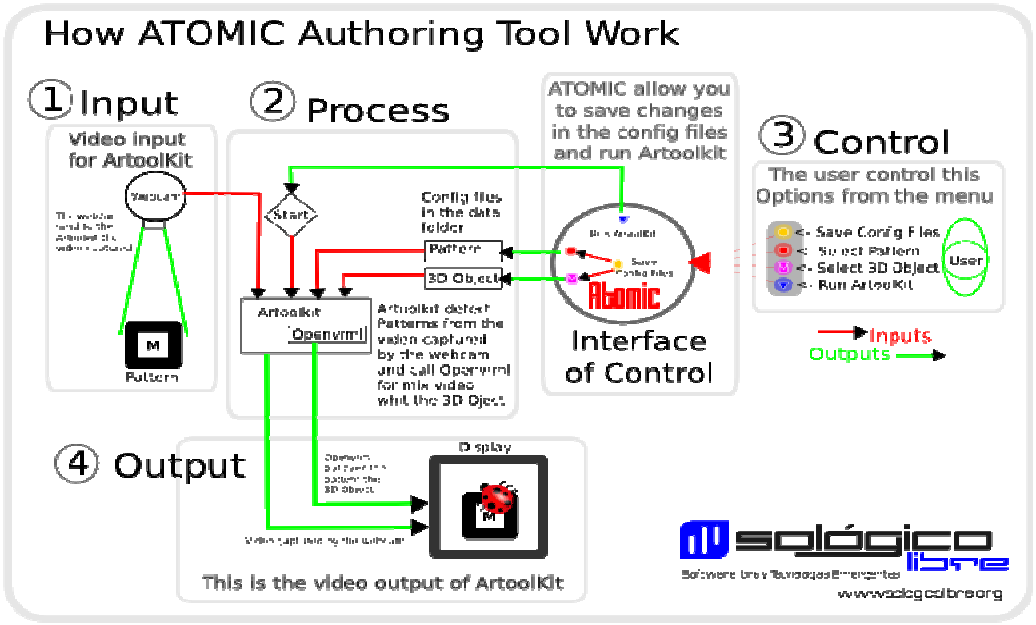

Figure 22. How Work ATOMIC (extracted from [84])

- PTAM (Parallel Tracking and Mapping) is a camera tracking system for augmented reality [87]
- ISMAR (International Symposium on Mixed and Augmented Reality) is the premier international meeting place for the Mixed and Augmented Reality research community [86] .

3 Toolsets

3.1 Introduction

3.1.1 What is a toolkit?

The term 'toolkit' is used to refer to a broad concept with a multitude of possible applications. However, given the current frame of DIY and users, we will focus here on the specifically relevant concept of **user toolkits**.

A user toolkit is *'a set of user-friendly design tools that allow trial-and-error experimentation processes and deliver immediate simulated feedback on the outcome of design ideas' [126]* . User toolkits are designed for 'do it yourself': with their basic product and building blocks for further personalisation, they invite the user to create the 'perfectly fit' product himself.

The toolkit concept is based on the idea to divide product development into sub-tasks, which need either information from customers or manufacturers. This 'task division' enables companies to address a wide range of consumers, and to avoid product failures, as customers can adapt the product to their own needs [127] [131] .

> *"The architecture of the humble pizza illustrates how this can be done. In the case of the pizza, many aspects of the design, such as the design of the dough and the sauce, have been made standard, and user choice has been restricted to a single task only – design of toppings. In other words, all need-related information that is unique to a given user has been linked to the toppings-design task only."[131]* .

Franke and Schreier [127] discern two different toolkit approaches: first, the **high-end toolkit**, which offers a large 'solution space' to users; and second, the **low-end toolkit**, which offers little 'design freedom'. The authors note that *'not all users need radically new solutions'* (2002). However, users can implement little functional improvements themselves, thus harvesting the social benefits of individualization and the psychological benefits of being the creator[127] .

Within the ICT sector, the concept of toolkits emerges in the 1980s, within the field of custom integrated circuit (IC) design and production.

3.1.2 Why use a toolkit?

For companies, it is interesting to be able to design products and services in such manner that users can contribute to them. Companies thus no longer need to anticipate all user needs, but can instead facilitate user-development through the means of a 'toolkit' approach. With the toolkit, companies can adhere to the heterogeneous needs of a wide range of customers [125] [128] . In other words, users get a chance to *'serve themselves'* [128] .

This approach is successful in open source software, where skilled users can adapt the 'code base', and in the gaming business, where a toolkit can range from a 'level' to a 'character builder' [129] . The companies' role thus shift from a creator to an '*enabler for users to develop new products and services themselves' [125]* .

> *"The basic idea behind toolkits is to allow individual customers to modify standard products to their individual needs. Hence, the underlying assumption is that these*

needs are highly heterogeneous (leaving many customers dissatisfied with standardized products)." [129] .

As *'customers will need tomorrow what leading-edge customers desire today'* [130] , it is interesting for companies to see which creative and innovative solutions from such lead users could be relevant for other users as well. There is nevertheless a risk that the user-generated solutions could come in conflict with the corporate branding strategies and/or intended image [128] .

Winning over consumers creates an opening to gather them into communities. This could attract users, at least those willing to be a part of it, and willing to contribute free time and effort, driven through the urge to gain status and/or resolve problems [125] . Dodgson et al [125] conclude that *'firms able to harness these collective efforts will gain competitive advantage.'*

Therefore, it could be valuable for companies to implement the toolkit concept, even if the market share that could exploit the full potential is rather small [128] . The toolkit could nevertheless function as: *'a crèche for interested but inexperienced users who could evolve into leading-edge users over time'* [129] .

[131] is convinced that this toolkit-approach will be taken up in most markets dealing with heterogeneous consumer needs. This uptake could be illustrated with examples of online mass customisation toolkits from corporations such as Dell (PCs), Mini Cooper (cars), Nike (sneakers) or IKEA (kitchens) [126] . There are also examples on the business-to-business (B2B) market (e.g. custom food products from Nestle Food Services) [127] .

Also for end-users, who get to chance to customize a product to their wishes, the toolkit can be beneficial. [128] state (based on their research on the toolkit approach of the open source software Apache) that users who adjusted the standard product showed a higher satisfaction level than those who did not. Even non-innovative users can profit from this toolkit-approach to improve the product for themselves. The toolkit enables them to adopt solutions from skilled users. This of course requires les skills and efforts than having to develop a solution on their own.

3.1.3 How does a toolkit work?

A good toolkit consists of five elements, which make sure that users have the information they need to carry out their adjusting tasks effectively. The added design costs should be less than added benefits.

The five basic functions of an effective toolkit [131] are enabling users to complete trail-and-error learning cycles; offering users a 'solution space' that covers the users design needs; enabling users to use them with their habitual design language and skills; containing libraries of common used modules; and ensuring that the custom products and services created can be produced on the manufacturers equipment.

We elaborate on each of those functions:

A) Toolkit enables users to complete trial-and-error learning cycles

The first characteristic relates to 'learning by doing', and addresses the 'trail-and-error' process of creating something that fits one's needs, until the moment it is used

and it becomes clear that it does not quite fit the purpose in mind. This uncovers the real needs, from where the process can restart [131] .

B) Toolkit offers users a 'solution space' that covers the users design needs

Economical production of custom products and services is only achievable

when a custom design falls within the pre-existing capability and degrees of freedom built into a given manufacturer's production system. This is called the "solution space" offered by that system. A solution space may vary from very large to small, and if the output of a toolkit is tied to a particular production system, the design freedom that a toolkit can offer a user will be accordingly large or small [131] .

C) Toolkit can be used with users' habitual design language and skills

The third characteristic relates to the 'user-friendliness', on which the effectiveness and the success of the toolkit depend. Users should not need to engage in a lot of additional training to be able to use it, as the toolkit should enable users to draw on their existing skills and their habitual design language [131] . Ideal would be a code base open to a wide range of programming languages in the 'high-end' toolkits, and a 'drag-and-drop' interface in the 'low-end' toolkits. This will be appealing to a broad audience.

D) Toolkit contains libraries of common used modules

Custom designs are seldom novel in all their parts. Therefore, libraries of

standard modules that will frequently be useful elements in custom designs are a valuable part of a toolkit for user innovation. Provision of such standard modules enables users to focus their creative work on those aspects of the design that are truly novel [131] .

E) Toolkit ensures that the customized products and services can be produced on the manufacturers equipment

Finally, the "language" of a toolkit for user innovation must be convertible

into the "language" of the intended production system at the conclusion of the user design work. If this is not so, then the entire purpose of the toolkit is lost – because a manufacturer essentially has to do the design over again [131] .

3.2 Mobile devices

3.2.1 Entire platform

3.2.1.1 Apple Apple iPhone

The iPhone is a smart phone designed and marketed by Apple Inc. oriented to Internet and multimedia experience. iPhone comprises device, OS and a collection of software applications, and it combines a Smartphone, communications interfaces and hardware human interfaces in a small space. The iPhone runs over an operating system know as iPhone OS, based on Mac OS X, that includes Core Animation software and Open GL API components, which is responsible for the interface motion Graphics. iPhone OS user interface is based on the concept of direct manipulation using multitouch screen and sensors such as accelerometer.

iPhone supports Apple applications, as well as from third-party development through the SDK that Apple provides to developers. These software development kit is free only for University enviroments, and needs a Mac OS X computer and the Apple's suite of development tools Xcode, that provide support for project management, code editing, building executables, source-level debugging, source-code repository Management and performance tuning.

SDK is broken down into the following framework sets:

- Cocoa Touch. Includes multi-touch events and controls, accelerometer, localization an camera support.
- Media. Open AL, Open GL, audio/video/image controlls, Core Animation and Quartz
- Core Services. Networking, Core Location, Threads and embebed SQLite BBDD.
- OS X Kernel. Supports TCP/IP, sockets, file system, security and Power Management.

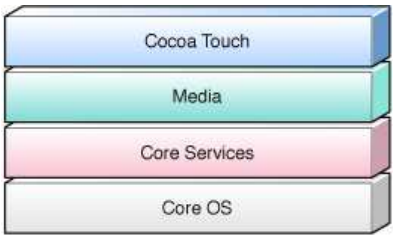

Figure 23. iPhone SDK structure.

The focus of application development is the Xcode application. Xcode is an integrated development environment (IDE) that manages all of the information associated with application, including the source files, build settings, and rules needed to put all of the pieces together.

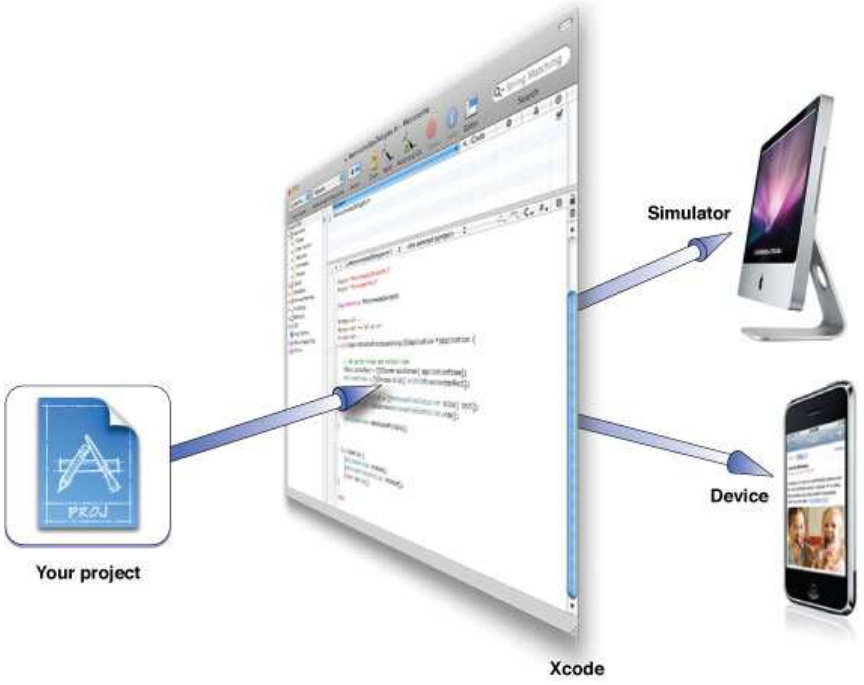

Figure 24. Xcode Interface.

Xcode comes with an advanced text editor, documentation and research assistant. When we build an application in Xcode, we have a choice of building it for iPhone simulator or for a device. The simulator provides a local environment for testing applications to make sure they behave essentially the way you want. After you are satisfied with your application's basic behaviour, you can tell Xcode to build it and run it on an iPhone or iPod touch connected to your computer. Running on a device provides the ultimate test environment, and Xcode lets you attach the built-in debugger to the code running on the device. Two important tools that Xcode provides are **Interface Builder**, to assemble your application's user interface visually, and **Instruments**, to analyze the performance of your iPhone applications while running in the simulator or on a device.

3.2.2 OS platform

3.2.2.1 Symbian S60

Symbian is an operating system designed for Mobile devices and smartphones developed by Symbian Ltd. In 2008, Nokia acquired Symbian Ltd. This operating system includes libraries, an user interface, and a development environment with common tools for creating open source applications.

The most of aplications can be download from Symbian Horizont and it can be developed in many languages, such as JavaME, Flash Lite, Python, Ruby, .NET, Symbian C++ and some web technologies. Each language needs a different development tool and not all of these tools are open licensed.

3.2.2.1.1 C++ Developers

For C++ developers is necessary to have ADT and SDK packages. **ADT (Application Development Toolkit) 1.0** is used with the SDK and contains all the tools necessary for application development. The Carbide.c++ IDE is the principle tool used for C++ development and is built on the Eclipse tool framework. Carbide plug-ins includes build, debug, static analysis, dynamic analysis, and a variety of other specialized tools and utilities. Carbide.c++ supports Symbian Build System v1 and v2 (Raptor). Carbide.c++ allows the developer to edit, build, and debug application on the emulator and on phone targets. The ADT also includes Carbide.ui Theme Edition. Carbide.ui allows the developer to create themes for devices and provides access to the more than 1,000 customizable theme elements in the UI.

Application Development SDK- (from Nokia). The Application Development SDK available is the Edition SDK from Nokia This will change with future Application Development SDKs, but until the Symbian Foundation creates its own SDK, the S60 5th Edition SDK from Nokia is being used as the SDK for Symbian^1. This SDK enables application development in Symbian C++, Open C/C++, Java and Web technologies. The SDK includes the all key resources needed for application development (documentation, API reference, examples, and an emulator), and is used with the Application Development Toolkit (ADT) for IDE support.

3.2.2.1.2 Java developers

Symbian Foundation does not have the full Java package for the time being because of licensing issues, only platform APIs and stub implementations for them are provided. The full implementation is available directly from Nokia for those S60 licensees that have relevant licenses from the appropriate 3rd party Java technology vendors

3.2.2.1.3 Flash Lite developers

Adobe supplies the industry-standard Flash Professional CS4 authoring tool for use in creating Flash/Flash Lite content, allowing designers to prepare anything from simple animations through to complex multi-component ActionScript-driven applications

3.2.2.1.4 Web technologies

Web Runtime consists of several components that allow installation and execution of Widgets. Widgets are essentially applications written in HTML/JavaScript/CSS, with JavaScript based access to some device APIs. Widgets execute inside the web browser and are very similar to web/AJAX applications we see on the web.

Widget development is supported by an growing set of plugins for major development environments such as Eclipse-based Aptana Studio, Microsoft Visual Studio and Adobe Dreamweaver.

3.2.2.1.5 Python developers

Python applications ("scripts") are simply text files containing code written in the Python programming language. The scripts can be written in any text editor and can be run either from within the Python Interactive Shell application on a device or the emulator, or as standalone applications on mobile device. The development

environment consists of the Python reference documentation, the Application packager for making your scripts into stand-alone applications and the phone SIS files. Both are provided for Windows and Linux/Mac

3.2.2.1.6 Ruby developers

Ruby is a dynamic programming language with a complex but expressive grammar and a core class library with a rich and powerful API. Symbian Ruby brings the Ruby programming language to the Symbian platform under the terms of the Ruby GPL License. It is a port of the official Ruby code base and should run most of the existing Ruby scripts. Symbian Ruby source code is hosted at the official Ruby central repository, so it is always in synch with the latest Ruby Core releases.

3.2.2.1.7 .NET developers

The .NET Compact Framework is a subset of the full .NET platform designed by Microsoft for mobile applications on Windows Mobile"Red Five Labs" provides Net60, the .NET Compact Framework on Symbian. Microsoft originally designed and developed the .NET platform with support for multiple languages and operating systems. Red Five Labs implemented the Common Language Infrastructure (commonly referred to as .NET) on the Symbian platform with support for C++, Java and .Net.

3.2.2.2 Maemo

Maemo is a software platform that is mostly based on open source code and strong mobile devices such as the Nokia N900 or the Nokia N810 Internet Tablet. The Maemo platform is the core stack for this kind of devices and it has been developed by Nokia in collaboration with many open source projects such as the Linux kernel, Debian, GNOME, and many more. The Maemo platform offers a set of development tools that allows developers or non-technical users (Maemo SDK) [49] to create applications on top of this platform. Furthermore, exist an open community developing software [50] around its that has over 20000 members that contribute to several hundreds projects in the Maemo Garage (place for working on various software projects related to the Maemo platform) [51] .

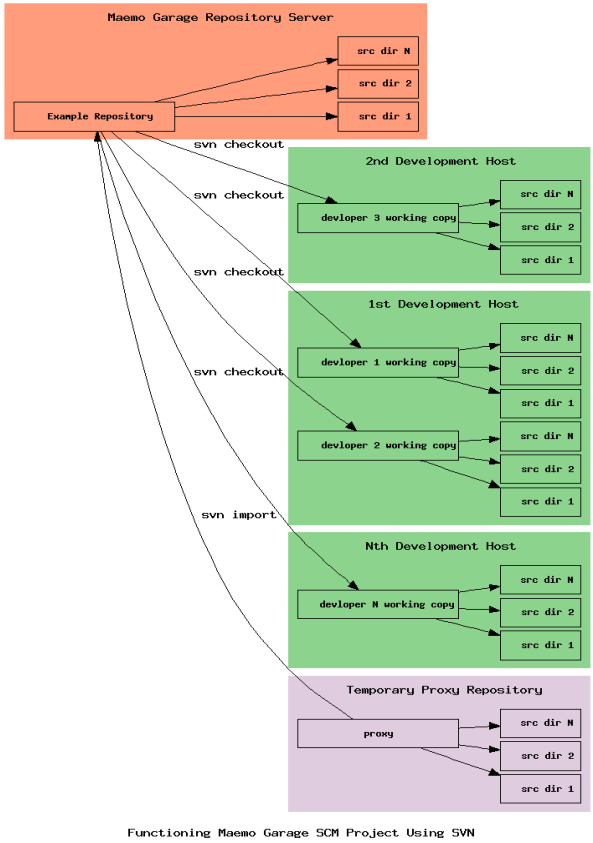

Figure 25. Maemo Garage Repository

The Maemo SDK will only install and run on a Linux operating system like Debian or Ubuntu. By means of a VMWare image, it's possible to provide a working Linux environment where install and run this sandboxed [52] . This means that the development process is very similar to a normal desktop GNU/Linux, so the developer is isolated for the kink of embedded development, such as cross-compiling, that are handled transparently by the packet Scratchbox [54] .

The user interface architecture of Maemo is based on GNOME framework, especially the GTK+ widget set [53] . The Maemo platform provide to end user many components such as the GStreamer multimedia framework, the GConf configuration management, and the XML library, that allow to extend GTK+/GNOME technologies by providing extensions for a mobile desktop. Hildon [55] is the framework that provides several libraries to interface with the desktop environment UI elements and services.

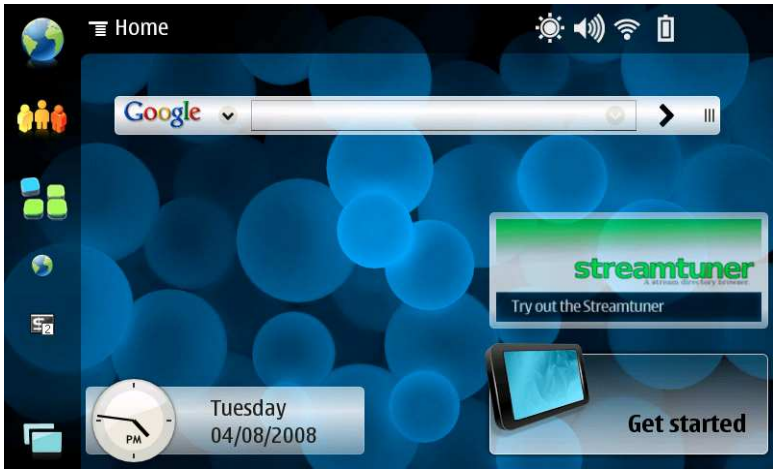

Figure 26. Maemo screenshot

Currently C is the only official programming language for maemo, but through the maemo community, unofficial support exist for several other languages, such as Ruby on Rails, Java, etc. Besides, Maemo provides support for the Eclipse development environment and others integrated development environments.

3.2.2.3 Google Android

Android is a software stack for mobile devices that includes an operating system based on the Linux kernel, middleware and key applications that is developed by Open Handset Alliance, integrated by Google, HTC, Motorola, Intel, Nvidia and other companies. The Android SDK [56] provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language.

At the moment, there are a lot of devices that run Google's Android, such as mobile device and Internet Tables device. This reference shows a list of hardware devices [58] .

Figure 27. Android desktop example

From the developer view, Google Android offers a complete software development kit that includes a variety of custom tools to develop mobile applications on this platform [59] . In particular, the Android Development Tools plug-in adds powerful extensions to the Eclipse IDE [60] , and the Android Emulator [61] , that provides a QEMU-based device-emulation tool for design, debug, and test the developed applications. Furthermore, Android SDK includes a variety of other tools for debugging, packaging, and installing your applications on the emulator and device. All application are developed on Java programming language, but it includes a set of C/C++ libraries used by various components of the Android system such as a System C library, FreeType for bitmap and vector font rendering, and some Media Libraries.

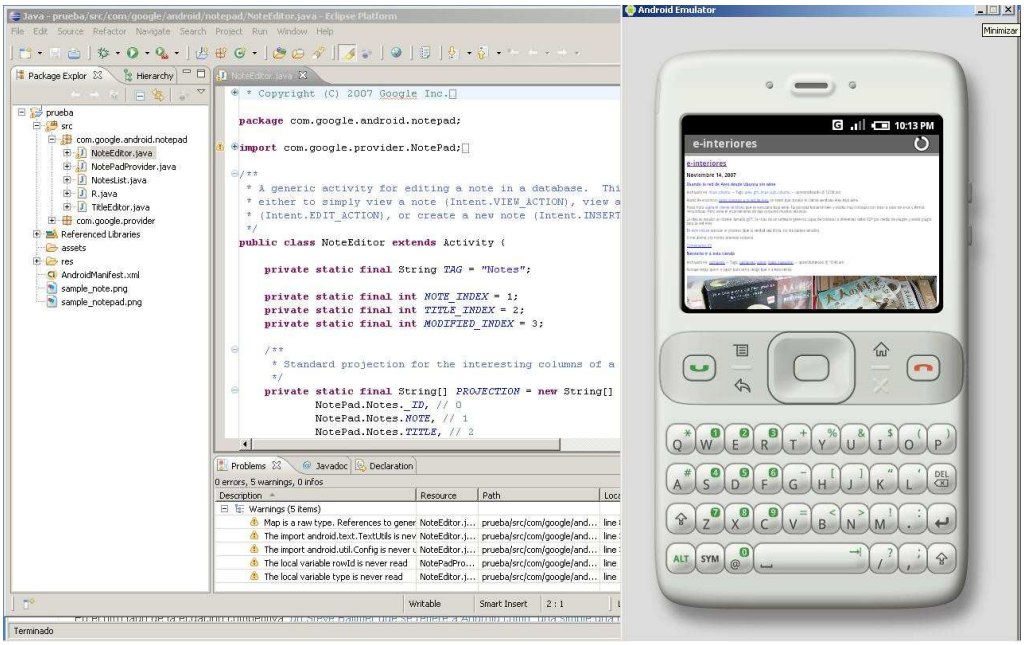

Figure 28. Android and Eclipse integration

The recommended way to develop an Android application is to use Eclipse with the ADT plug-in, but it is possible develop applications in another IDE, such as IntelliJ, or in a basic editor, such as Emacs [62] .

Finally, Android platform has a developers community with forums, mailing list and IRC channels [63] . Furthermore, developers could access to documents that explain the signing, versioning and publish application process [64] .

3.3 Hardware platform

3.3.1 Arduino

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software [98] .

The Arduino software consists of a development environment (IDE) and core libraries distributed under GNU General Public License and Creative Commons License. The IDE is written in Java and core libraries are written in C and C++. Arduino boards can be programmed using Arduino language or straight C/C+. Arduino language is a set of C and C++ functions that can be called from code.

Arduino community provides simple programas about basic Arduino behaviour. http://arduino.cc/en/Tutorial/HomePage. These basic examples controls:

- Digital and Analog I/O: turn on and off LEDs, read pushbuttons, play speakers, MIDI interfaces, analog sensors...
- Sensors: accelerometers, knock detection devices, ultrasonic object detection.
- Display control: matrix of LEDs and Liquid Cristal Displays.
- EEPROM control: clear, read and stores values in an EEPROM.

- Motors: Stepper Motor control and servos.
- Communications between Arduino board and PC: send data to the computer in order to process them and receive orders through keyboard, mouse or process running on computer.

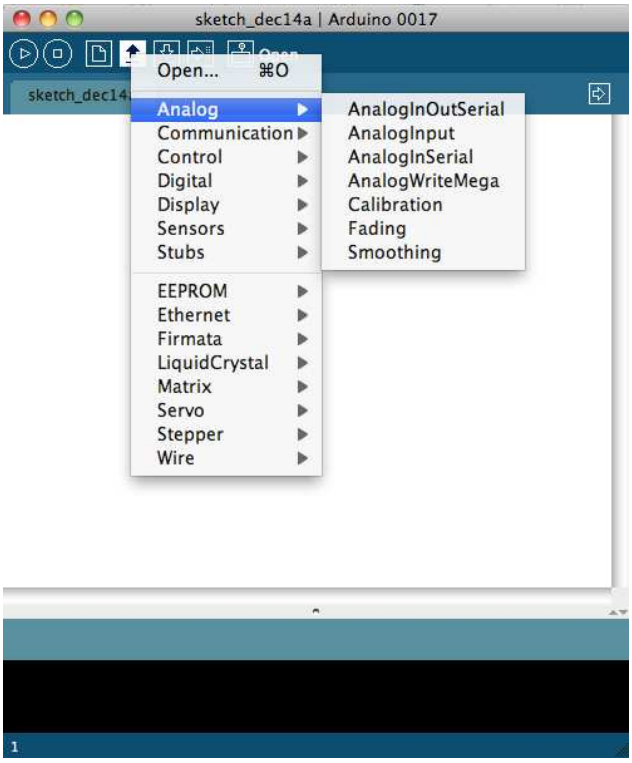

Figure 29. Arduino development enviroment.

3.3.2 Beagle board

This toolset has been covered in D1.1, including user aspects.

3.3.3 DIY Robots: Tux Droid, Lego Mindstroms

DIY robots are new concepts of toys that are construct by user self. This toolset includes a lot of sensors that permit to detect events and interact with environment. Furthermore, these toolset offer an easy programmable interface.

3.3.3.1 TUX Droid

In case of Tux Droid, this SDK allows it to announce events by gestures and by ALSA driven sound [102] [103] . The events are detected by specific gadgets, which are handled by the Tux Gadget Manager that is included in the tuxsetup package and is an easy to use graphical interface to access Tux functions [104] . At this reference [105] , it possible to find an OpenSource developers community dedicated to create smart-companion features for PC-controlled robots.

3.3.3.2 LEGO-MINDSTROM NXT [106]

Since the NXT version, Lego releases the device firmware as OpenSource [107] and provided several developer kits oriented to information on host USB drivers,

executable file format and bytecode reference (SDK); documentation and schematics for the NXT brick and sensors (HDK); and documents about the protocols used for Bluetooth communications (BDK).

The programming software that is provided by NXT version is NTX-G [108] . In older version like RCX, this SDK was based on the GUI Robolab. In both case, LabView develops them and there are two different programming interfaces, depends the kind of kit (educational or other) used. Also, this toolset allows use third-party languages such as Java, Ruby on Rails, C/C++, etc [109] .

Exists a strong community of developers and final users that offer designs, programming techniques and other ideas associated with Lego Mindstorms [110] . Lego also encourages sharing and peering by making software code available for downloading and by holding various contests and events.

3.4 Software platform

3.4.1 Mashups

3.4.1.1 WSMX Engine

From a user/developer point of view there are several tools to make easier work with WSMX[132] . Nowadays is as important make a potential technology as supply tools to work with them, like:

- **Task management**
    - GUI (HTTP): The WSMX Management Console is available at http://localhost:8080/ *(port as specified in the configuration).* It contains general information about the running WSMX instance and facilitates basic administration tasks.

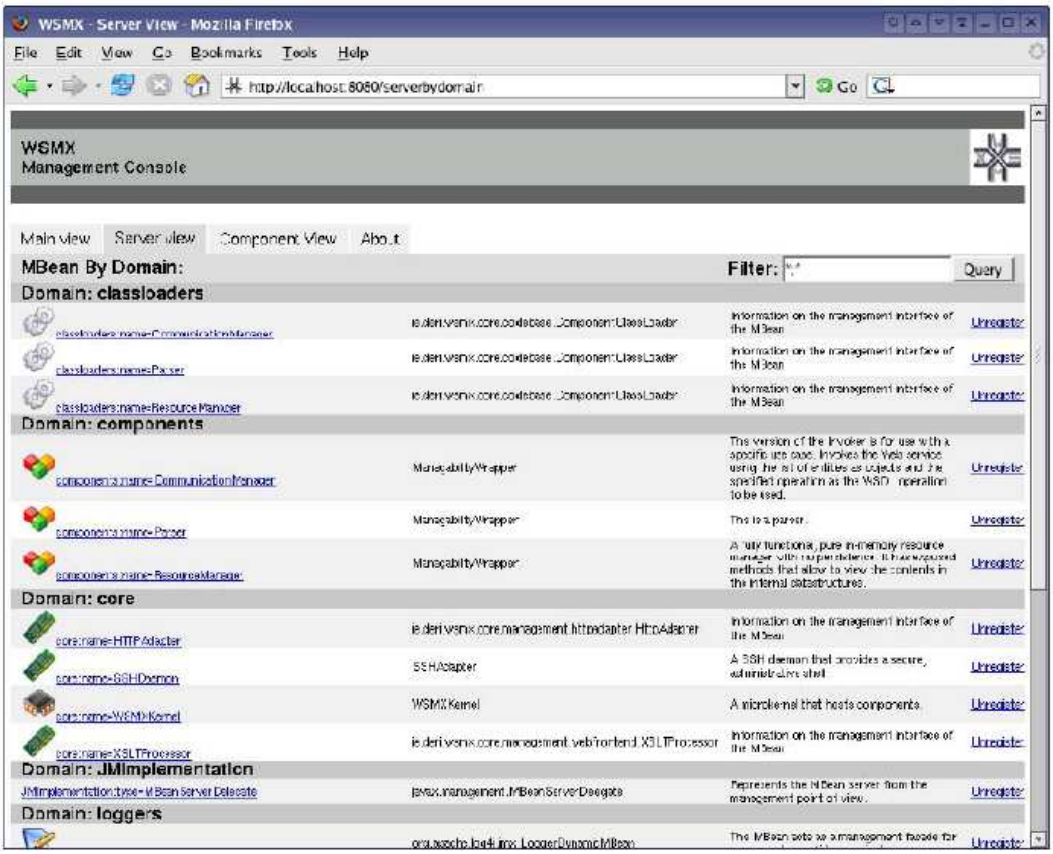

Figure 30. WSMX Management Console (GUI/HTTP)

- TUI (SSH): A Terminal User Interface (TUI) for the basic administration tasks as depicted in above figure is available via SSH using ssh root@localhost –p 8090

Figure 31. WSMX Management Console (TUI/SSH)

- **Monitoring tools**
  - WSMX Monitor: WSMX Monitor is a graphical tool (plugins) for monitoring the state of each of the WSMX components within the WSMX system, along with the system itself, for the Web Service Modeling Toolkit (WSMT).

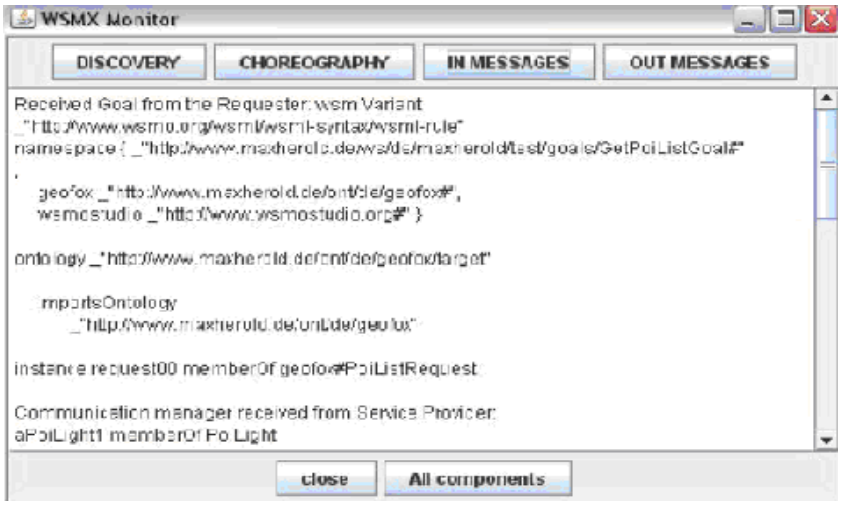

Figure 32. WSMX Monitor

- **Tools related to WSMX**
  - WSMT Web Services Modeling Toolkit [133]

WSMT is an Integrated Development Environment (IDE) and has been created for the rapid creation and deployment of the tools as plugins for Semantic Web Service.

WSMT has three main areas in design-time: "*WSML editors"*, "*WSML Discovery view, Cache view, and WSML-Reasoner"*, and "*Mapping editor and Munit"*; and has one area in run-time: "SEE perspective: Interfacing with semantic Execution Environment external systems". This perspective include features a servers view to connect to a WSMX instance for viewing and modifying stored WSML artifacts; and a SOAP Message view to send messages to the WSMX EntryPoints web service and view the results.

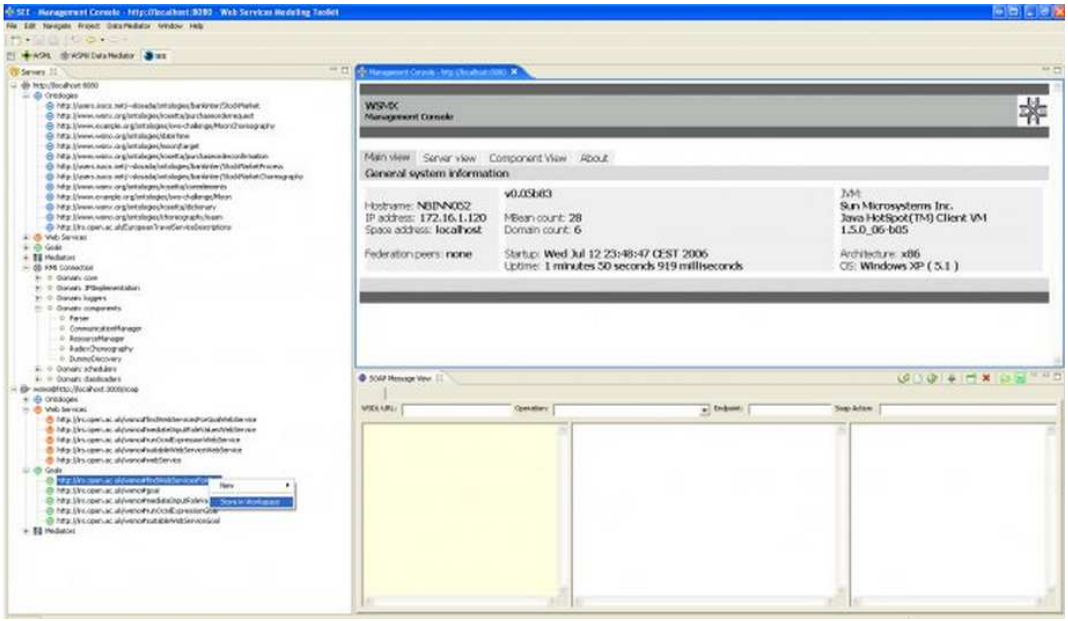

Figure 33. The SEE Perspective

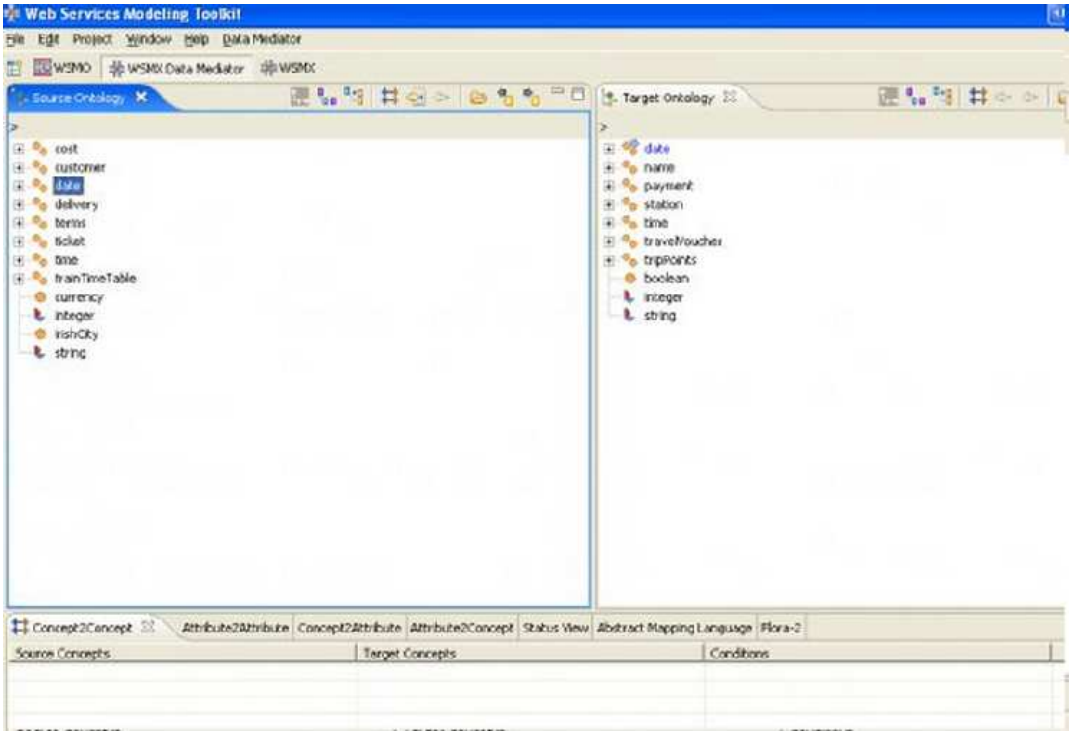

Figure 34. WSMX Data Mediation Mapping Tool - Suggestion

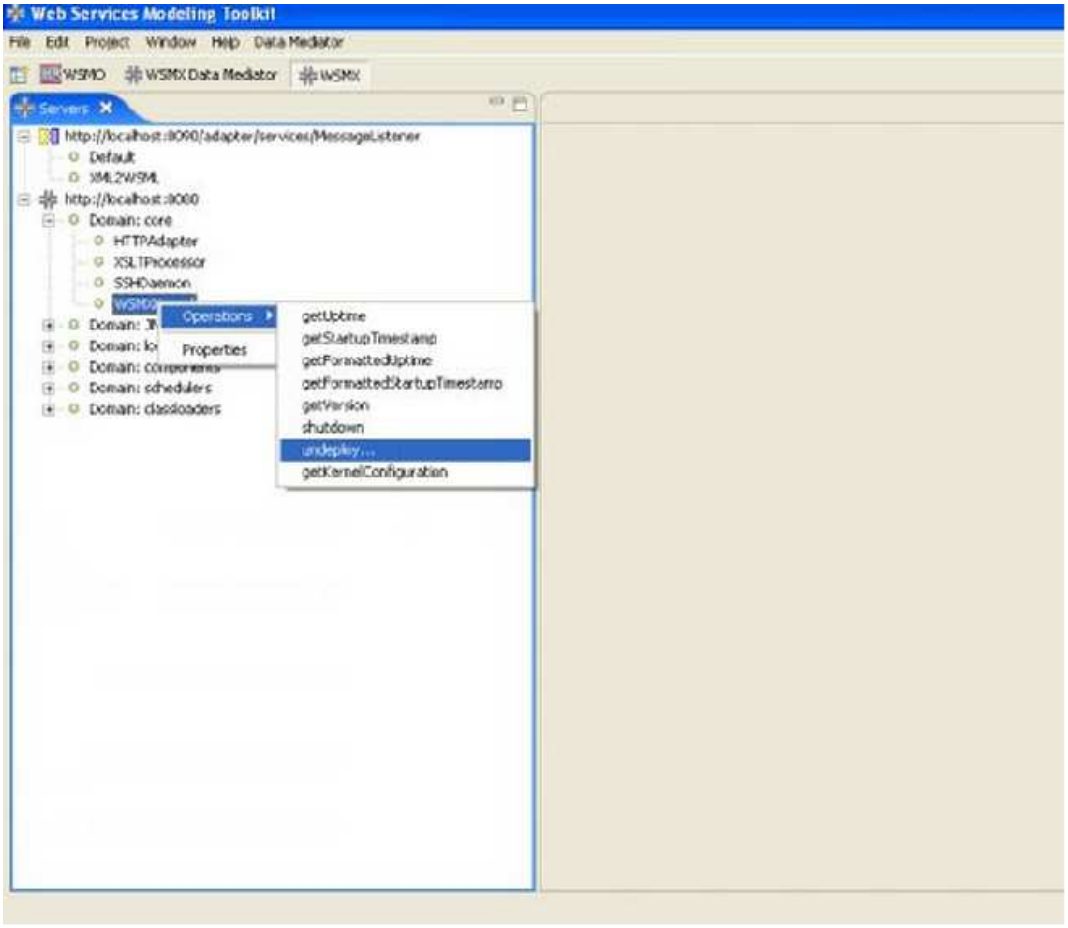

Figure 35. WSMX Management - Invoke MBean Operation

- *WSMO Studio* [134] .

WSMO Studio is an Eclipse-based tool for Semantic Web Services modelling based on WSMO. On of their functionality is providing front-end for ontology / service / goal repositories by a WSMX Adapter (is an Eclipse plug-in integrated with WSMO Studio, which is accessible through the *WSMX Management* perspective).

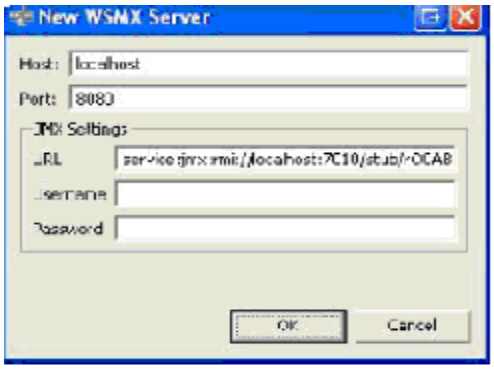

Figure 36. Adding a new WSMX Server

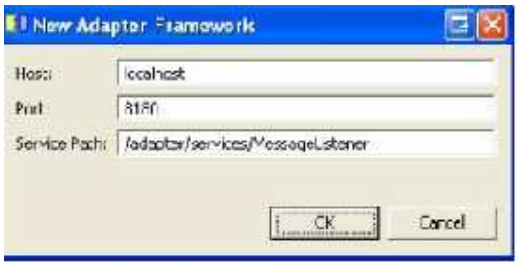

Figure 37. Adding an Adapter Framework

Adapter Framework View**:** This UI component provides functionality for: *Viewing deployed adapters, Deploying a new adapter, Un-deploying an adapte, Testing an adapter.*

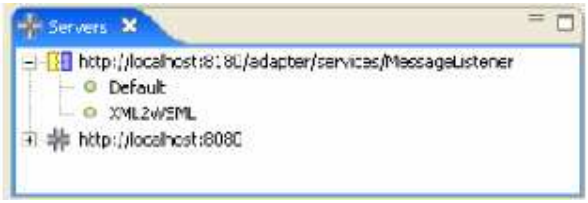

Figure 38. Deployed adapters

WSMX Server View**:** This component provides functionality for: *Viewing deployed components, Viewing the properties of a component, Invoking operations on deployed components, Accessing the WSMX Management Console.*

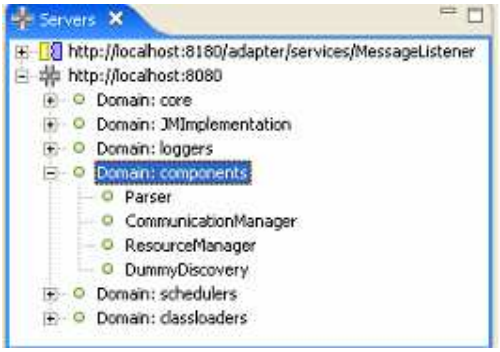

Figure 39. Viewing deployed components

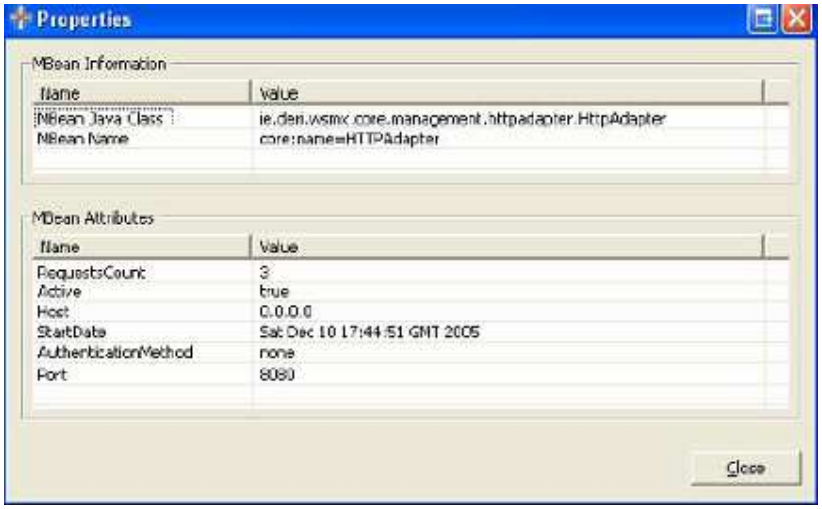

Figure 40. Component properties

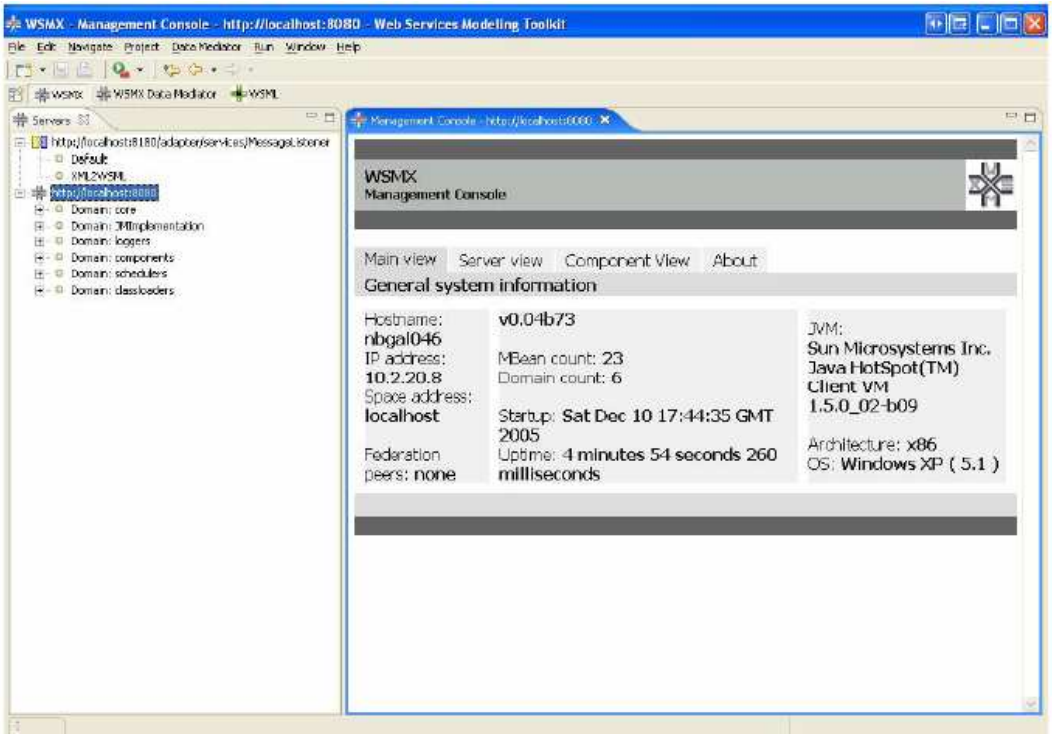

Figure 41. WSMX Management Console

- **Development customizable components**
    - The WSMX Integration API is a collection of libraries required for the integration of loosely coupled components with the main WSMX system. Components must implement interfaces from the provided infomodel to make this integration possible.

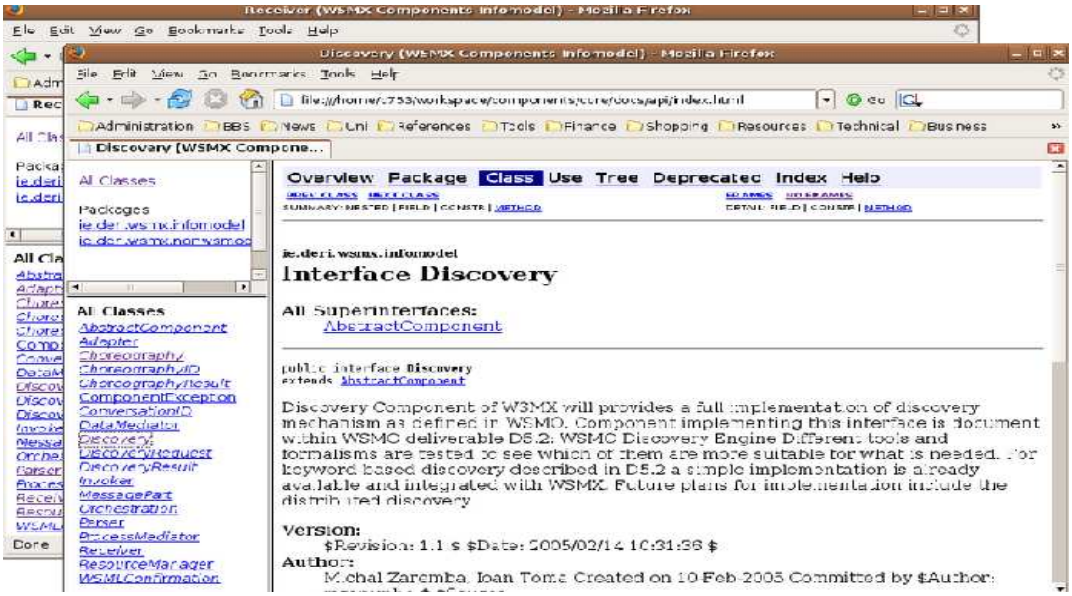

Figure 42. WSMX Integration API

3.4.1.2 Lotus Mashups

This toolset has been covered in D1.1, including user aspects.

3.4.1.3 Yahoo! Pipes

Yahoo! Pipes is a free online service from Yahoo! that provides a graphical user interface for building data mashups that aggregate web feeds, web pages, and other services, creating Web-based apps from various sources, and publishing those apps [97] .

From user point of view, Yahoo! Pipes provide a visual editor that allows, by dragging pre-configured modules onto a canvas and wiring them together, run your own web projects, or publish and share your own web services without ever having to write a line of code [96] . The Pipes editor is a JavaScript authoring tool that lets you create and edit Pipes in an intuitive visual interface. The editor consists of the following three panes:

- The Library pane on the left hand side lists available modules and saved Pipes.
- The Canvas pane in the centre is the main work area for assembling Pipes.
- The Debugger is a resizable pane at the bottom, which lets you inspect Pipe output at various stages in your Pipe.

It possible to create and edit Pipes by moving modules onto the Canvas from the Library pane and wiring them together with your mouse.

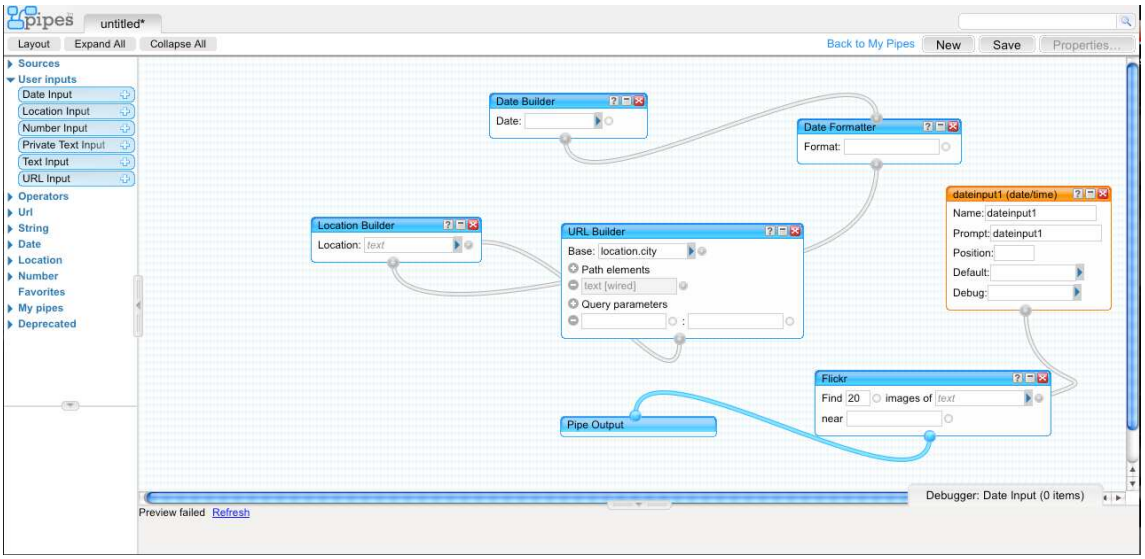

Figure 43. Yahoo! Pipes Visual Editor.

3.4.1.4 SOA DIY cartography Tools

This toolset has been covered in D1.1, including user aspects.

3.4.1.5 Microsoft Popfly

As it happened with the Google Mashup editor, Popfly was a webapp that enabled users to create mashups. A user could join data coming from different sources and together with a visualization tool create their own web service. In August 2009, the service was taken down.

3.4.1.6 Intel Mashmakes and MARGMASH

This toolset has been covered in D1.1, including user aspects.

3.4.1.7 MARMITE

This toolset has been covered in D1.1, including user aspects.

3.4.1.8 EZWEB

This toolset has been covered in D1.1, including user aspects.

3.4.1.9 OPUSE SCE

This toolset has been covered in D1.1, including user aspects.

3.4.1.10 Apple Automator

This toolset has been covered in D1.1, including user aspects.

3.4.1.11 Facebook

Facebook is the most widely used social network worldwide at present. It is a website where users build their own profile, keep in contact and get real-time information about their friends, networks or events. What distinguishes Facebook from other

competitors is the concept of platform. Facebook exposes an API[1] that allows the development of applications to be used on the site. This API provides explicit access to core features, such as profiles, photos, events or friends of the users. Likewise by the usage of the Facebook Markup Language (FBML), developers can customize the looks of their app. These applications put together by a user become a true mashup. In fact the user interface (mashup) is refined by the user as they see fit.

Furthermore, Facebook also provides another set of APIs, known as Facebook Connect [2], that allow 3rd party applications outside Facebook itself to access the user data, friends, photos, and more. As a result it brings the Mashup outside Facebook website and into the 3rd party website.

3.4.1.12 HyperCard

This toolset has been covered in D1.1, including user aspects.

3.4.1.13 OpenKapow

Openkapow [15] is an open service platform, where both experimented developers and non skilled users can build their own services (called robots) and run them from openkapow.com. These robots accesses web sites and allows to use data, functionality and even the user interface of other web sites in an graphical way [16] .

From the user perspective, OpenKapow is a visual development environment which offers a friendly user-interface for non-technical users to create applications called robots. A robot in openkapow is a small program that automates what a person can do in a browser. This includes navigating web sites by clicking on links and submitting forms, extracting data from a site, etc. Robots are created in the development environment RoboMaker without any programming and robots are then hosted and run on openkapow's servers. The behavior of a robot can be affected by input values (for example the username and password to use to log in to a password protected site) and the robot produces an output (for example the current rate of a specific stock).

RoboMaker is the visual development environment used to create robots. In RoboMaker a robot is created by putting together steps and configuring those steps. This is done using a point-and-click interface that includes a browser view that allows the user to see the page the robot interacts within the same way as if it was in a normal browser such as Internet Explorer or Firefox. As shown in next picture:

[1] http://wiki.developers.facebook.com/index.php/API

[2] http://developers.facebook.com/connect.php

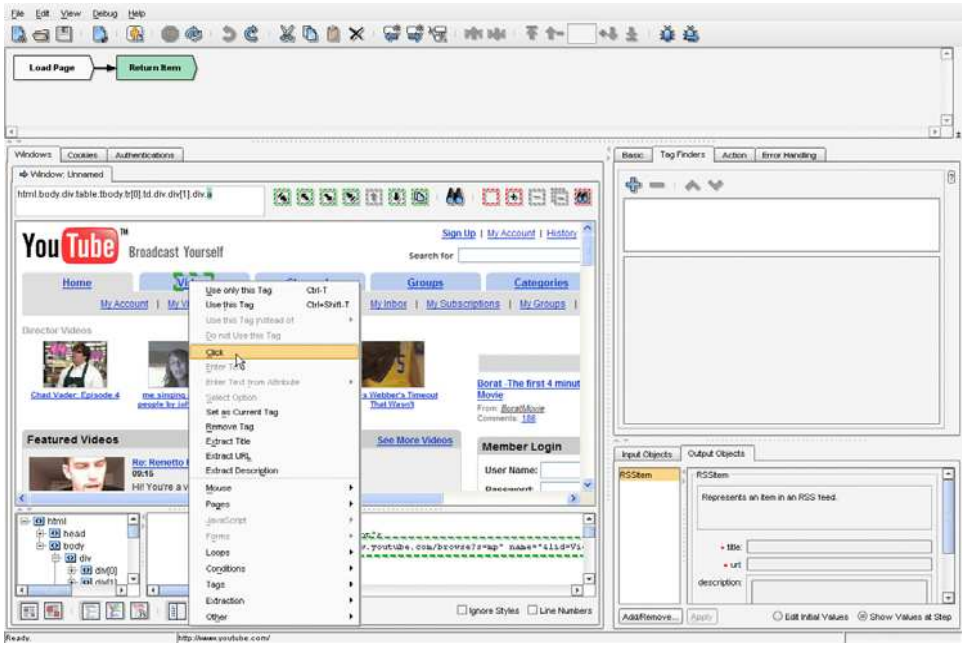

Figure 44. Visual development interface

There are 3 different types of openkapow robots that you can create, RSS/Atom feed robots, and REST robots which are used as base for mashups development. A REST robot is a robot that is run as a REST [17] service, this is a web service that is available from a normal URL. The robot then outputs the result of its work in for example XML, HTML or JSON. REST robots are normally used to create an API-like interaction with a web site and the REST robots are usually called from within a program (written in for example PHP, Ruby on Rails, C# or Java).

The creation of these REST robots is completely guided by means of a wizard, so non-professional users can use the development environment with-out having technical skills.

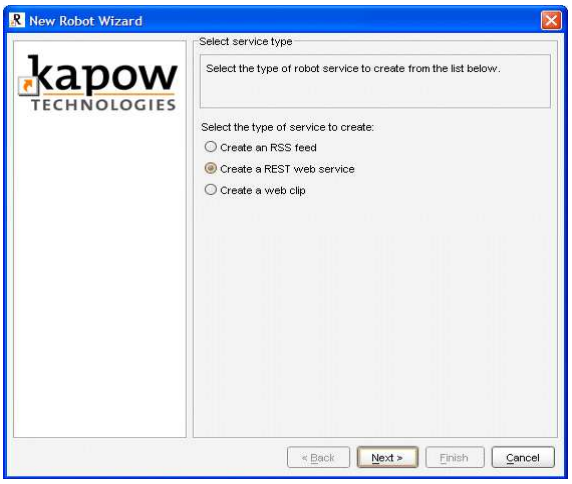

Figure 45. RoboMaker Wizard

All the actions to be performed by the robot are showed in a list when "right-clicking" in any of the source content page elements (see Figure 46).

Figure 46. Actions to be performed by the robot

Once the robot is complete, it can be published in the Web and hosted in the OpenKapow Servers.

3.4.2 Video editing tools

In this section, we'll describe different tools for video editing. Two types of tools will be presented: PC-based Video editing tools and Web 2.0 video editing tools.

3.4.2.1 PC-based Video editing tools

Cinelerra [39] is addressed to two types of moviegoers: producers who create new content and revisit it for further refinement, and consumers who want to acquire the content and watch it. Cinelerra has many features such as composition, resolution processing, (Figure 47). Cinelerra is not intended for consumers.

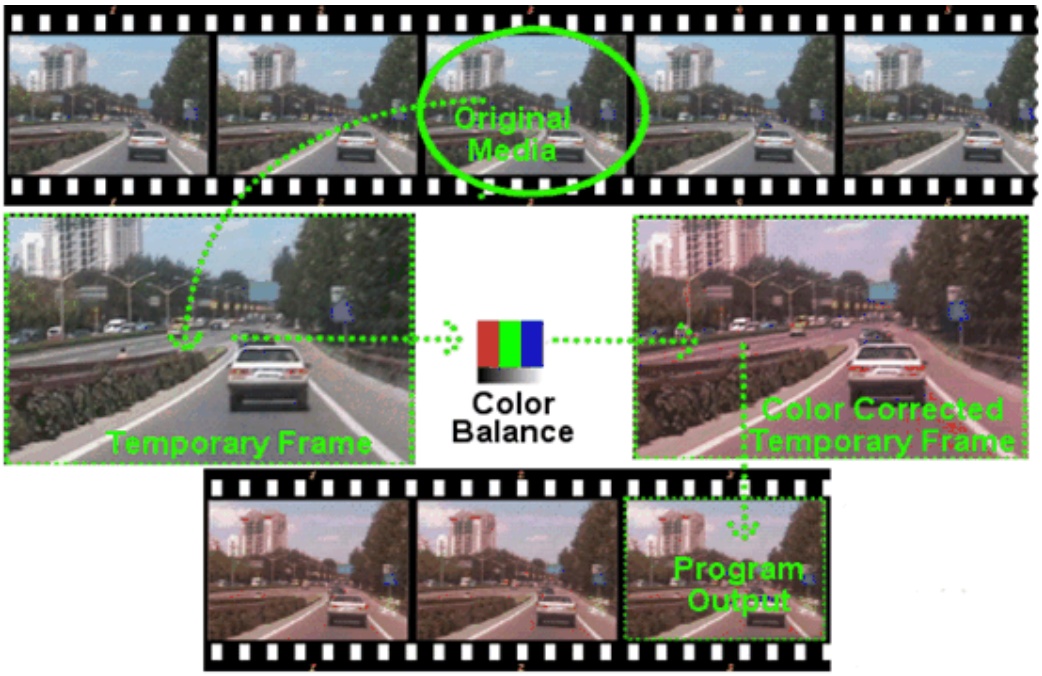

Figure 47. Cinelerra screenshot [39]

Kdenlive [40] is a modern filmmakers which allow to mix different kinds of media, including video, audio and images. Kdenlive is built upon Kdenlive video engine and multimedia framework and ffmpeg frameworks, which provide features to mix virtually any kind of media. The main features of Kdenlive are:

- Mix different media without prior import:
  - Any Video, audio or image files supported by Kdenlive
  - Custom profiles including resolutions, frame rates, PAR and DAR
- Support for a wide range of codecs and formats:
  - Mpeg2, mp4 and h264 video.
  - Mp2, mp3 and ac3 audio.
  - Lossless video (SNOW lossless codec, etc ...).
  - Free video (Ogg vorbis, etc ...).

The Figure 48 shows the main window of HCI of Kdenlive. Kdenlive should be used by video amateurs as well as advanced users.

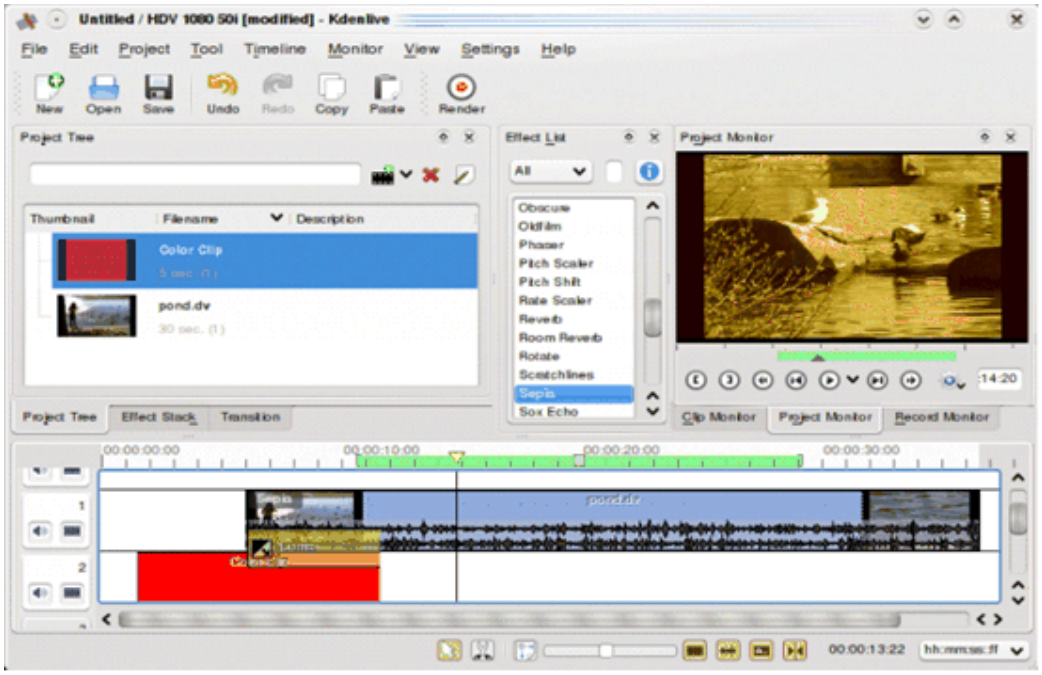

Figure 48. Kdenlive main window [40]

Blender [41] is the free open source 3D authoring tool, available for all major operating systems under the GNU license. Blender provides large component of modelling, texturing, lighting, animation and video post-processing functionality in one package (Figure 49). The Key Features of Blender are:

- Fully integrated creation suite. Blender offers tools for the creation of 3D content, including modeling, uv-mapping, texturing, rigging, skinning, animation, particle and other simulation, scripting, rendering, compositing, post-production, and game creation;
- Cross platform, with OpenGL uniform GUI on all platforms, ready to use for all versions of Windows (98, NT, 2000, XP), Linux, OS X, FreeBSD, Irix, Sun and numerous other operating systems;
- High quality 3D architecture enabling fast and efficient creation work-flow;
- User community support by forums for questions, answers, and critique at http://BlenderArtists.org and news services at http://BlenderNation.com;
- Small executable size, easy distribution.

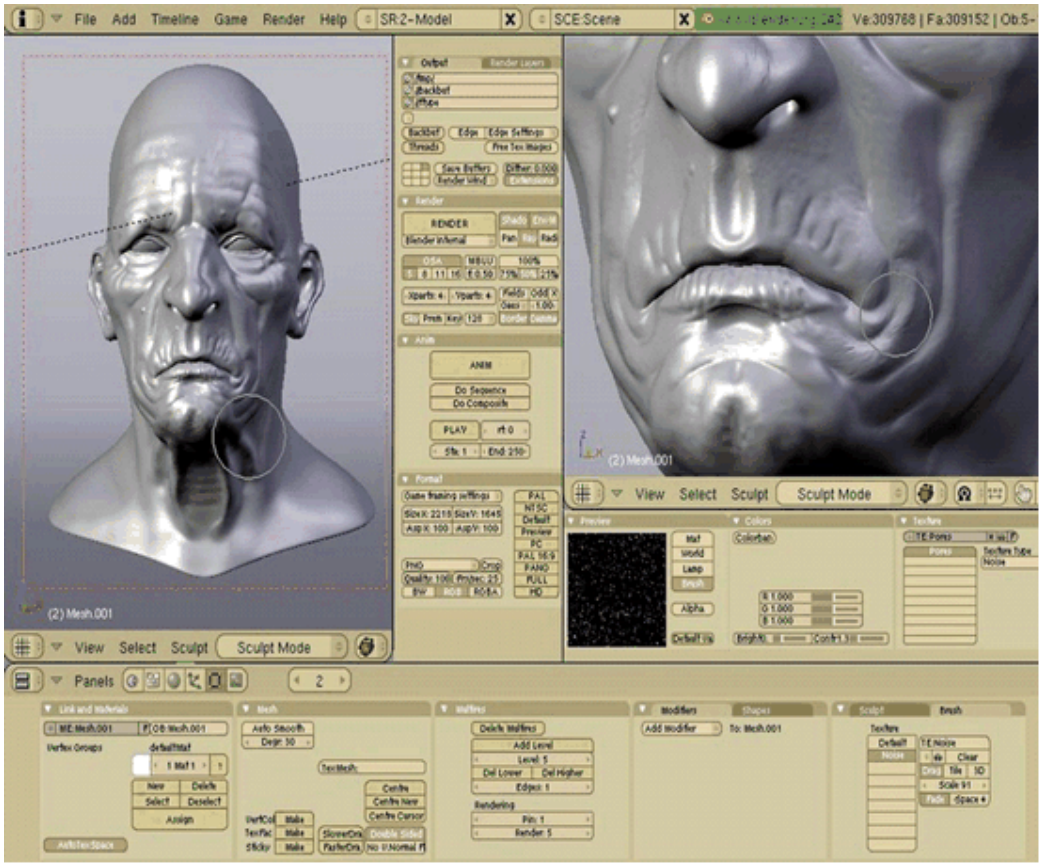

Figure 49. Blender HCI [41]

PiTiVi [42] is an opensource video editor written in Python for the high-level logic and user interfaces (Figure 50). PiTiVi allows fast development time and an easy extension of component through native python plug-ins.

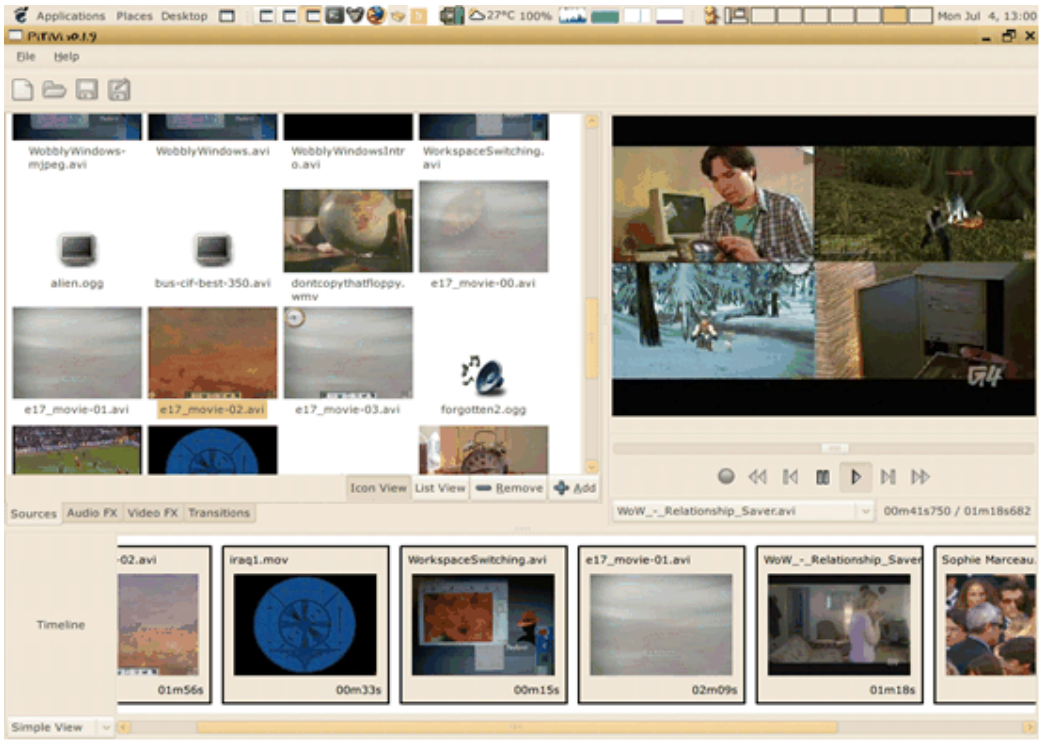

Figure 50. Main window of PiTiVi [42]

Kino [45] is a non-linear DV editor for GNU/Linux (Figure 51). With Kino, we can load multiple video clips, cut and paste portions of video/audio, and save it to an edit decision list (SMIL XML format). Kino can export the composite movie in a number of formats: DV over IEEE 1394, Raw DV, DV AVI, still frames, WAV, MP3, Ogg Vorbis, MPEG-1, MPEG-2, and MPEG-4.

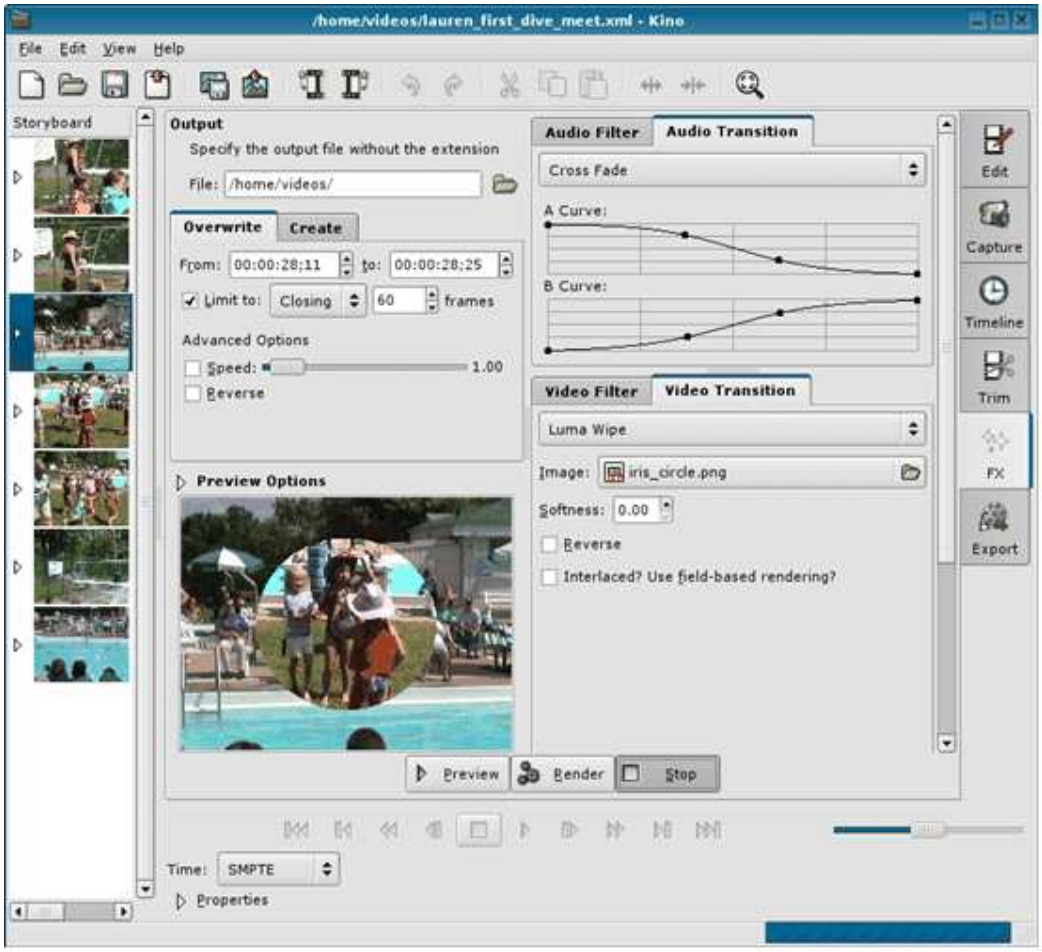

Figure 51. Kino video editing user interface [45]

Avidemux [43] is a free open-source program designed for multi-purpose video editing and processing. Avidemux provide a user interface (Figure 52) and script to edit videos. The scripting engine used by Avidemux is SpiderMonkey, and it is an ECMAScript/JavaScript engine.

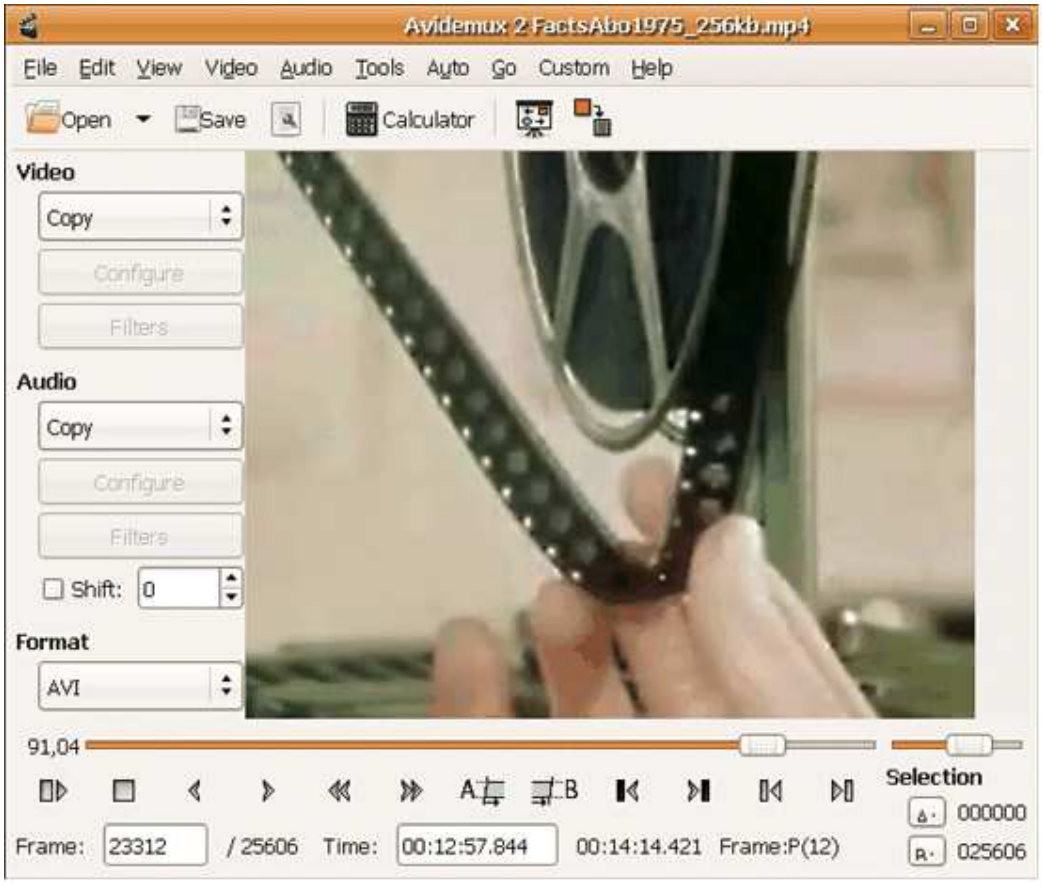

Figure 52. Avidemux HCI video editing [43]

Jahshaka [44] is an open source, free, video editing software and special effects, post-production in real time (Figure 53). It works by modules (animation, titling, keying, tracking, paint, calibration, etc..) And is open source and free under GNU GPL. Jahshaka makes possible to import image files (JPG, PNG, GIF, TIFF) and 3D (3ds, Obj) and animate the whole (movement in 3D space, changes shape, size, opacity).

Figure 53. Jahshaka video editor user interface [44]

3.4.2.2 Web 2.0 video editing tools

Cuts [46] provide online interface to upload your own video, or grab one from YouTube for example, add some pre-recorded sound clips. The creations can be shared with friends (Figure 54).

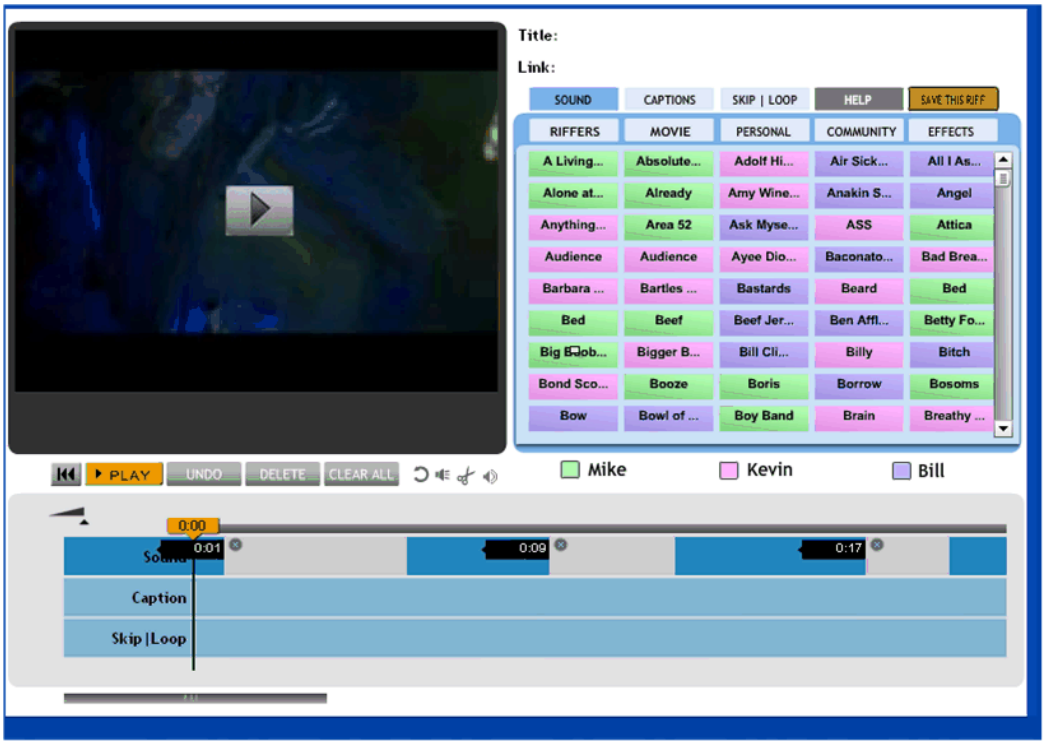

Figure 54. Cuts online video editor main window [46]

JayCut [47] mixer allows you to make creative masterpieces, or mixes for short, out of your video clips and photos. The created video can be exported in facebook or myspace. JayCut supports more to over a hundred input formats and allow exporting to a cell phone just as easily as to video sites, blogs and Social Networks (Figure 55). The key features of JayCut are:

- Multiple Upload: Allows users to upload multiple files with one click
- Easy editing: Trim, remix and combine videos and photos within minutes
- Collaborative Editing: Let other user remix and continue your work
- Export to any format: Export mixes to over a hundred input formats incl. cell phones

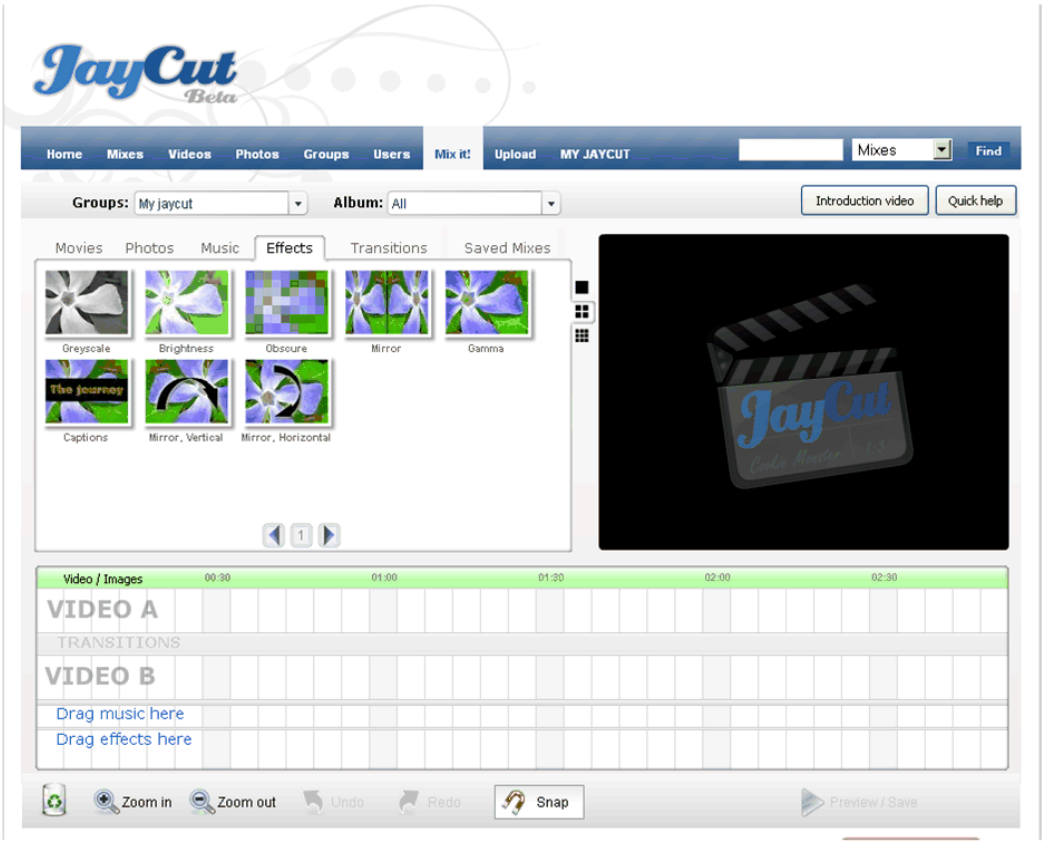

Figure 55. JayCut online HCI video editing [47]

MovieMasher [48] is a set of Adobe Flash™ applets that provide front-end tools for common video editing tasks. The major functions found in movie masher are: trim, composite and time shift video, mix and fade multiple audio tracks, add effects, transitions and titling (Figure 56**Error! Reference source not found.**). The **MovieMasher** Applet can integrated in PHP server and source code is provided under Mozilla Public License 1.1

Figure 56 - MovieMasher Applet video editing online HCI [48]

3.4.3 Other

3.4.3.1 ITEA-Amigo

This toolset has been covered in D1.1, including user aspects.

3.4.3.2 AmbientTalk

AmbientTalk [191] is a scripting language which embodies the ambient-oriented programming paradigm [192] which is described in D1.1. While it is targeted at programmers, rather than end-users and professional amateurs, the language is designed to provide straightforward abstractions that allow programmers to intuitively deal with the volatile connections and lack of infrastructure that characterise mobile ad hoc networks.

AmbientTalk is a dynamically typed, object-oriented scripting language to develop applications for mobile ad hoc networks. Due to the particular requirements of this setting, AmbientTalk provides dedicated abstractions, which are not readily available in other contemporary programming languages.

- Event-driven: AmbientTalk has built-in support for actor-based concurrency: AmbientTalk applications consist of one or more actors (each encapsulating a suite of objects), which communicate with one another in an asynchronous and event-driven way. There are no threads, no locks, no deadlocks and no data-level race conditions.

- Distributed: AmbientTalk provides built-in language constructs to enable objects to dynamically discover one another in a mobile ad hoc network. Communication between objects hosted on different devices is necessarily asynchronous and is implicitly buffered to guard against transient disconnections,

- Embedded: AmbientTalk applications run on top of a Java Virtual Machine, and can reuse class libraries and applications developed in Java. Hence, AmbientTalk can be used as a scripting language to describe the distributed behavior of Java applications.

The AmbientTalk interpreter is freely available as an open-source project [193] At present, tool support for the development of AmbientTalk applications is provided in the form of Eclipse and TextMate plugins (to develop and run AmbientTalk applications) and integration with the Causeway message-oriented distributed debugger [194]

Due to the fact that AmbientTalk is embedded in Java, AmbientTalk programs can only be run on devices which provide a suitable Java Virtual Machine (JVM). Three JVM profiles are currently supported:

- J2SE (1.4 and higher): Java 2 Standard Edition: this profile is designed to provide a JVM for desktop and laptop computers.

- J2ME CDC (1.1 and higher): Java 2 Micro Edition Connected Device Configuration: this profile is designed to provide a JVM designed and optimized for limited devices. The CDC profile targets high-end smart-phones and supports reflection and tcp/ip sockets.

- Android (version 1.5 and higher): The Android platform is described in section 3.2.2.3 of this document. Its software stack provides Java on mobile phones and Internet Tablets.

3.4.3.3 SOA4ALL Platform

SOA4All Studio is the web-based user front-end. It allows the creation, provisioning, consumption and analysis of services published in the SOA4All delivery platform. It consists of three different components:

- Provisioning Platform, it enables the annotation of both REST and WSDL services, through MicroWSMO and WSMO-Lite annotations respectively.

    - Process Editor, is a component of the Provisioning platform, that allow users to create, modify, share and annotate executable process models, based on a light-weight process modeling language (LPML). This language is a subset of both BPEL and BPM, and aims at hiding away most of the service composition complexity, still providing enough expressive power to create useful compositions.

- Consumption Platform, serve as an interface for users to discover and invoke services that fulfill their will. It takes into account the user profile and context to offer a better composition.

- Analysis Platform makes use of the data gathered by the monitoring system to extract meaningful information.

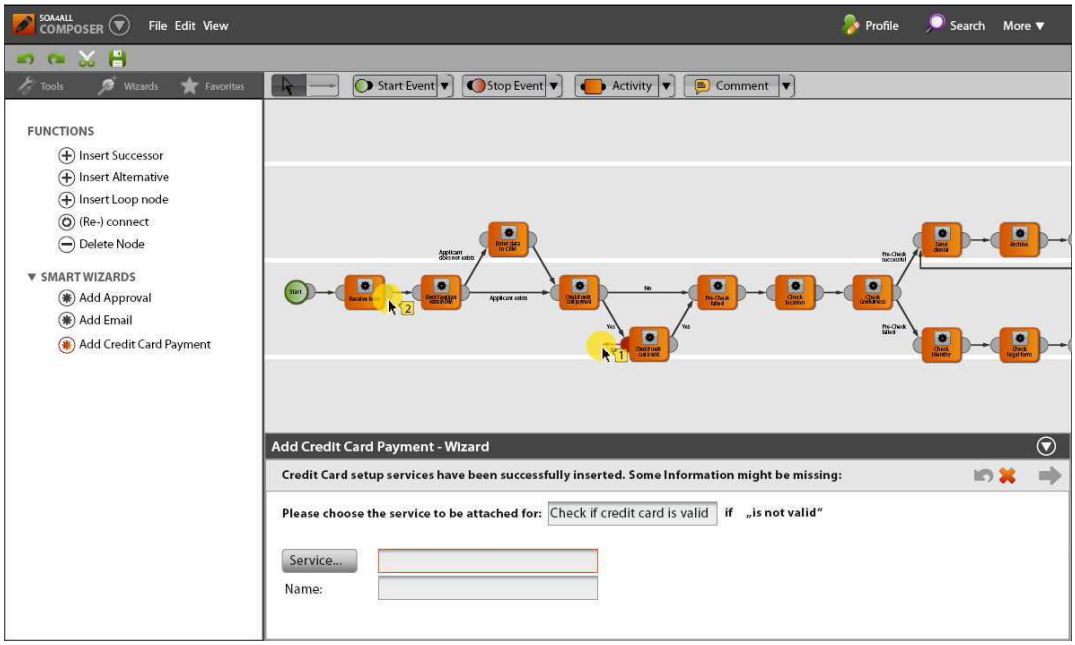

Figure 57. SOA4All Composer

3.4.3.4 Ontology Engineering Tools

In DIY-SE, there are three main Ontology Engineering (OE) tools – the Developing Ontology Grounded Methodologies and Applications (DOGMA) Studio Workbench /Collibra Studio Workbench, the Ontology-based Data Matching Tool Set, and the Semantic Decision Table Tool Set.

3.4.3.4.1 DOGMA Workbench/Collibra Studio

Collibra is a spin-off company of VUB STARLab. Collibra Studio[3] is a tool suite that enables IT professionals and business analysts to work together to re-conciliate and apply the semantics of existing information sources. Collibra Studio allows the creation of business semantics models that provide unambiguous definitions, identifications and mappings towards existing data sources in a way that is both simple and powerful.

Collibra Studio offers a user-friendly environment in which end users can provide facts and fact types in natural language. The software then creates rich semantic models automatically. This approach enables any stakeholder to easily capture the meaning and relationships of the relevant concepts. These Business Semantics can be easily exported to various formats such as CSV[4], XML[5], UML[6], RDF[7] and OWL[8].

[3] http://www.collibra.com/trial

[4] comma-separated values

The Collibra Studio is designed and built based on the DOGMA Workbench, which is a collection of OE tools that supports DOGMA-based technologies, methods and tools.

An important aspect in the Meaning Evolution Support Systems (DOGMA-MESS) methodology is the community dynamics (De Leenheer and Debruyne, 2008), as illustrated in Figure 59, is characterized by Nonaka's four modes of *knowledge conversion*: *socialization, externalization, combination*, and *internalization* [136] .

At the heart of the community dynamics is the Ontology Server, that bridges the semiotic gap between the community system parts. There are three types of *knowledge workers*: the *knowledge engineer*, the *core domain expert* (CDE), and the *domain expert* (DE). As we will show, in DOGMA-MESS, the involved ontology evolution processes (*community grounding, rendering, alignment, and commitment*) are inherently driven by the social knowledge conversion modes.

1. Community Grounding: In this phase, shared conceptions of the world under discussion that emerged from socialization are analyzed by the CDE. With the assistance of the KE, he identifies the key conceptual patterns that are relevant to be further externalized to the ontology. It results in a generalized upper common ontology (UCO) which represents the conceptualizations that are common to and accepted by the community.
2. Perspective Rendering: All participating stakeholders' DEs render their perspective on the UCO, by specializing the conceptual patterns, resulting in a set of diverging stakeholder perspectives (SPs). Doing so, ontology evolution is grounded (bottom-up) in the community, starting with the variety of terminologies found in the community itself. This allows DEs to syntactically and semantically nuance their intensions in a more natural manner using their own vocabulary. In order to impose UCO reuse, different types of perspective reuse policies can be formalized, including articulation, specialization, and application. A reuse policy is formalized by a set of applicable operations on a perspective [137] .
3. Perspective Unification: In the lower common ontology (LCO), a new proposal for the next version of the common ontology is produced, combining relevant material from the UCO and various stakeholder perspectives. Basically, there

[5] Extensible Markup Language (XML) is developed by W3C (http://www.w3.org/XML/)

[6] Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. The standard is managed, and was created by, the Object Management Group (OMG, http://www.omg.org/).

[7] The Resource Description Framework (RDF) is one of World Wide Web Consortium (W3C) specifications originally designed as a metadata data model. It provides interoperability between applications that exchange machine-understandable information on the Web. (http://www.w3.org/TR/PR-rdf-syntax/)

[8] The Web Ontology Language (OWL) is a family of knowledge representation languages for soring ontologies, and is defined by the World Wide Web Consortium. Its semantics is based on Description Logic. It is intended to provide a language that can be used to describe the classes and relations between them that are inherent in Web documents and applications. (http://www.w3.org/TR/owl-guide/)

is only a very simple rule: all (selected) definitions need to be full specializations of the conceptual patterns in the UCO. This, however, is overly simplified. In the ontology evolution process, despite reuse policies, the constructivist paradigm should allow to override the reuse policies, and hence new definitions to be created that are not (complete) specializations, but represent new insights for the CDE in preparing new evolution rounds, for example. This makes the alignment process far from trivial. This process is conducted collaboratively by all involved DEs, the CDE, and the KE.

4. Perspective Version Commitment: The part of the LCO that is aligned by the community forms the legitimate UCO for the next version of the common ontology. All participating organizations finally internalize and commit their instance bases to the new version.

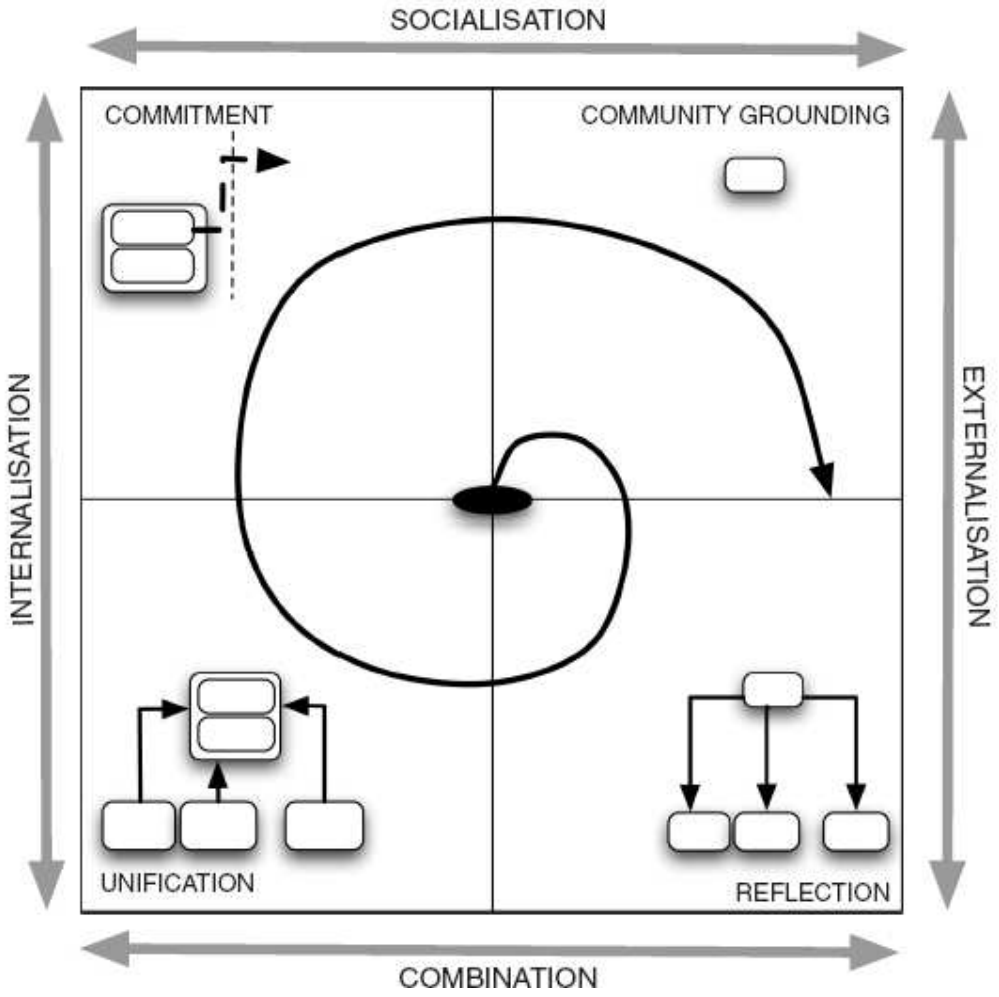

Figure 59. DOGMA-MESS ontology evolution spiral model [135]

In all phases, the views of all stakeholders are considered. This fourfold collaborative ontology evolution process is iteratively applied until an optimal balance of

differences and commonalities between organizational and common perspectives are reached that meets the communication goals.

A normal use scenario starts with semantic reconciliation, when the wizard starts, it will first ask for the context. Users can use standard browsing functionality to find the right context. In the example shown in Figure 60, the user has chosen to use a context with the name "Wine delivery", which is identified by its URI: "http://en.wikipedia.org/wiki/Wine". The context selected here will serve as a sort of reference for any fact types in the concept pattern.

Figure 60. Define your context with Collibra Studio

After the context is defined, the end user can introduce new concepts in the NormTree editor and browse the ontology with the outline wizard (Figure 61).

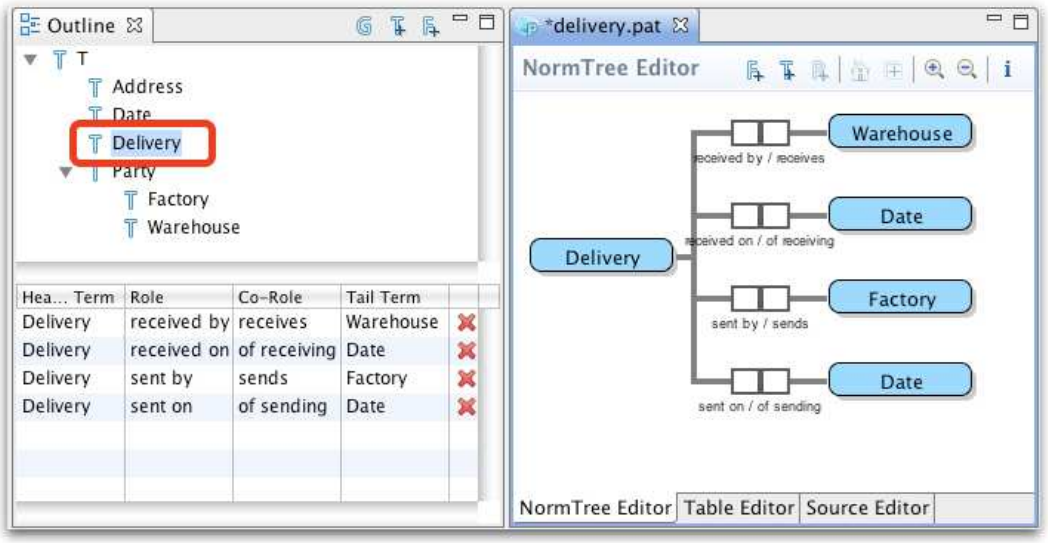

Figure 61. Introduce new concepts and browse ontology with Collibra Studio

The problem in OE is not on what ontologies are, but how they become operationally relevant and sustainable over longer periods of time, and how proper methodology and tool support can be provided. DOGMA-MESS, extending the fact oriented and layered ontology framework DOGMA, is a collaborative ontology evolution methodology that supports stakeholders in iteratively interpreting and modeling their common ontologies in their own terminology and context, and feeding back these results to the owning community. Comparing to other ontology editors, such as Protégé[9], the Collibra Studio is supported and can be easily extended with a set of DOGMA-MESS modules: Version Manager, Community Manager, and Perspective Manager.

3.4.3.4.2 ODMF Tool Set

In ontology engineering (OE), there exist many ontology matching approaches, e.g., algorithms and (open) tools. Some EU projects, such as Knowledge Web (KW[10]) and OpenKnowledge[11], have investigated a lot of efforts on ontology matching/integration algorithms and tools. For example, KW has produced 318 papers on ontology matching/integration algorithms and tools. An example of an ontology matching algorithm is Genetic Algorithm based Ontology Matching (GAON, [138]). Without doubt, ontology matching is an important hot topic in Ontology Engineering (OE).

However, the ontology-based data matching framework (ODMF) is not about ontology matching or ontology integration. Instead, it is to study ontology-based data matching and ontology-based data schema matching.

[9] http://protege.stanford.edu/

[10] http://knowledgeweb.semanticweb.org/

[11] http://www.openknowledgeproject.org/

The problem of *ontology-based data matching* is not the problem of *ontology matching*. The goal of the latter is to solve the problem of semantic inconsistency while integrating/merging more than *two* ontologies. The goal of the former (also the scope of ODMF) is to find the similarities between two data sets, each of which corresponds to (or can be annotated with) one part in the ontology. There is *only one* ontology in the particular problem.

There exist some data matching algorithms, as it has been always a hot topic in the data base community. Studies on *ontology-based* data matching or ontology-based data schema matching are few and limited. For instance, Topical Ontology for Directories' Editing (TODE [139]) treats ontology as a hierarchical tree, which contains only *is-a* relationship [140] do not specify domains of ontologies.

Any kinds of ontologies from different domains and contexts can be used with their approach. It is convenient and general on the one hand, but raises *problems of ambiguity* on the other hand. We intend to have a framework by taking all the possible parameters, functionalities, and semantic reasoning power brought from OE.

ODMF is designed and built based on a survey, take the most suitable algorithms, tackle their limitations such as mentioned above and design and implement a generic matching framework as such. Classical methods, such as using linguistic methods for concept searching, using WordNet [141] as the external dictionary and applying Graph Matching principles, will be included in our method. This innovative combination will be different from the effort of others.

The ODMF is a collection of different matching algorithms and strategies. A matching strategy is a combination or composition of different algorithms. Each strategy contains at least one algorithm. Figure 62 shows the design of ODMF, which contains the following components: (1) term base, (2) domain ontology, (3) application ontology, (4) matching module, (5) interpreter module, and (6) comparison module.

a. The term base contains the (multilingual) terminological information useful to describe or interpret domain concept descriptions.

b. The domain ontology contains the concepts, concept properties, and concept relations relevant for the domain.

c. The application ontology contains the ontological constraints that map information in the knowledge resource to the domain ontology.

d. The matching module provides an interface for the matching of two data elements. Different matching strategies, e.g. string, lexical, and graph, are supported by this module. Via the ODMF interface different matching implementations may be activated.

e. The interpreter module makes use of the term base, the domain ontology, the application ontology, and the matching module to interpret application data. Given two character strings Context and Object that denote (a) a concept in the domain ontology, and (b) an instance in the application ontology for that context, the interpreter will return the object ids that fit the specified objects. Conversely, the interpreter module can

translate a given object id into the combination of a Context and Object character string. During the translation, the interpreter module will take into account the specified language id. If the language id refers to English, then the Context and Object character string will be (and considered to be) in English.

f. The comparison module makes use of the term base, the domain ontology, the application ontology, and the matching module to compare two data objects. Given two sets of object ids and a strategy id, the comparison module will return a similarity score for the data objects based on domain information and using the selected strategy, i.e. matching implementation.

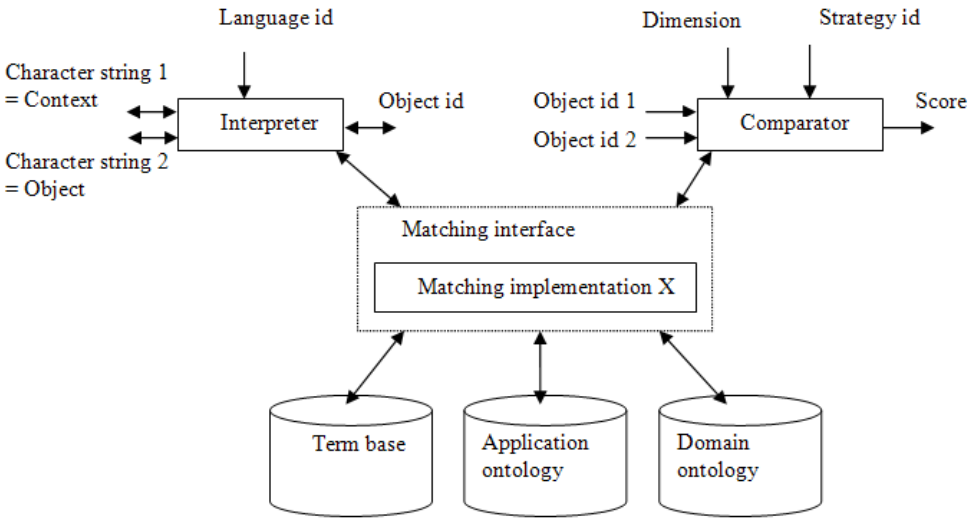

Figure 62. Design of ODMF

The ODMF tool set contains many tools, among which three are possibly useful for DIY-SE.

- *ODMF Multilingual WordNet Editor (MWE)*, which is used to manage the knowledge base of the ODMF. In particular, we use it for managing the upper common ontology and terminology. It is not restricted to any particular domains. Currently it contains over 120,000 concepts.
- *Competence Ontology Client Application (COCA)*, which is also used to manage the knowledge base of the ODMF. In particular, it allows different organizations to manage the lower common ontology. It is the implementation of three methodologies. They are DOGMA methodology [142] , DOGMA-MESS methodology [143] and PAD-ON methodology [144] .
- *ODMatcher* (Prolix Ontology-based Data Matching Tool), which is a tool to evaluate different matching strategies and algorithms in the ODMF. The end users of ODMatcher are knowledge engineers, e.g. ontology engineers.

Before explaining the software tools we will first explain how and why the terminology and ontologies are stored in several databases.

We divided the information resources into a) an upper ontology, b) a lower ontology, and c) an organisational ontology. The upper ontology contains terminological and ontological information based upon WordNet [141] . These general concepts may be used to semantically annotate competences in the lower ontology. To manage the upper ontology we developed the Multilingual WordNet Editor (MWE).

The MWE software tool (see Figure 63) allows linguists (such as, terminologists, translators and lexicographers) to manage an ontologically structured terminological database. The structure of the terminological database is a Categorization Framework [146] . The MWE makes use of the Categorization Framework API [145] to manage the terminological and ontological information. As a basis for the terminological information we imported WordNet 2.1 information.

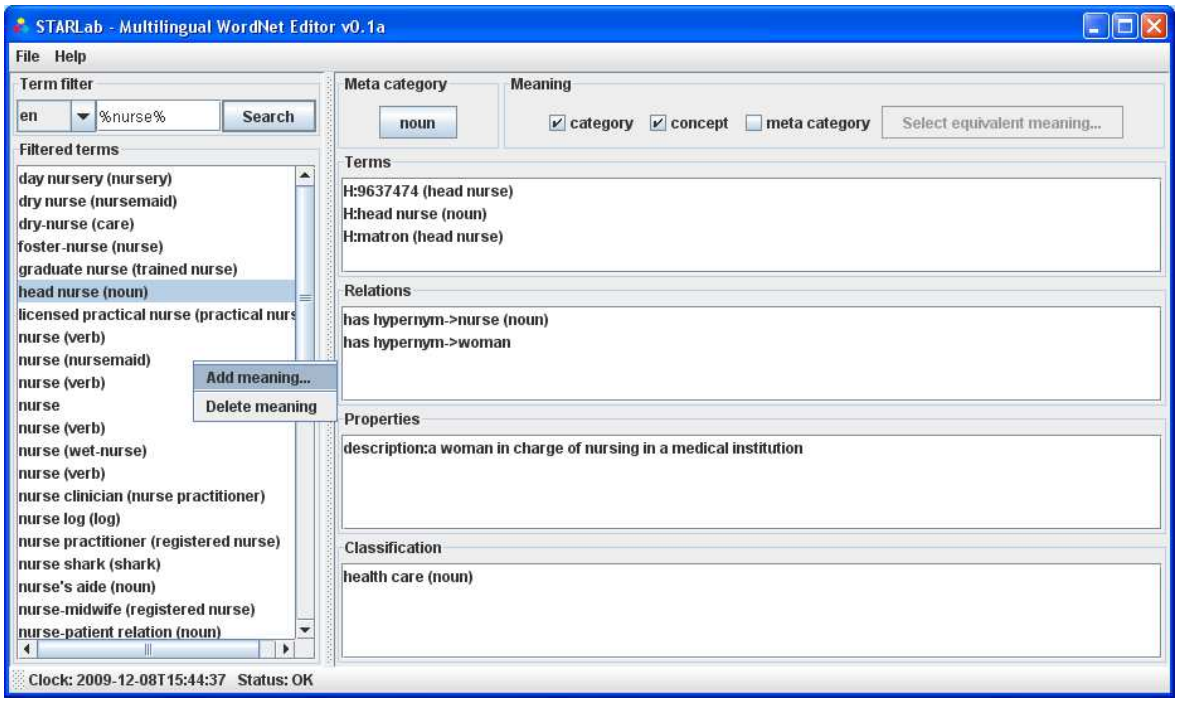

Figure 63. Managing terminology in the Multilingual WordNet Editor (MWE)

Figure 63 illustrates that a regular expression in combination with a language can be used to filter a list of terms. A term refers to a specific meaning (synset in WordNet). In the example, the term “head nurse” is selected. The term between brackets is the meta category for the meaning. As meta category the first hypernym was taken when converting the terminology from WordNet. If no hypernym existed, the linguistic category of the word (adjective, adverb, noun, and verb) was taken as a meta category. The meta category helps to disambiguate terms from the filtered list.

The *Competence Ontology Client Application (COCA)* has been used to allow organizations to manage a shared ontology for competency-based Human Resource Management (HRM). The two main concepts in the HRM domain are competences

and qualifications. Other concepts are competence level, context, competency, and qualification level.

Figure 64 illustrates for example how a qualification may be added in the ontology. For a qualification it is possible to manage the (multilingual) terminology, classify the qualification with context categories, add properties with extra information about the qualification, and to specify required initial competencies and provided end competencies.

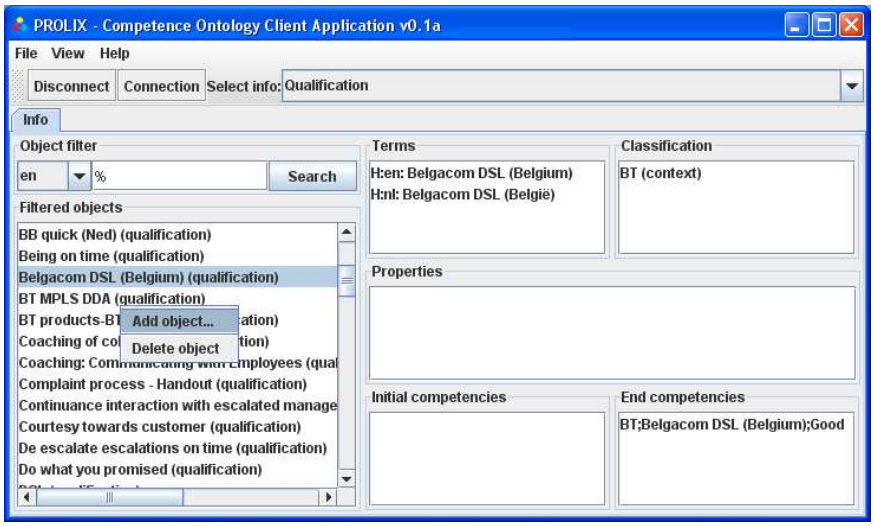

Figure 64. Managing qualifications for the HRM Ontology with COCA

The COCA is also used to semantically annotate competences. The main purpose of developing COCA was after all to allow us to evaluate different ways of semantically annotating data. Figure 65 illustrates for example how the competence “Be able to do internal escalations” may be annotated using concepts from the upper ontology. To facilitate this process, the application analyses the title and description of the competence via tokenization and lemmatization. The set of identified lemmas is then used to retrieve possible concepts from the upper ontology. The user can then simply select the suggested concepts that best describe the competence.

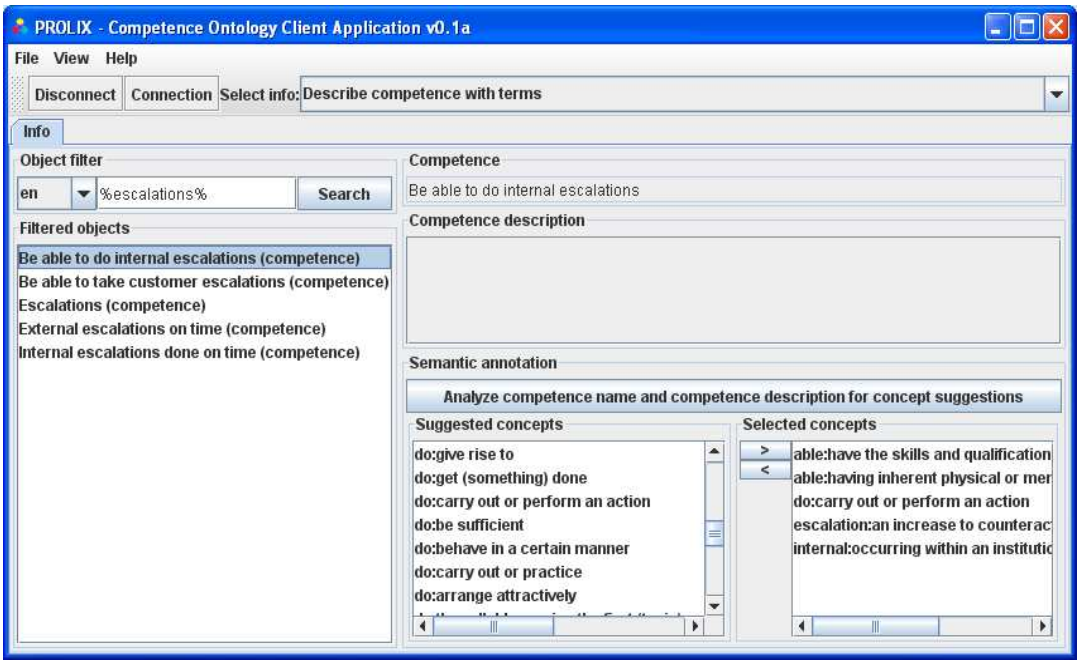

Figure 65. Semantically annotating data with the COCA

The COCA provides four means of semantically annotating data: a) using terms for concepts from the upper ontology, b) using free key concepts aided by a linguistic pattern, c) using lexons (elementary facts), and d) using semantic relations between instances.

Some examples of semantic annotations for each of these techniques to annotate the competence “Be able to do internal escalations” are as follows:

a) may be annotated with the following concepts:
   - able: have the skills and qualifications to do things well
   - able: having inherent physical or mental ability or capacity
   - do: carry out or perform an action
   - escalation: an increase to counteract a perceived discrepancy
   - internal: occurring within an institution or community

b) may be annotated with actions, instruments, locations, manners, objects, persons, times:
   - action: do
   - object: internal escalation

c) may be annotated with lexons:
   - Agent, is able to do, is ability of, Internal escalation

d) may have the following semantic relations:
   - is moderately similar to “Internal escalations done on time”

Although the scope and application domain concerning ontologies in DIY-SE are still not yet defined, it is possible to reuse the both the ontology creation methodologies and the semantic annotation methods from COCA.

The ODMatcher makes use of the information in the upper ontology, the lower ontology, and the organisational ontology. The organisational ontology contains information specific for an organisation e.g. instances of functions, persons, and tasks. The software tool allows us to manage the organisational ontology and to evaluate the different algorithms within the ODMF.

The main purpose of the ODMatcher is however to compare different data elements associated with competencies. It is possible to compare two sets of data elements with each other. In HRM domain, a data element may be a competence, a competency, a function, a person, a task, and a qualification. In DIY-SE, it can be a tool, an application component and a smart item. By pressing the Analyse button the similarity scores of two elements will be calculated (Figure 66 and Figure 67).

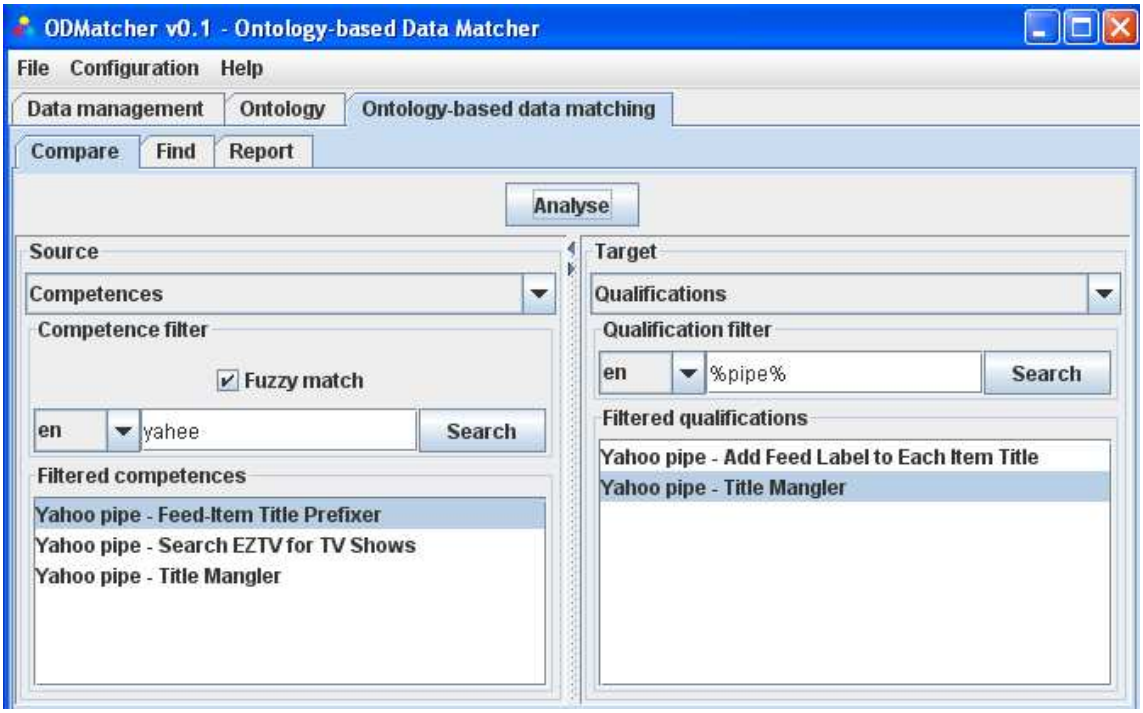

Figure 66. Comparing two data items with the ODMatcher

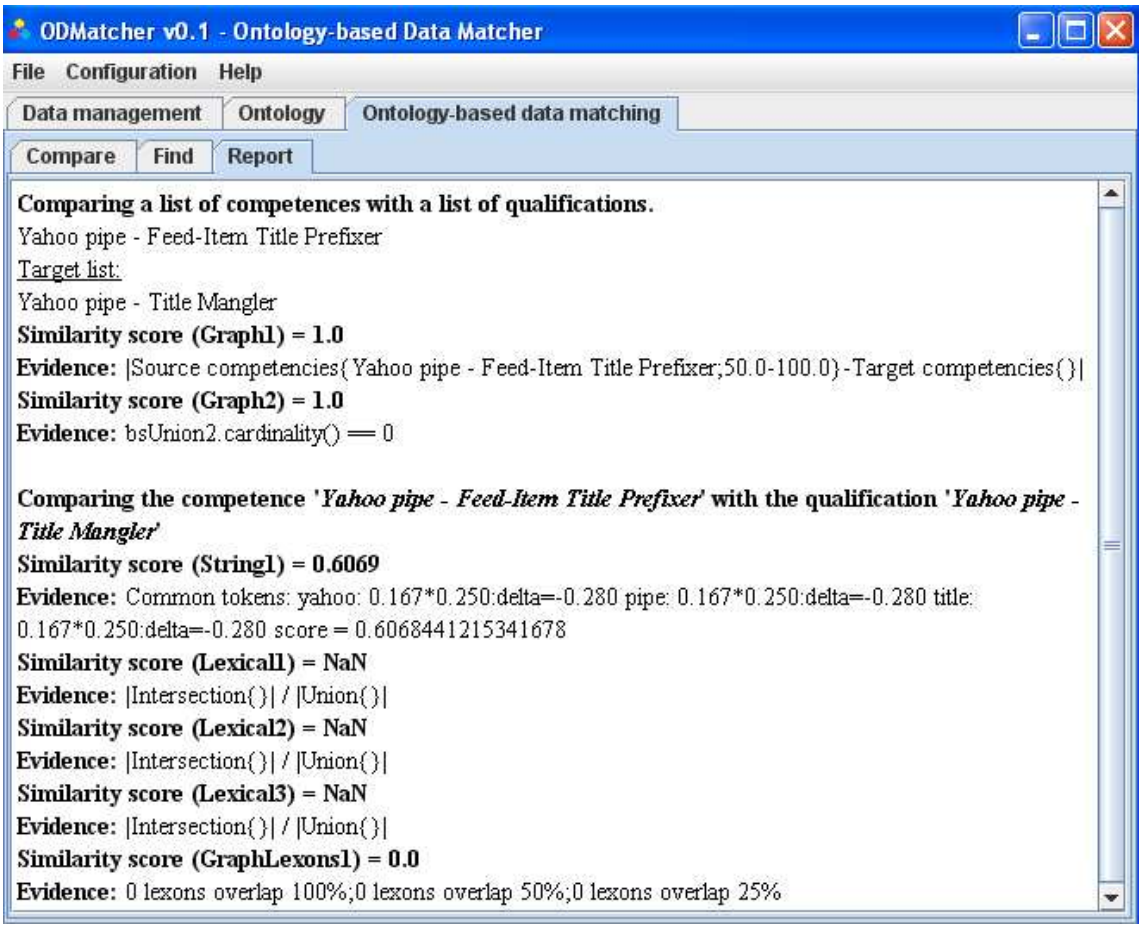

Figure 67. Matching results using different algorithms in the ODMatcher

3.4.3.4.3 Semantic Decision Table Tool Set

Researchers have been investigating the study of decision tables for more than fifty years. As an important tool to support Information System Management, decision tables have many outstanding advantages, i.e. they are easily learned, readable and understandable by non-technical people. Seeing the advantages, the interest of decision tables has been rising steadily. However, often the definition of concepts, variables and hidden (or meta-) decision rules that underlie remain implicit. When decision tables get larger, ambiguities, content inconsistencies and conceptual reasoning difficulties arise. The situation gets naturally worse when a group of decision makers need to build decision tables in a collaborative environment. Thus, the concept of Semantic Decision Table (SDT [150] [148]) is proposed.

SDT provides a means to capture and examine decision makers' concepts, as well as a tool for refining their knowledge and facilitating knowledge sharing in a scalable manner. An SDT is the result of annotating a (set of) decision table(s) (or any well structured decision resources) with (domain) ontologies. It is modelled based on the framework of Developing Ontology-Grounded Methods and Applications (DOGMA). We have designed a methodology to assist a decision group to create SDTs [150] . With regard to the technical issues of SDT, Semantic Decision Rule Language

(SDRule-L/SDRule-ML [149]) and Decision Commitment Language (DECOL) are designed and implemented to model, store, reason and publish SDT rules.

SDT tool set developed in Java is a collection of software modules corresponding to different functionalities that supports the SDT engineering cycle. It runs in the Dogma Studio Workbench.

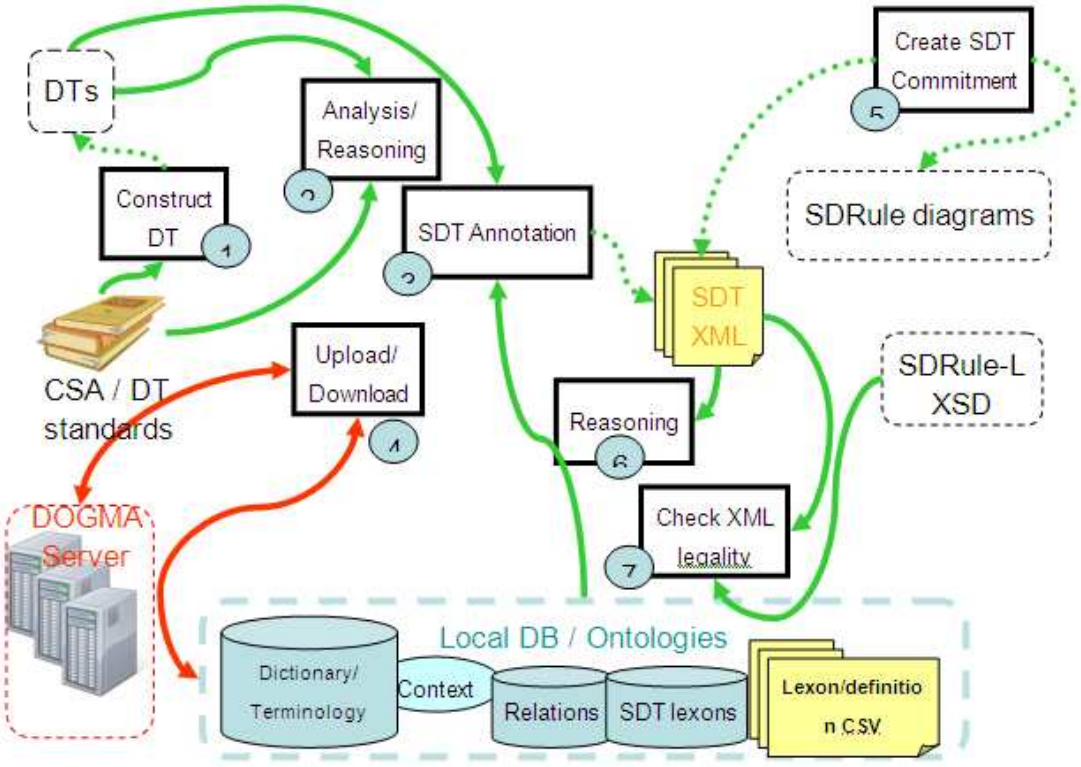

Figure 68. design of SDT tool set

Figure 68 shows the design of the SDT tool set. Figure 69 is the screenshot. On the top, it shows an SDT in the form of a decision table. In the view of “domain ontology”, the ontology is visualized as a tree. The middle view shows SDT analysis methods, each of which corresponds to a particular task. For instance, a user can check whether the SDT is complete or not by clicking “completeness”.

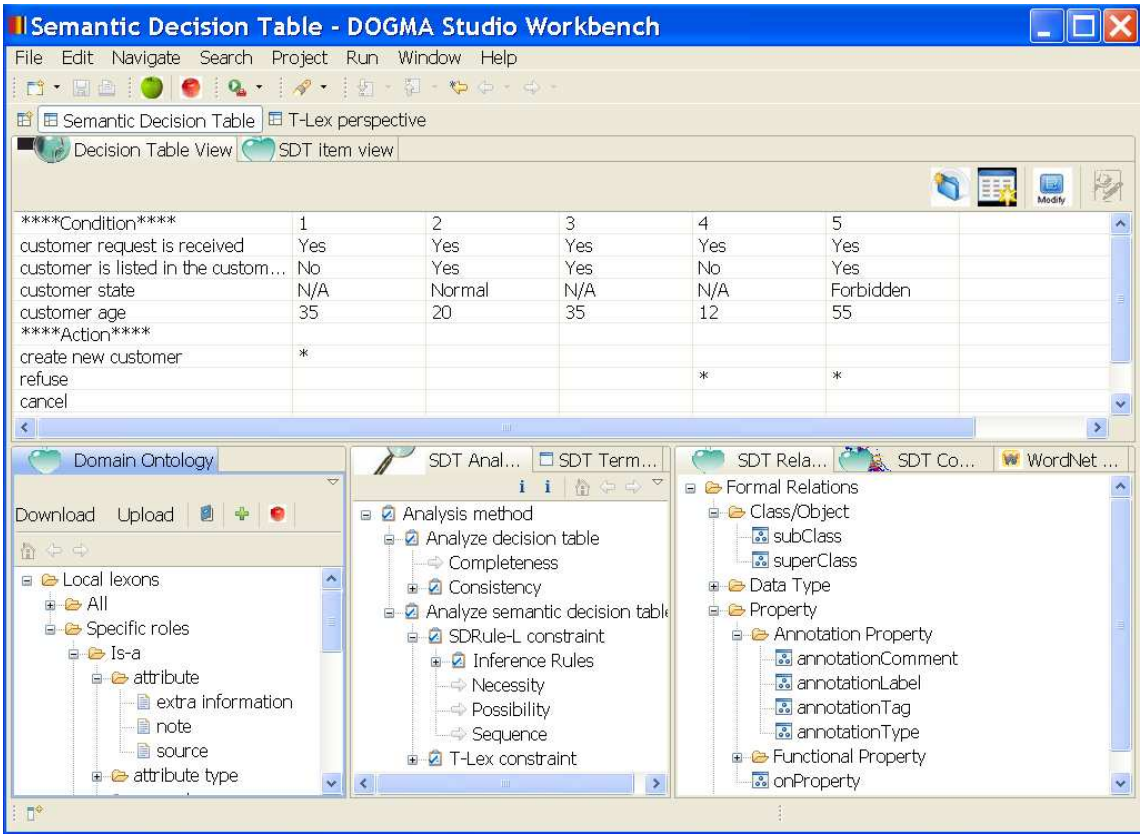

Figure 69. screenshot of SDT tool set

SDT has many advantages comparing to other decision support tools. As it is, first of all, a decision table, therefore, it contains all the advantages as a decision table.

Comparing to IF-THEN-ELSE statement, SDT can organize rules in a more compact manner. In addition, nested and dirty style of if-then-else and switch-case code is normally not recommended in software programming. SDT can associate many independent conditions with several actions in an elegant way.

Comparing to flowcharts, there are much more advantages that an SDT can bring, such as

- Clear enumeration of all operations performed
- Clear identification of the sequence of operations
- Easily learned
- Easy to construct, modify, and read
- Effective means of communication between people in and out of the data processing field; i.e., not limited to computer applications
- Concise and compact form of definition and description suitable for use in analysis, programming, and documentation

- Can be used to document applications involving complex interactions of variables
- When applied to computer systems, decision tables foster better use of subroutines, promote efficiency of computer runtime, and provide a complete data check for debugging
- Directly adapted and possibly converted directly to computer operations through symbolic logic and computer programs

There is a mathematical equivalence between a decision table and a decision tree. Comparing to a decision tree, the SDT in tabular format is much more compact than the equivalent decision tree. Although many academic researchers like to use decision trees more than decision tables, spread-sheets are more welcome by non-technical business people.

In addition, the approach to SDT supports a business community to create shared business rules and shared decision tables, which can be further “shared” by machines. As Semantic Web, Semantic Decision Tables show the advantages of semantic technologies, such as enhancing the interoperability between the software agents.

In DIY-SE, SDT can be used to store business rules (a kind of ontological commitments in OE) that make use of domain ontologies. For instance concerning the searching component, we can model rules like “if all the string matching algorithms failed, then use graph matching algorithm A” using SDT. Or, “if Yahoo pipes are available, then search component set A and B”,

4 Interaction and user interface modeling techniques

Interaction and user interface modelling techniques were originally developed to assist interface designers and software engineers in the implementation of user interfaces. The models raise the level of abstraction and allow the user interface designers to focus on the interaction issues instead of on the implementation details. The objective was to achieve products with less errors and better usability.

Modelling user interfaces at a high abstraction level has the additional advantage that the designers may discuss with end-users about the design before the actual implementation. This will only be possible if the model is at a sufficiently high abstraction level using domain-specific terms. Various tools also supported these discussions by generating mock-up interfaces from the models, that could be tested by the end-users. This is a great support for the User Centred Design approach and gives a better hands-on feeling than e.g. paper-based UI tests with end-users.

Ideally a properly modelled interface would be converted into target-code by a press of the button. In practice most tools do generate code for the user interface, but additional models and often a significant amount of software engineer effort is still needed to produce the final application.

Modelling was found to be advantageous also when designing for multi-platform applications. While software development kits facilitated programming for multiple platforms by implementing standardised widgets and UI libraries, modelling would even allow for more flexibility. For example at the advent of mobile services, various approaches were deployed to model on-line services so that web-based and mobile (e.g- wap-based) services could be automatically generated and maintained.

Researchers have also been working on modelling methods that would allow other modalities to be deployed without too much effort or fore-knowledge from the interaction designer. Thus an on-line service could be mapped to voice-interaction for the visually impaired.

Besides the actual user interface, also the service itself including the content it provides may benefit from proper modelling. A service can provide personalised features when the user's interests and way of use are properly modelled. Content rendering can be improved by a model of the target platform or device.

While most of these modelling techniques should not be visible to the end-user, many will be deployed by professionals and/or internally in the system. Therefore this chapter intends to provide an overview of modelling techniques relevant for interaction and user interface modelling. This will include the visual tools often provided for these modelling techniques, the format of the models that may also be used internally in the system, and a brief description of the techniques themselves and how the various models depend on each other.

4.1 UI modeling techniques

The design of user interfaces was initially often left to software engineers, who would build the interfaces with the technologies at hand. Out of best practices gradually the

WIMP paradigm (Windows, Icons, Menues, Pointer) developed and libraries of widgets became available to facilitate the development of (graphical) user interfaces. Software development kits (SDK's) provide support for the widget based UI contruction and often feature visual editors to construct the UI's. Software development was enhanced with modeling techniques utilizing process diagrams, finite state machines, object models etc. eventually united in the UML modeling language and tools. While UML is widely accepted and used, similar attempts for user interface modeling have not yielded such support.

Various UI modeling techniques have been proposed. The first attempts were done related to the development of User Interface Management Systems in the early 1980's [177] Early modelling focused on graphical user interfaces (GUI's), later approaches have tried to facilitate modeling UIs for multi-channel (i.e. mostly GUI's with various viewport dimensions), multi-modal, personalized and context aware use.

Szekely and others have determined the typical steps taken by designers of GUI's that should be supported by the modeling methodologies [177] :

- Determine the presentation units
- Determine the navigation between presentation units
- Determine the AIO's (Abstract Interaction Object) for each presentation unit
- Map AIO's into concrete interaction objects
- Determine the window layout

Most current modeling methodologies follow a similar, but extended approach. Paterno has strongly supported task-based modeling with his Teresa tool. Here the starting point are not the presentation units, but the tasks at hand and the objects that need to be manipulated. After the proper modelling of tasks and objects, the remaining steps resemble those of Szekely including the generation of an abstract user interface, concrete UI and implementation. The need for personalized and adaptive user interfaces has introduced also other models that can be used during the design process. Most notably the user model, device model and context model. Figure 70 provides an overview of the models used in the design process. The figure and model description have been adapted from the Nomadic Media project deliverables12.

[12] http://www.hitech-projects.com/euprojects/nomadic-media/

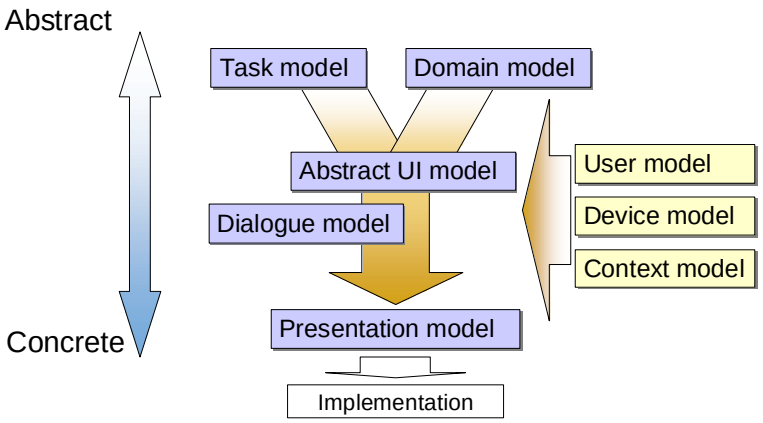

Figure 70. Models used for UI design

The task model will generally include a description of the goals to be achieved and the tasks that must be performed in order to achieve that goal. A description of the task sequence, optional and parallel tasks is often also included.

The domain model captures the concepts relevant for the domain of the application. In its most concrete form, this may take the shape of a class hierarchy, with the classes representing domain items.

The abstract UI model is a collection of abstract interaction objects (AIOs) that represent for example the need to obtain information from the user (input) or convey information to the user (output), or an aggregation of other AIOs. The abstract UI cannot be rendered on a device, but needs to be converted to a concrete user interface, with concrete interface objects (CIOs) first.

The presentation model is the concrete user interface as rendered on a device. The UI consists of concrete interface objects (CIOs) most often represented by widgets. The presentation model describes how the CIOs are rendered and thus contains information on the layout strategy, the look-and-feel, the voice type, etc. Sometimes the concrete interaction objects are separated from the presentation model into a concrete UI model that can still be adapted to the final rendering device. This final adaptation can be done for example by style-sheets. The final implementation in a specific target language is no longer a model, but follows from the presentation model.

The dialogue model may be part of the abstract or presentation model, but is sometimes also defined separately. It captures the flow that input is expected or information is presented to the user. This is particularly important for speech-driven UI's, but a properly designed dialogue can also improve traditional UI's that are composed of several views.

The user model, device model and context model capture features of the user, the rendering device and the user/device context, respectively. None of the models is a part of the actual user interface definition, but they significantly affect the generation of a concrete user interface from the abstract models. Design decisions and automatic generation rules are based on these considerations.

Below a short summary of the most relevant modeling methodologies and tools available.

4.1.1 Teresa

Teresa is a methodology and tool developed by Paterno et al at the HCI Group of ISTI-C.N.R. since the 90's 13 . Its starting point is a task model called ConcurTaskTrees. Recent developments have also included mapping of multimodal UI and the newest version is called Multimodal Teresa.

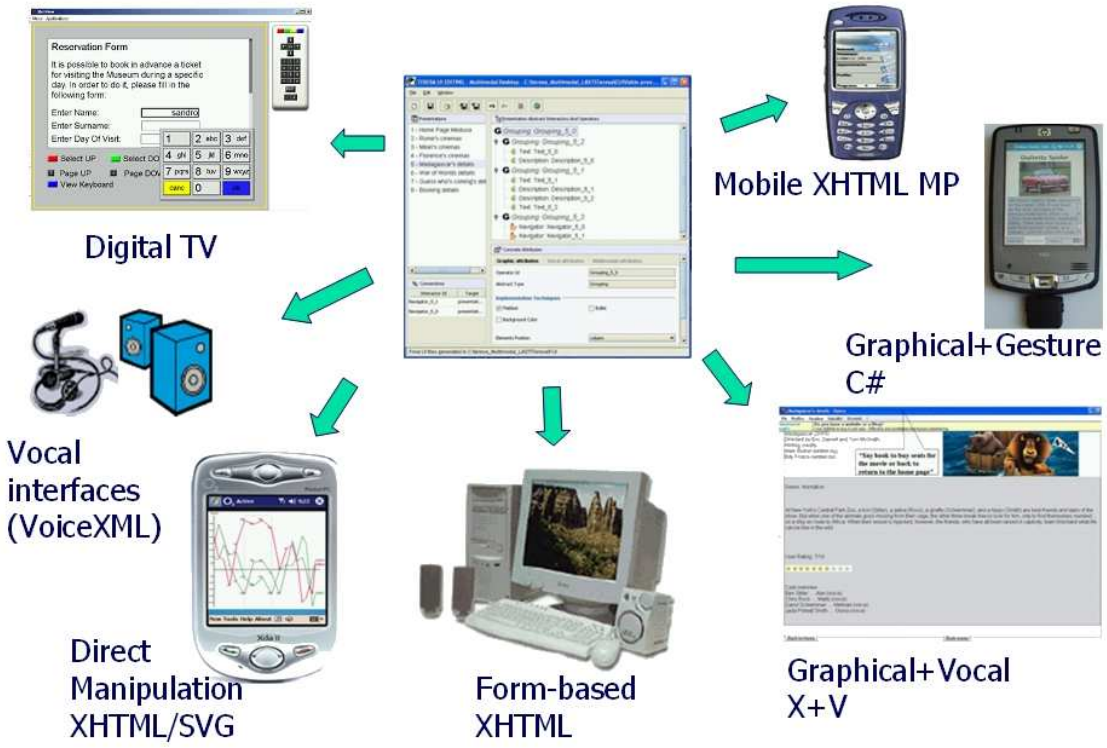

Figure 71. MULTIMODAL TERESA, a transformation-based environment, supporting multimodal interfaces

Multimodal Teresa is intended to provide a complete semi-automatic environment supporting a number of transformations useful for designers to build and analyse their design at different abstraction levels and consequently generate the user interface for a specific type of platform.

The Teresa approach starts from the CTT task model. It generates presentation sets and transitions from the model, and these combined with the tasks are used to generate an abstract user interface description. From there on, target-specific attributes are added and culminate in the final user interface. The process of UI definition according to Teresa is presented in Figure 72. Here CTT are the concurrent task tree definitions, the AUI is the Abstract user interface, the CUI are concrete user interfaces for desktop or mobile use, which are finally implemented in the target

[13] http://giove.isti.cnr.it/tools/TERESA/index.html

specific language.(XHTML in this example). This process picture is taken from early research, but the process has remained the same.

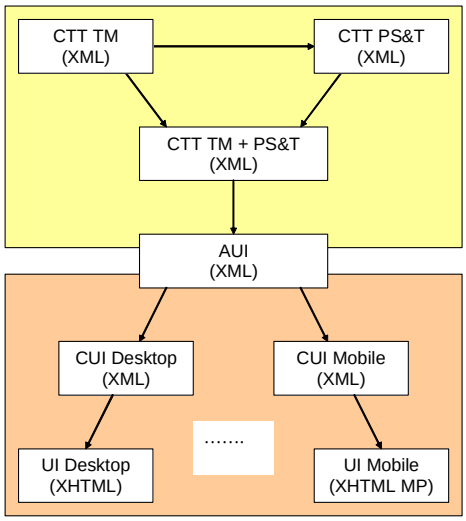

Figure 72: Teresa design process

The abstract user interface elements are taken from a set of specified elements and constructs listed in Figure 73.

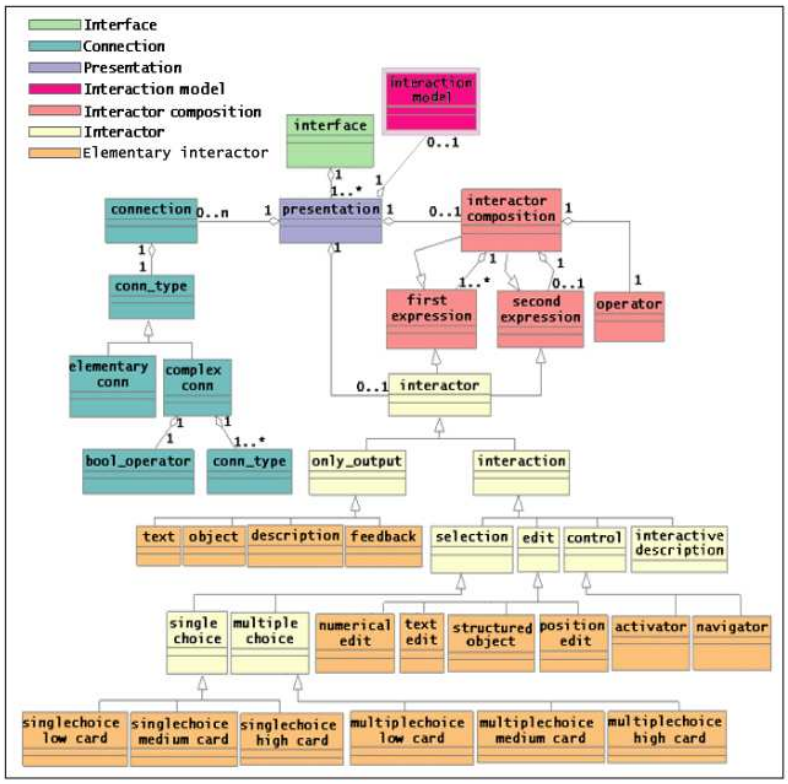

Figure 73. Teresa Abstract UI model elements

More information about Teresa can be obtained from the web site13 and the publications made on the methodology and tool, most notably Paterno's book [175] and a recent journal paper [176] .

4.1.2 UIML

The User Interface Markup Language [179] is a meta-language for providing a representation of a UI that can be mapped to existing languages such as Java or HTML. UIML acts as a framework for defining your own user interface vocabularies that are made up of interface parts (placeholders for user defined UI elements), properties (of UI elements), and events. The goal of UIML is to "develop a specification for an abstract meta-language that can provide a canonical XML representation of any user interface (UI)" [178] UIML was first developed by the Harmonia company (starting 1997)[180] and later enhanced in cooperation with the Human-Computer Interaction center at Virginia Tech. The standardisation of the language has been taken over by the OASIS technical committee (OASIS User Interface Markup Language TC).The current version is 3.0, while a draft of 4.0 is available..

4.1.2.1 Methodology and coverage

UIML is a meta-language and therefore only provides a structure to embed the UI models. Actual implementations need the specification of a vocabulary, that contains the details of the target format. The model used by UIML resembles that of a typical widget model for graphical user interfaces, although it allows for further extension. The amount of detail in the UI description very much depends on the vocabulary and the parameters that developers will specify in their applications. Typical examples from Harmonia include very target-specific language constructs, and therefore include e.g. device specific graphics or VoiceXML constructs. Also several device-independent vocabularies have been developed [181] .

The UIML approach depends on a renderer, that is capable of rendering the UIML description either to a known UI language or directly as e.g. a Java UI. Renderers for various target languages have been constructed in research projects and are available for e.g. C++, HTML, Java, .NET, Symbian, QT, Visual Basic, VoiceXML and WML.

4.1.2.2 Presentation and structure

A typical UIML document will consist of sections containing a UI structure, UI style, and a UI behavior. The structure section of the document defines the composition of UI parts, where each part provides a name of a UI element (as defined in a vocabulary). The style section is then written to provide the properties of each of the declared UI elements along with a description of the mapping (using a peers tag) of the UI parts to concrete components (such as Java widgets or HTML constructs), as well as style attributes such as Fonts. A peers definition can contain a collection of different mappings to different target components. Each mapping is indicated by using a presentation tag. Finally, a behavior tag can be used to provide rules for communicating with a host application at runtime.

Tools

There are various rendering tools available for UIML, but no proper design environment. UIML's philosophy is not to function as a design language per-se, but as a mediator between environments. As such it should be regarded mostly as an exchange format, and less as a design methodology (although some research has taken this approach as well).

4.1.3 UsiXML

UsiXML (which stands for USer Interface eXtensible Markup Language) is a XML-compliant markup language that describes the UI for multiple contexts of use such as Character User Interfaces (CUIs), Graphical User Interfaces (GUIs), Auditory User Interfaces, and Multimodal User Interfaces. In other words, interactive applications with different types of interaction techniques, modalities of use, and computing platforms can be described in a way that preserves the design independently from peculiar characteristics of physical computing platform[182] .

UsiXML was developed by Limbourg, Vanderdonckt et al in the Cameleon project and followups [183] The initiative follows a similar design philosophy as Paterno's Teresa. It provides models for domain concepts and tasks, abstract user interfaces, concrete user interfaces, context information, resources, and transformations. UsiXML has been supported by a large range of tools. GrafiXML is a graphical tool for designing graphical user interfaces for various platforms that stores its design in UsiXML, see Figure 74 for an example. The VisiXML is a similar tool built on the Microsoft Visio environment. ScetchiXML utilises a scetch board for designing a rough interface, which can then be refined with other tools.

. .

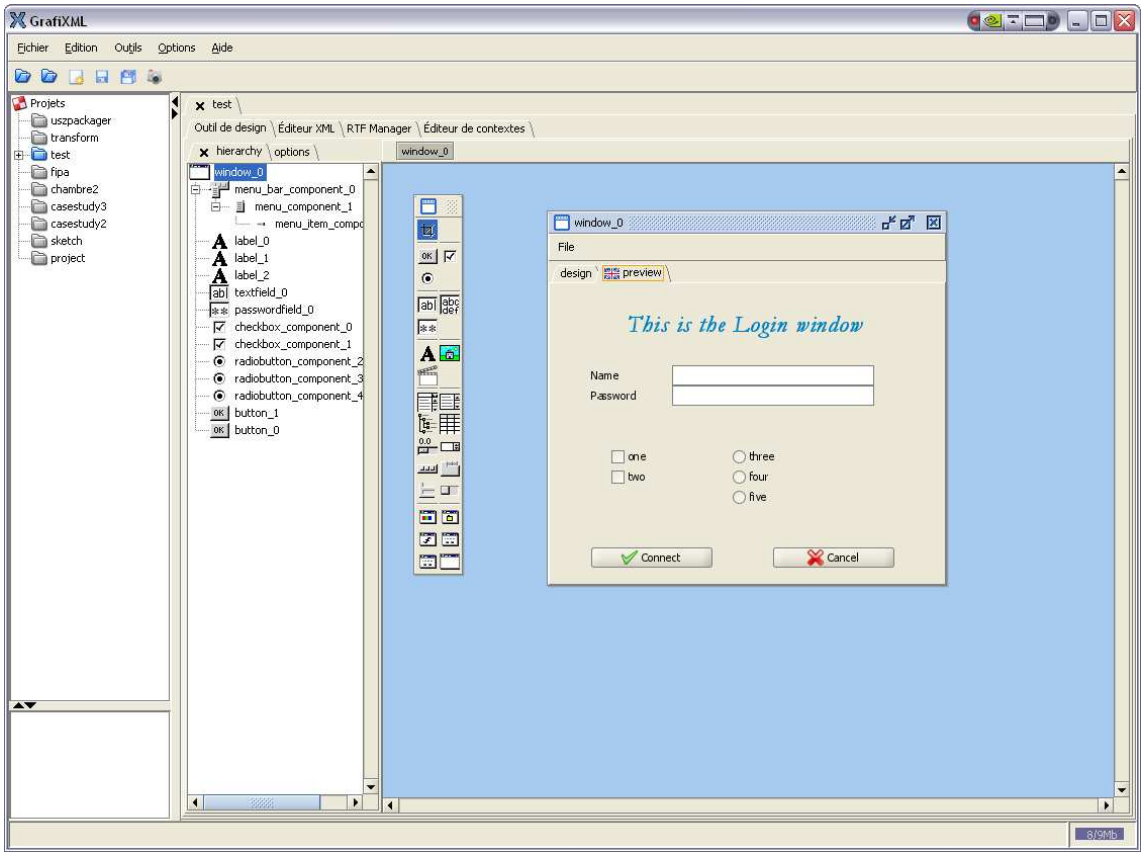

Figure 74. The GrafiXML tool for UsiXML editing and UI design

The IdealXML environment supports building user interfaces using patterns and helps to create abstract user interfaces from task models (Figure 76). The transformation process can also be assisted by tools like KnowUI, which incorporate the designer's knowledge, TransformiXML, a generic transform tool using formal graph mapping algorithms.

Code can be generated by means of plug-ins for the environments, for example to Java, or XHTML. The UI's can be rendered also by means of RenderXML. Various interpreters also help to render UsiXML on e.g. SVG, Flash or Tcl-Tk, or even into a haptic environment.

Several projects are exploiting UsiXML beyond the originally intended scope. There are technologies for using it in 3D environments, for spitting the interfaces across different platforms and for designing UI's for work processes, to mention a few. Work on rendering the UsiXML interfaces onto multimodal platforms has also been done (MultimodalXML). Figure 75 gives a good overview of the typical model transformations used in UsiXML, in this example for multimodal rendering. Tools for verifying the UsiXML models and checking for proper implementation of usability rules also exist.

UsiXML is supported by many tools, as shown. It supports the methodology of designing user interfaces starting from task and domain models, via abstract and concrete UI's, but is flexible enough to also support different design paths.

UsiXML is being proposed for standardisation. For this purpose, the W3C model-based user interfaces incubator group charter was formed. Parts of UsiXML may be used for these standardisation efforts. This W3C action initiates and pursues efforts of defining a common UIDL. The mission of the Model-based User Interfaces Incubator Group, part of the Incubator Activity, is to evaluate research on model-based user interface design as a framework for authoring Web applications and with a view to proposing work on related standards.

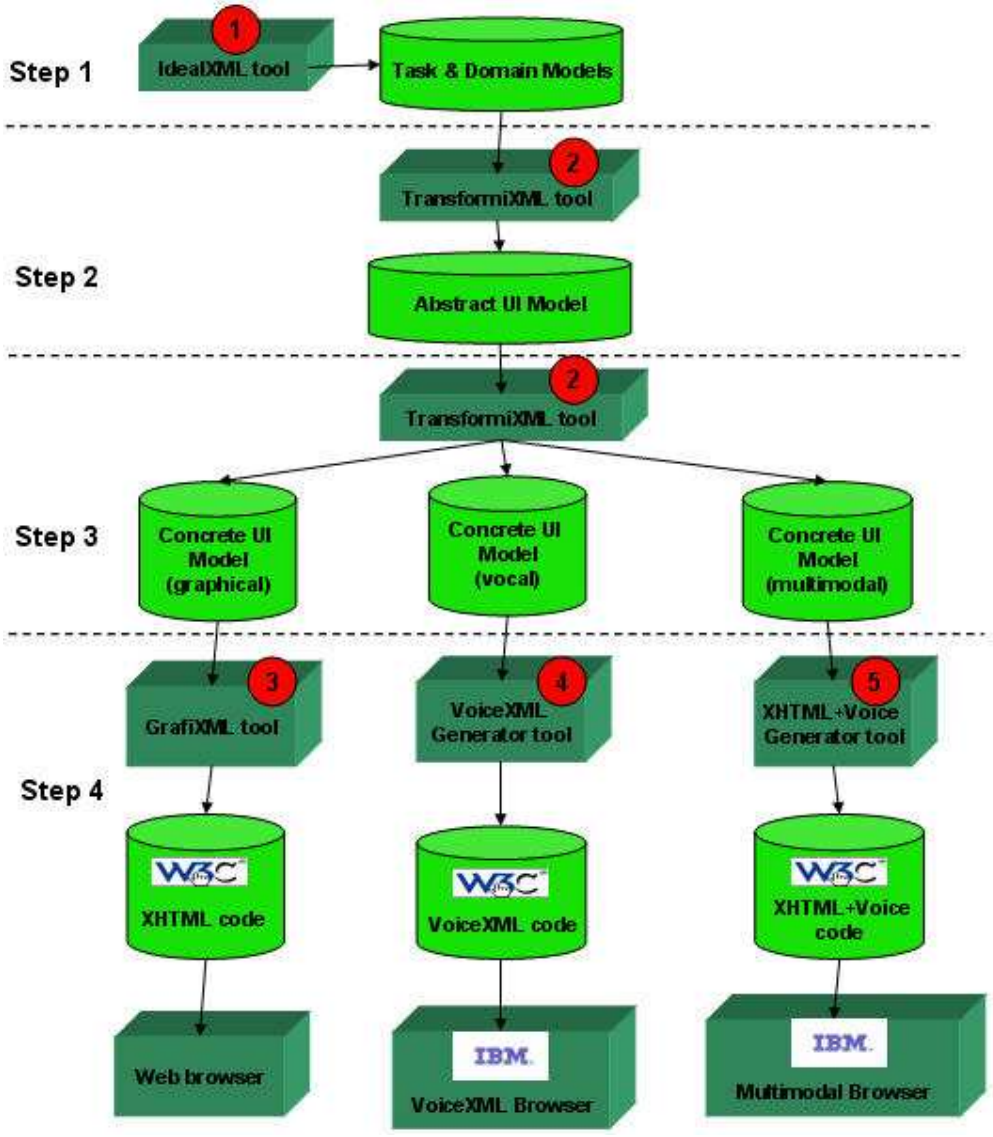

Figure 75. UsiXML model transformations for multimodal interfaces

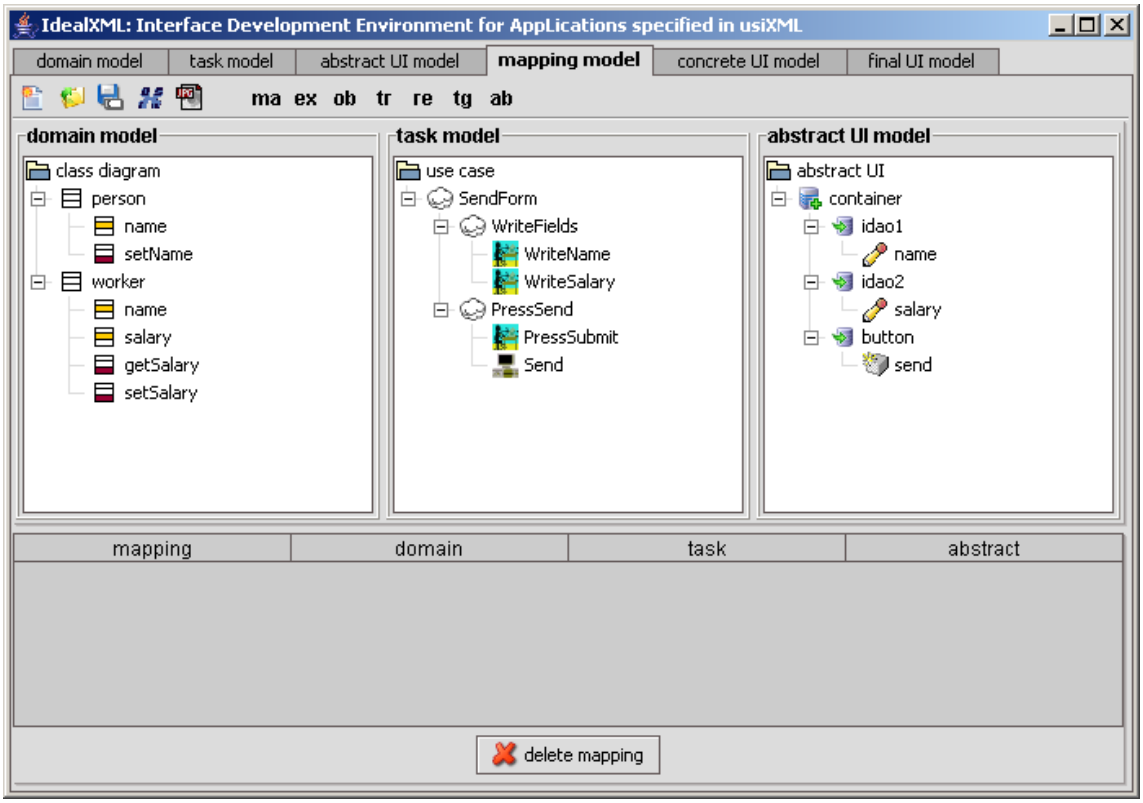

Figure 76. Example of the IdealXML environment with models at three levels of a UI

4.2 GUI modeling

Modelling of graphical user interfaces may be part of an SDK, which typically provides visual tools for modeling a GUI by means of adding widgets to panels and windows. Sometimes also support for the flow of dialogs (e.g. for mobile UI) is provided. As an overview of these tools is provided elsewhere, we will not address them here.

Modelling GUI's at a higher level of abstraction is particularly useful when the features of the final interface are not completely known. The flexibility envisioned by HTML, which can be used with a great variety of viewport dimensions, is a good example of this (we refer to the original basic HTML without the extensive use of frames, tables and clickable graphics used nowadays). Perhaps this is the reason, that user interfaces have been implemented in web-sites, first using forms, later XForms, Flash, Java applets, etc.

An interesting example of GUI modeling is the XUL format developed by Mozilla. Also the concrete user interface models of the various UI modeling techniques described in the previous section provide good examples.

4.2.1 XUL

The eXtensible User interface Language (XUL) [184] is an XML based language developed by Mozilla for describing window layout. The goal of XUL (and indeed Mozilla's general strategy) is to build cross-platform applications such as web browsers that can easily be deployed to several PC operating systems. The look and feel of UI elements are separated by programming logic and presentation. The presentational aspects consist of a stylesheet and a collection of images to form a "skin". This enables application developers to work independently of UI developers.

A XUL UI consists of a set of structured interface elements (windows, menubar, button etc.) along with a predefined list of attributes. In order to make these elements interactive however, a separate scripting language is required.

XUL relies on multiple existing web standards and web technologies, including CSS, JavaScript, and DOM. Such reliance makes XUL relatively easy to learn for people with a background in web-programming and design. XUL has no formal specification and does not inter-operate with non-Gecko implementations. However, it uses an open source implementation of Gecko, tri-licensed under the GPL, LGPL, and MPL. XUL provides a portable definition for common widgets, allowing them to move easily to any platform on which Mozilla applications run.

4.2.1.1 *XUL documents*

Programmers typically define a XUL interface as three discrete sets of components:

- content: the XUL document(s), whose elements define the layout of the user interface
- skin: the CSS and image files, which define the appearance of an application
- locale: the files containing user-visible strings for easy software localization

4.2.1.2 *XUL elements*

XUL defines a wide range of elements, which roughly belong to the following types:

- top-level elements, such as window, page, dialog, wizard, etc.
- widgets, like label, button, text box, list box, combo box, radio button, check box, tree, menu, toolbar, group box, tab box, colorpicker, spacer, splitter, etc.
- box model, e.g., box, grid, stack, deck, etc.
- events and scripts, for instance: script, command, key, broadcaster, observer, etc.
- data source, e.g., template, rule, etc.
- others, e.g., overlay (analogous to SSI, but client-side and more powerful, with higher performance), iframe, browser, editor, etc.

One can use elements from other applications of XML within XUL documents, such as XHTML, SVG, and MathML.

While XUL serves primarily for constructing Mozilla applications and their extensions, it may also feature in web applications transferred over HTTP.[14]

4.3 Abstract UI

The need to get access to information anytime, anywhere, from different kinds of devices is no more restricted to a few privileged groups of people. This need has become very common in everyday life of general Internet users. Laptops, personal digital assistants (PDAs) and cell phones are the prototypical examples of personal access devices. Each device has its own particularity in terms of input (keyboard, handwriting recognition, speech) and output capabilities (size of screens). Because of this diversity, more than often, service providers need to develop user interface for each kind of device individually. It requires a lot of effort and resources specialized in application development of each device in order to make the provided service available on multiple devices.

Abstract User Interface refers to the abstract representation of user interface (by mean of an abstract UI representation language). This abstract representation does not assume any particular modality (visual, aural, tactile, etc) or form of delivery to the user. It should ideally provide all of the service-specific information necessary to build a user interface for any user. Given a sufficiently expressive abstract representation of a service, user interfaces for multiple contexts could be developed quickly, using support tools. They could even be generated entirely automatically. Abstract representations do not capture all of the information that a human designer would use in creating a high quality user interface, so generated user interfaces based on such representations would typically be less polished.

An abstract UI model is a definition of a user interface at a high level, which does not contain any implementational details. In a typical UI modeling environment, the abstract UI will result from the task model and the domain model and provide an intermediate step towards the concrete user interface. An abstract UI is composed of abstract interaction objects (aio's), which define the purpose of the interactor, but leave the decision of the implementing widget to the next step. Ideally an abstract UI could be translated to any user interface implementation and modality. Depending on the purpose of the modeling system, the level or abstraction can be very high, or already contain some hints (e.g. structure) of the implementation.

4.3.1 Qualities of an abstract UI representation language

An abstract representation of UI should meet the following requirements:

- Applicable to any target: Potential targets include physical devices such as home and office appliances or public information kiosks and vending machines; virtual services such as airline reservations, currency exchange, online shopping and directory services; and software applications such as email, spreadsheets and games.

[14] This description of XUL is partly taken from Wikipedia

- Applicable to any delivery context (delivery context is the combination of user, environment and device): The abstract representation should take care of user's expertise, age, culture, health, and language. It should also take into account the physical device's physical form and input/ output capabilities.
- Simple, extensible, flexible and personalizable.

Provide a means to personalize the UI is vital for the usability of an abstract UI representation language. It should be flexible and simple enough to allow making personalized UI for different groups of user and devices.

4.3.2 Principle abstract UI representation languages available

Main UI representation languages available at the time of the writing of this document are the following [164] :

- User Interface Markup Language (UIML)
- Extensible Interface Markup Language (XIML)
- XForms
- Alternative Interface Access Protocol (AIAP)
- AUI in the Teresa environment
- AUI part of UsiXML

Here it should be noticed that UIML needs a vocabulary to act as abstract UI, as it provides only a meta-structure for the interface. XIML is a closed initiative by RedWhale. Xforms are mainly intended for use in web environments, and therefore contain a rather application specific interface definition.

The elements of a typical AUI can be shown in a class hierarchy, Figure 77 shows an example of the UsiXML AUI classes.

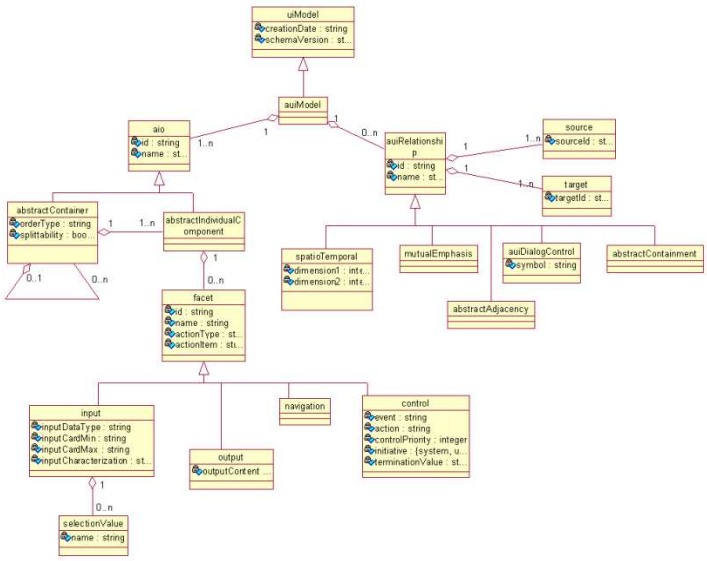

Figure 77. Class hierarchy of the UsiXML AUI elements

4.4 User modeling

A user profile is a (structured) data record, containing user-related information including identifiers, characteristics, abilities, needs and interests, preferences, traits and previous behavior in contexts that are relevant to predicting and influencing future behavior [1] .

User profile can be described as a set of data representing the most relevant features of the user. This user profile can be exploited to provide relevant, personalized and context-sensitive information delivery, helping to provide personalized access to resources and services and also to identify people with related interests.

User profiles can be classified as following:

- Static, means it contains information that rarely or never changes.
- Dynamic, refers to the data that changes frequently.
- Explicit, means that the information is obtained explicitly, using online registration forms and questionnaires resulting in static user profiles
- Implicit, refers that the information is obtained by recording the navigational behaviour and or the preferences of each user.

4.4.1 Challenges

The main challenge is to present the information that users want in a way that makes sense to them. In traditional approaches for user profiling, the user has to explicitly create the profile, and manually keep it up to date. However, getting personalized information “anytime, anywhere and anyhow” is not an easy task. The user profiles should be evolved naturally with ongoing changes in user behaviour and preferences patterns.

The main obstacles to user profiling can be split in organizational and user obstacles.

At organizational level:

- **User** has an important role in the way the business process is designed and implemented. If this is not contemplated in the business strategy and processes, a redesign will be necessary, implying costs.
- **Responsibility**: It is important to define who is responsible for what, especially when more than one organization make use of the same user profiles. Issues such as who will keep the user profile up-to-date or who is entitled to make changes are relevant aspects to be solved.
- **Legal obstacles**: Some political regulations govern the legal conditions for public organizations engaging in user profiling. These regulations are different from the market regulation guiding the private sector.

At user level:

- **Access**: users need to have access to communication and information technologies in order to be able to use their user profile

- **Acceptance**: user has to accept the use of user profiling. For doing so, it is very important the trust, control and privacy issues.

4.4.2 State of the art

The use of profiles for personalization is not new, but current initiatives and tools rely on an explicit, manually entered user profile; so these use profiles go out of date very quickly and lack relevance.

- The main drawbacks user profiling is facing are:
- How to mine accurate user profiles from observed behaviour?
- How to deal with the changing nature of user interests?
- How distinguish between long term and medium term interests?

The current available standards for user profiling are the following:

- *vCard* specification from the Internet Mail Consortium is a means of Personal Data Interchange (PDI) [2] , which automates the traditional business card. It can be used to store mandatory directory information (name, addresses, telephone…), geographic and time zone information, and also graphics and multimedia. The vCard specification has multiple language support and is transport and operating system independent (based on RFC 2425 and RFC 2426).
- *IMS Learner Information Package (LIP)* specification [3] offers a data model describing the user characteristics required for the general purpose of recording and managing learning related history, goals and accomplishments; engaging the user in a learning experience; discovering learning opportunities for user.
- *IEEE Public and Private Information (PAPI) Specification [4]* represents student records and its development is moving towards harmonisation with IMS. It specifies data interchange formats, facilitating communication between cooperating systems. User records are divided into personal information and performance information and these are maintained separately. A key feature of this standard is the logical division, separate security, and separate administration of several types of learner information.
- *Global TV-Anytime Specification*: The TV-Anytime Forum [5] is an association of organisations that seeks to develop specifications to enable audio-visual and other services based on mass-market high volume digital storage in consumer platforms. The TV-Anytime Metadata specification employs metadata to describe content, user preferences, consumption habits, for targeting a specific audience.

An overview of methods for building a user profile can be found in [6] . User modelling issues and guidelines are also presented in [7] , concentrating on modelling of user knowledge, plans, and preferences in a domain. It focuses on stereotype (as opposed to *individual*) profiles.

In summary, both academia [8] and enterprises [9] have experimented with user profiles for personalization. [10] and [11] allow to create implicit user profiles by looking over the shoulder of a user performing their usual tasks (including email, document management or web browsing. User does not have the obligation to manually create a profile, so data collection is easier and the risk of 'data entry fatigue', or inaccurate profiling, is reduced. [12] and [13] are works focused on collecting terms from visited web pages rather than tags applied by a user.

As we have mentioned, **dynamic** user profiles are those, which update as the user task is changed. [12] also presents such an approach using a hierarchical organization of users' interests. [14] uses a more graph-like representation. In both works an artificial change of task context is employed.

4.5 Task modeling

The highest abstraction level used for designing user interfaces is the task level. A task could be described as an elementary step that the user or the system is performing during the planned interaction. Therefore tasks are easy to comprehend and task models can also be discussed with non-designers.

The best known task model is the Concurrent Task Trees notation developed by Paternó et al. According to Paternó: "CTT is a notation aiming at supporting engineering approaches to task modelling". CTT contains a set of operators that allow rich definitions of tasks and their relations. The tasks, with the aid of CTTE and TERESA tools, can be presented as graphical task trees, which present the relations between tasks with arcs and graphical notations. An example task tree is presented in Figure 78.

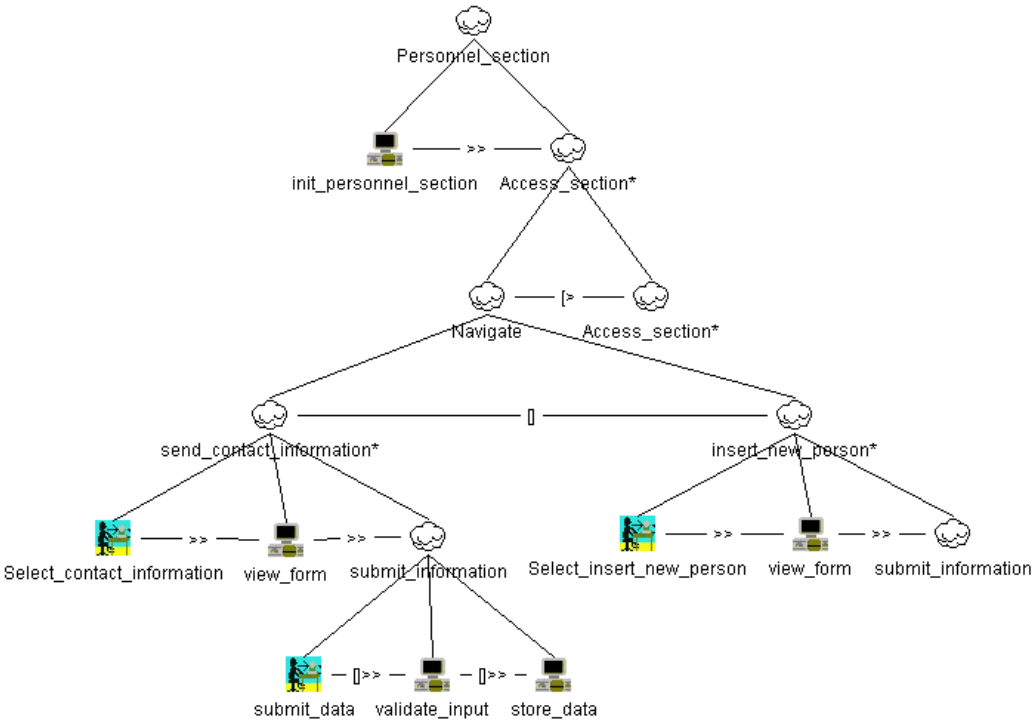

Figure 78. A partial CTT model

To date, CTT seems to be the richest and most widely recognised task modelling technique. The notation is also used in UsiXML.

4.6 Semantics in interaction modeling

Semantic Web services, designed to be easily composed software powered resources for use in applications, provide simple interfaces to functionality that can be very useful for human users. Ontologies are use to explicitly formalize the properties and structure of contextual information to guarantee common semantic understanding among different architectural components.

The applicability of ontologies for context modeling and facilitating application reasoning has been well researched [111] [112] . Specifically, the OWL-DL (Web Ontology Language- Description Logics) formalism, rooted in the decidable fragment of first-order logic, provides a powerful platform for a formal and machine-processible structure to context information collated from diverse sources [113] .

With an ontology-based context model providing a common, formalized structure, a number of interconnected components are required to provide a generic mechanism for context querying and reasoning.

Upper level ontologies and domain ontologies comprise many occurrences of a variety of ontology design patterns (OPs) [114] . These ontologies are generally large and densely axiomatized. Therefore, the development of dedicated application programming interfaces (APIs) instead of generic solutions like RDF or OWL APIs eases the adoption of such ontologies.

4.7 Multimedia documents modeling

The multimedia documents modeling is used for many applications and services using a variety of media like multimedia authoring, indexation, retrieval…

A model of multimedia document is a model of document which serves to describe different aspects for a multimedia document including logical, spatial, hypertext structures and temporal dimension.

The description of a media corresponds to elementary and\or composed concepts.

The elementary concepts or the "low level" correspond to independent descriptive fields. For example, the areas of dominant colors, the histogram of images, the author's name of the media...

The composed concepts correspond to the relationship between descriptive fields. The value of these fields can be specified or not. The nature of the relations ("consists of ", "belongs in ", "is the son of ") can be also specified or not.

There are a significant number of standard models of multimedia document such as HTML, HyTime [165] , MHEG-5 [166] and SMIL [167] as well as no standard models as OCPN [171] , CMIF / CMIFed [168] , Madeus [169] , ZYX [170]

There are also other models for specific applications as "A Synchronization Model for Hypermedia Documents Navigation" [172] and "The LimSee3 Multimedia Authoring Model" [173]

These various models were detailed in the multimedia literature as MHEG ISO Standard [166] and "A Comparison of Multimedia Document Models" publication [174] .

Hereafter, a comparison of main multimedia document models Figure 79 from [174]

	HTML	DHTML	SMIL	MHEG-5	HyTime	ZYX
Temporal Model	-	script	interval-based	event-based	point-based	interval-based
Spatial Model	absolute positioning	absolute positioning	absolute positioning	absolute positioning	absolute positioning	absolute positioning
Interaction						
Navigational	+	+	+	+	-	+
Design	-	+	-	+	-	+
Reusability						
Granularity						
Media Elements	+	+	+	+	+	+
Fragments	-	-	-	-	+	+
Documents	+	+	+	+	+	+
Kind of Reusage						
Identical	+	+	+	+	-	+
Structural	-	-	-	-	+	+
Identification/Selection	+	+	+	-	+	+
Adaptation						
Parameters of Adaptability						
User Interest	-	+	-	MHEG-6	-	+
Technical Infrastructure	-	+	+	MHEG-6	-	+
Definition of Alternatives						
Static	-	+	+	MHEG-6	-	+
Dynamic	-	-	-	-	-	+
Presentation-neutral Representation						
Multimedia Functionality	very low	high	medium	very high	low	high
Semantic Level	medium	very low	medium	low	very high	high

Figure 79: Comparison of main multimedia document models

Several languages are created to describe multimedia documents:

The group W3C: XHTML, SMIL, SVG,

The group MPEG: MPEG 4 BIFS, MPEG 4 Laser,

Other group: Adobe Flash.

5 Flexible environment for interface creation

5.1 Rich interactive experiences in smart environments

The evolution of HCI to environments that are able to adapt themselves to the user needs and to provide customized interfaces to the services available at each moment (smart environment) is represented by RIE (Rich Interactive Experience). From user's point of view, this interaction is converted in an Interactive Experience when the space is adapted by the context [88] and user's preferences to let the users to perform monotonous tasks in an easier way. In other words, users don't limit their actions to carry out some tasks, but they are able to change their environment to suit their preferences and access the available services.

A smart environment system is based on a set of devices, which gather information about the environment (sensors), and a set of devices able to actuate over the environment to change its conditions (effectors). At this point, the user-system interface acquires a high importance because it provides a certain degree of interactivity that the smart space will employ to adapt the environment. The preferences of a given user may be unknown to the system; for example, when it is the first time a user enters a given space. Even if the user preferences are known, they may change over time, or they may be affected momentarily by facts unknown to the system. Sooner or later the user will need something different than what the system thinks she needs, and there must be a way for the user to express that to the system. Furthermore, the system may needs communicate with the user.

Rich interface represents a communication way, between users and environments, more expressive for the final users that allow communicating essential information only. The usability concept is present in the correct design of this kind of UI (user interface) because it is a feature that determines the help offered by the environment to users to complete some task, so helped to understand best the smart space.

5.1.1 Qualities of a rich interactive experience

The defining qualities of a rich interactive experience are elusive, but it possible to identified four essential features (extracted from [89]):

1. ***Seamless***. Interactive software produces a *seamless* user experience when it provides immediate responses and smooth transitions between tools, modes, states, displays, and other focal points within the application.
2. ***Focused***. A *focused* experience has a purpose that is clearly defined at the outset and continuously reinforced.
3. ***Connected***. *Connected* applications will transparently locate and exchange information with a variety of data sources and communication devices to shield users from the complexity of having to manage the connections themselves.
4. ***Aware***. Applications seem to be aware of what the user is trying to do when they recognize the current operating context (location, goals, tasks, applications) and use this information to transparently facilitate user and task needs.

5.2 Examples of flexible smart environments

One of the key features of smart spaces is their ability to adapt themselves to the preferences, desires and needs of their inhabitants. To achieve this ability, smart spaces must be able to acquire knowledge about the inhabitants and their surroundings. The information upon which this knowledge is built is called context [88] .

Existing systems cover different domains such as tourist guides, indoor information system and smart environments. Focussing in smart environments, there are systems that adapt depending of context information or user needs [95] :

- i-Room focuses on human computer interaction (HCI) in a single interactive meeting room [90]
- Gaia defines Active Spaces as physical spaces coordinated by a responsive context-based infrastructure [91] . This infrastructure is made available to service applications by means of an operating system (Gaia OS), which provides context and event management services to running programs. As a future work, the Gaia team plan to federate Gaia Services to aggregate different active spaces.
- Cooltown uses the technologies behind the Web to provide pervasive nomadic computing in urban environments [92] . In Cooltown, interest places and resources are tagged with URLs or other identifiers that can be retrieved by users' personal devices by means of bar codes, RFIDs or IR transceivers. URLs may be used to access the different services related to their associated points of interest, and other identifiers (such as ISBN codes) may be resolved to URLs which link to the services related to the identified item. Resources are grouped in places, and for each defined place there is a place manager, which maintains directories of resources, acting both as a resolver for looking up resources in that place from their identifier and as a Web server providing information about resources.
- The Galaxy Service Model is intended to provide a hierarchical service structure for a Smart Space Laboratory [93] . Galaxy uses a set of smart devices, which are called u-Textures and Smart Furnitures. These devices may be aggregated to form a Smart Space Laboratory. The Galaxy Service Model allows exporting services provided by the different smart devices to create applications, which can be composed into other applications, resulting in a multi-layered service composition. Service discovery is performed hierarchically.
- COBRA takes advantage of multi-agent systems to develop context-aware applications [94] . It is based on a broker-centric architecture used to provide runtime support for context awareness in an Intelligent Meeting Room. In COBRA, the environment is divided in domains, and there is a broker for each domain, which is an autonomous agent that manages and controls the context model of the specific domain. Though COBRA brokers are intended mainly for context sharing, its centralized approach to management in each domain and the possibility of sharing context between domains through context federation is closely related to the approach we have taken in this work to develop our hierarchical architecture for smart spaces.

5.3 Platform architecture for multimedia documents adaptation

In case of multimedia document creation, we can distinguish three main types of multimedia adaptation architecture. Hereafter, a brief description of every principle of architecture used in multimedia adaptation.

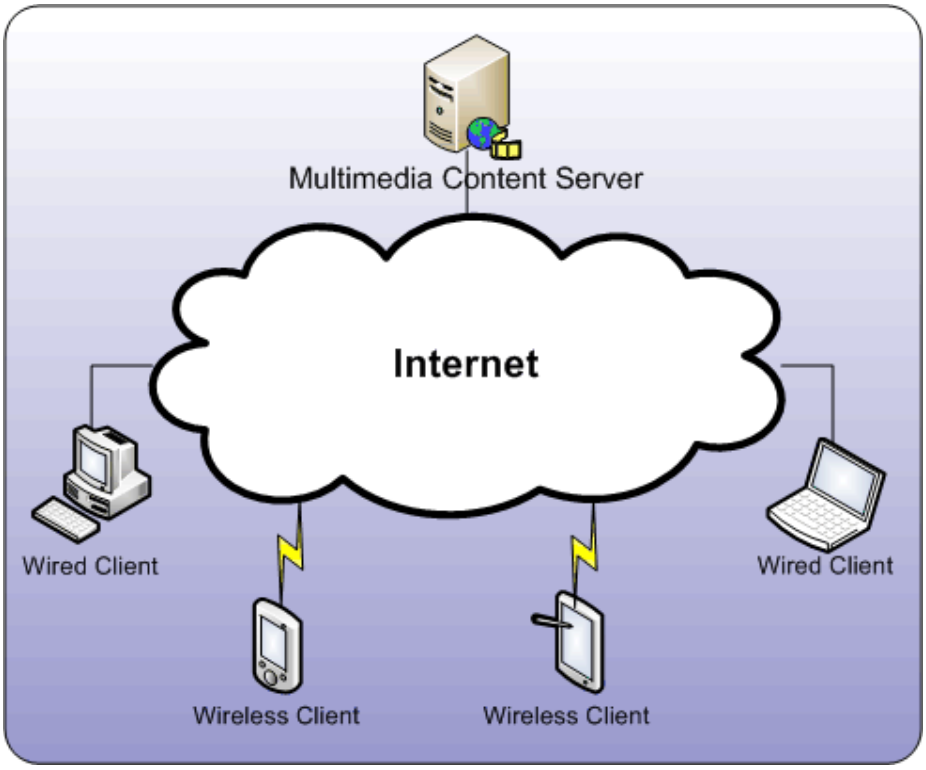

Figure 80. Client / Server model

5.3.1 Server architecture oriented

This architecture is a part of the 2 third party product architectural model Client/Server represented at Figure 80.

This architecture gives the responsibility to the server to:

- Discover the capacities of the customer and the available network constraints
- Decide the best strategy of adaptation

It can consist in 2 types of adaptation:

- (Offline) static Adaptation: the server stores several versions of the same multimedia document for the users needs.
- Dynamic Adaptation (One the Fly): the server takes charge dynamically of the adaptation at the user request.

5.3.2 Client architecture oriented

This architecture is a part of the 2 third party product architectural model Client/Server represented at Figure 80. The client terminal is responsible for the content adaptation depending of software / hardware capacity.

- We can quote the example based on the technique of adaptation built around the target platform: the use of style sheets such as XSLT to transform documents.

5.3.3 Proxy architecture oriented

This architecture is a part of the Client / Proxy / Server model represented at Figure 81. In this approach, a proxy is installed between the client and the server; this proxy dedicated for example to estimate the network bandwidth or to have an access to the user profile.

This architecture brings an added value by avoiding the charge of consumer specific tasks of resources for the client and the server.

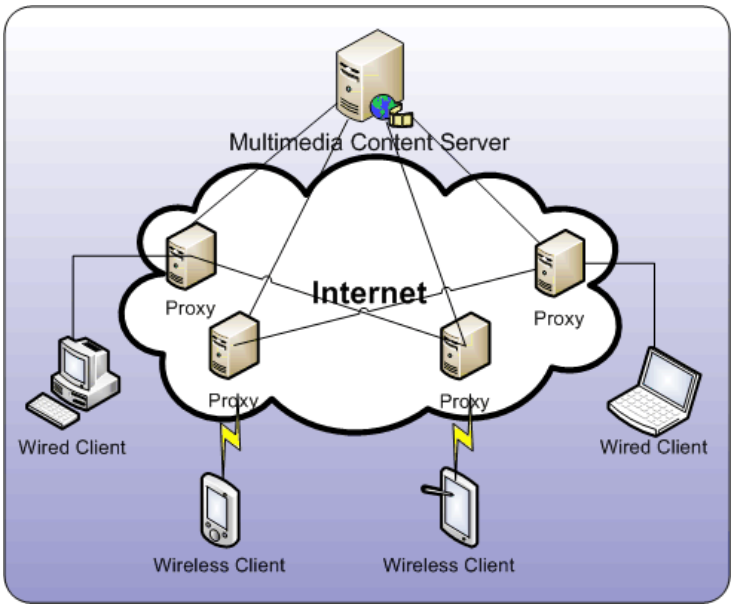

Figure 81. Client / Proxy /Server model

6 Cooperative communities for software and services creation

6.1 Examples of (physical) environments that stimulate user creativity

The aim of this section is to provide several entry points of virtual/physical/mental environments in which DiY related activities can be catalysed.

6.1.1 Tinkering & Pottering

In a DiY related setting, pottering & tinkering are often very important mental processes that drive the design of the things people are creating. In most cases, pottering happens unconscious [190] . A good example of this is when someone is working on an electronics project suddenly discovers something else, his/her attention gets drawn away from the original project and unconsciously starts working in something else. It are often the activities that someone switches to whilst 'pottering' where the real interest or passion of someone can be found. Pottering is closely related to procrastination, although the difference is that pottering is still an acivity that might eventually enhance the original attention focus wheras procrastination operates on delaying tasks in a more conscious way.

6.1.2 Physical environments

- Fablab (http://fab.cba.mit.edu/)

A fablab is a physical space in which people of any background can work, create or learn about creating objects. In a fablab people get access to machinery like 3D printers, laser cutters and vynil cutters to create their own projects. Typically fablabs also organise workshops around a certain topic to stimulate people's creativity.

- Hackerspace (http://hackerspaces.org)

Much like a fablab, a hackerspace is phsical location where people gather to work and learn about creative projects. The main difference from a fablab is that a hackerspace specifically works around technology related projects. This means that actual materialisation of objects is not as central as it is in a fablab.

6.1.3 Virtual environments

- Virtual worlds

A well know example of a virtual world which is second life. Second life kicked off in 2003 and provides a virtual 3D platform in which people can come together to chat and socialize but also to create virtually together. Over the years people have manipulated the platform in such a way to construct games, host virtual meetings, meet new people, etc.

- Makers online community

Within the DiY world, many people are communication through blogs and forums. The most well known are the MAKE: blog and the Instructables forum/website. Based

on these virtual ‘hangouts’ a new subcommunity emerged called “makers”. Makers are people that post instructions & photos of everything they have created themselves. Based on these postings, online conversations & discussions emerge which lead to sharing of methods and new project ideas.

7 Discussion

User interfaces have evolved from text and cryptic commands to Graphic User Interfaces really fast. Of course, nowadays nobody may imagine that a new service is going to be based on a text-interface. In the same way, future services will not be based only on Graphic Interfaces as the ones we are using now, but in more powerful and user adapted ones. These kinds of new interfaces are emerging at this moment and start to be part of our life in mobile phones, video games and movies.

As one of the main ideas for proposed services is that they should be user adapted, it is needed to know who is accessing the service. In this way, user identification and user location is a precondition for our work. One of the approaches to identify users may be the use of the interface itself, to apply any combination of biometric mechanisms to know who is there: face recognition, fingerprints, iris or body geometry may be used for this purpose. In fact, if we use some interfaces for gesture interaction, this identification may be done by guessing the user by the use of the device or the user dimensions; if we use a voice commanded interface, the voice itself may be used to identify the user; etc.

Previous paragraph has shown some interfaces that are not used as shown very frequently at this moment, but the technology exists. And there are some massive video games that use some of them, like the *wii mote*.

On the other hand, we use to interact with servers or devices by using a single interface (in fact, at least two interfaces: one for input and one for output). While this is not completely true (we use to have at least a couple of input devices: keyboard and mouse or similar), it is mainly true for output devices. But when a new service is designed for multiple users, multiple output interfaces will be needed, and not all of them will be identical. For instance, a service may be provided by a personal computer for some of the users, a TV set for some others and mobile phones for a third group. And even a single user may have several output interfaces, like a TV set to follow Formula 1 races and a personal computer to obtain live timing data from cars.

As a rising technology, augmented reality is being adopted by many services that allow users to watch the real world with some added elements which provide an added value. Augmented reality, as a concept, is present from the early 90's, but the technology has been unable to offer a good solution for it until recent dates.

Considering the devices that may be used to obtain services, mobile devices (like iphone, net computers, etc.) have allowed the user to be connected to Internet everywhere. If users are connected, they may access any service provided by the network.

From the service provider point of view, some platforms, like arduino, allow users to create electronic devices with very little electronic knowledge, than may sense the environment and capture useful data. In the same way, anyone may build a robot by using the Lego Mindstorms platform. All of them may be connected to Internet, so they may be part of any user created service. The other element we need to create a service is a software platform that allows users to create these services in an easy way. Some of the existing platforms need some programming knowledge, but some

others provide a graphic programming language to create new services by connecting legacy services, and some data providers. A good example of this technology is Yahoo pipes.

Communities for software creation exist for long time. The Linux world is, probably, the best-known paradigm. If you need a new driver for your graphic card you may create it by your self or you may cooperate with other programmers to do it and allow other people to use it. This paradigm may be adapted to the do-it-yourself developing environment, where service prosumers may create new services by adding new value with very little effort. Just a new small piece of hardware or software, mixed with other services or parts of them, will provide a new service to the community.

At this moment it is not possible to know how the user interaction and application creation is going to be into the do-it-yourself environment. The panorama of devices, interfaces and service platforms is huge enough to difficult this process. But it is a true benefit rather than a problem: the more possible platforms, devices and kinds of interface, the more probability that any user may find his or her best way to create the service they need, and the more probability that any other user will adapt that proposed service to a new device, interface or need: The do-it-yourself ecosystem has started.

8 References

[1] van der Geest, T.M., van Dijk, J.A.G.M., Pieterson, W.J. : "Alter Ego: State of the art on user profiling. An overview of the most relevant organisational and behavioural aspects regarding User Profiling", Telematica Instituut, Enschede, 2005.

[2] Internet Mail Consortium Personal Data Interchange, available at http://www.imc.org/pdi/

[3] IMS Learner Information Package specification, available at http://www.imsglobal.org/profiles/index.cfm

[4] IEEE Public and Private Information Draft 8 specification, available at http://www.edutool.com/papi

[5] Getting Education Systems Talking Across Leading-edge Technologies (GESTALT) project Web site: http://www.fdgroup.com/gestalt/.

[6] Rich, E.: "Users are individuals: individualizing user models", International Journal of Man-machine Studies 18(3), 199—214, 1983.

[7] Kobsa, A.: User Modelling: Recent work, prospects and hazards, Adaptive User Interfaces: Principles and Practices (Schneider-Hufschmidt, T. Khme, U. Malinowski, eds. 1993.

[8] Gauch, S., Speretta, M., Chandamouli, A., Micarelli, A.: "User Profiles for Personalized Information Access", The Adaptive Web, Springer LCNS 4321, pp. 54-89, 2007.

[9] [Karat, C. M., Brodie, C., Karat, J., Vergo, J., Alpert, S. R.: "Personalizing the user experience", IBM Systems Journal, 42(4), 2003.

[10] Goecks, J. and Shavlik, J.: "Learning users' interests by unobtrusively observing their normal behaviour", in Proceedings of the 5th international Conference on Intelligent User interfaces, New Orleans, January 09 – 12 2000.

[11] [Middleton, S. E., De Roure, D. C., & Shadbolt, N. R.: "Capturing knowledge of user preferences: ontologies in recommender systems", in Proceedings of the 1st international Conference on Knowledge Capture, October 22 - 23, 2001.

[12] Godoy, D., & Amandi, A.: "User Profiling for Web Page Filtering", IEEE Internet Computing 9 (4), pp- 56-64, 2005.

[13] Kim, H., Chan, P. K.: "Learning implicit user interest hierarchy for context in personalization", Applied Intelligence, 28-2, pp. 153-166, 2008.

[14] Nanas, N., Uren, V., de Roeck, A.: "Exploiting Term Dependencies for Multi-Topic Information Filtering with a Single User Profile", lecture Notes in Computer Science, 3025, pp. 400- 409, 2004.

[15] OpenKapow, Consulted on January, 11th 2010. [http://openkapow.com/]

[16] D1.1- SoA of applications, type of users, business models, and UCD . DiYSE Project

[17] Wikipedia: RST, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/Representational_State_Transfer]

[18] S. Mitra and T. Acharya. "Gesture Recongition: A Survey" IEEE TransactionsOn Systems, Man, and Cybernetics-Part C: Applications and reviews, Vol. 37, No.3, May 2007.

[19] Jaimes, A. and Sebe, N. 2007. Multimodal human-computer interaction: A survey. Comput. Vis. Image Underst. 108, 1-2 (Oct. 2007), 116-134.

[20] T.B. Moeslund, L. Norgaard, A brief overview of hand gestures used in wearable human computer interfaces, Tech. rep., Aalborg University, Denmark, 2002.

[21] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland, "Pfinder: real-time tracking of the human body," Automatic Face and Gesture Recognition, IEEE International Conference on, pp. 51, Second IEEE International Conference on Automatic Face and Gesture Recognition (FG '96), 1996.

[22] S.A. Niyogi and E.H. Adelson, "Analysing and recognizing walking figures in XYT" IEEE conference on Computer Vision and Pattern Recognition 1994

[23] Han, J., Awad, G., Sutherland, A., and Wu, H. 2006. Automatic Skin Segmentation for Gesture Recognition Combining Region and Support Vector Machine Active Learning. In Proceedings of the 7th international Conference on Automatic Face and Gesture Recognition (April 10 - 12, 2006). FGR. IEEE Computer Society, Washington, DC, 237-242.

[24] Mohan, A., Papageorgiou, C., and Poggio, T. 2001. Example-Based Object Detection in Images by Components. IEEE Trans. Pattern Anal. Mach. Intell. 23, 4 (Apr. 2001), 349-361.

[25] Qing Chen Georganas, N.D. Petriu, E.M. , "Hand Gesture Recognition Using Haar-Like Features and a Stochastic Context-Free Grammar", IEEE Transactions on Instrumentation and Measurement, 2008

[26] T. Zhao R. Nevatia, "Stochastic Human Segmentation from a Static Camera". EEE Workshop on Motion and Video Computing, p. 9, 2002

[27] Mathias Kölsch, Matthew Turk, "Fast 2D Hand Tracking with Flocks of Features and Multi-Cue Integration," cvprw, vol. 10, pp.158, 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 10, 2004

[28] Ahn, Y., Kim, M., Park, Y., Choi, K., Park, W., Seo, H., and Jung, K. 2009. Implementation of 3D gesture recognition system based on neural network. In Proceedings of the 9th WSEAS international Conference on Applied informatics and Communications (Moscow, Russia, August 20 - 22, 2009). N. E. Mastorakis, M. Demiralp, V. Mladenov, and Z. Bojkovic, Eds. Recent Advances In Computer Engineering. World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, 84-87.

[29] N. Jojic et Al, "Tracking Self-Occluding Articulated Objects in Dense Disparity Maps".IEEE International Conference on Computer Vision, Corfu, Greece, September 1999.

[30] H. Sidenbladh et Al. "Stochastic Tracking of 3D Human Figures Using 2D Image Motion" European Conference on Computer Vision, D. Vernon (Ed.), Springer Verlag, LNCS 1843, Dublin, Ireland, pp. 702-718 June 2000.

[31] Theobalt et Al. "Combining 2D Feature Tracking and Volume Reconstruction for Online Video-Based Human Motion Capture", Proc. IEEE Pacific Graphics 2002), Beijing, China.

[32] Mikic et Al. "Human Body Model Acquisition ant Tracking Using Voxel Data" International Journal of Computer Vision 2003.

[33] C. Theobalt et Al. "Enhancing Silhouette-based Human Motion Capture with 3D Motion Fields"

[34] Ho-Sub Yoon, Jung Soh, Younglae J. Bae, Hyun Seung Yang, Hand gesture recognition using combined features of location, angle and velocity, Pattern Recognition, Volume 34, Issue 7, 2001, Pages 1491-1501, ISSN 0031-3203.

[35] Feng-Sheng Chen, Chih-Ming Fu, Chung-Lin Huang, Hand gesture recognition using a real-time tracking method and hidden Markov models, Image and Vision Computing, Volume 21, Issue 8, 1 August 2003, Pages 745-758, ISSN 0262-8856

[36] M. Yeasin, S. Chaudhuri, Visual understanding of dynamic hand gestures, Pattern Recognition, Volume 33, Issue 11, November 2000, Pages 1805-1817, ISSN 0031-3203, DOI: 10.1016/S0031-3203(99)00175-2.

[37] Hong, P., Huang, T. S., and Turk, M. 2000. Gesture Modeling and Recognition Using Finite State Machines. In Proceedings of the Fourth IEEE international Conference on Automatic Face and Gesture Recognition 2000 (March 26 - 30, 2000). FG. IEEE Computer Society, Washington, DC, 410.

[38] H. Zhou, H. Hu, A survey—human movement tracking and stroke rehabilitation, Tech. rep., CSM-420, Department of Computer Science, University of Essex, UK, 2004.

[39] Cinelerra, Consulted on January, 11th 2010. [http://cinelerra.org/]

[40] Kdenlive, Consulted on January, 11th 2010. [http://kdenlive.org/]

[41] Blender, Consulted on January, 11th 2010. [http://www.blender.org/]

[42] Pitivi, Consulted on January, 11th 2010. [http://www.pitivi.org/]

[43] Avidemux, Consulted on January, 11th 2010. [http://www.avidemux.org/]

[44] Jahshaka, Consulted on January, 11th 2010. [http://jahshaka.org/]

[45] Kino, Consulted on January, 11th 2010. [http://kinodv.org/]

[46] Rifftrax, Consulted on January, 11th 2010. [http://www.rifftrax.com/cuts]

[47] Jaycut, Consulted on January, 11th 2010. [http://jaycut.com/]

[48] Moviemasher, Consulted on January, 11th 2010. [http://www.moviemasher.com/]

[49] Maemo, Consulted on January, 11th 2010. [http://maemo.org/]

[50] Maemo Community, Consulted on January, 11th 2010. [http://maemo.org/community/]

[51] Maemo Garage, Consulted on January, 11th 2010. [https://garage.maemo.org/]

[52] Maemo VMware, Consulted on January, 11th 2010. [http://maemovmware.garage.maemo.org/]

[53] Maemo Dev. Doc., Consulted on January, 11th 2010. [http://maemo.org/development/documentation/Quick%20Start%20Guide/]

[54] Scratchbox, Consulted on January, 11th 2010. [http://www.scratchbox.org/]

[55] Hildon, Consulted on January, 11th 2010. [http://live.gnome.org/Hildon/]

[56] Android SDK, Consulted on January, 11th 2010. [http://developer.android.com/sdk/index.html]

[57] Wikipedia: Android, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/Android_google#Features]

[58] Wikipedia: Android Devices, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/List_of_Android_devices]

[59] Android Dev. Tools, Consulted on January, 11th 2010. [http://developer.android.com/guide/developing/tools/index.html]

[60] Android Dev. Tools (adt), Consulted on January, 11th 2010. [http://developer.android.com/guide/developing/tools/adt.html]

[61] Android Dev. Tools (emulator), Consulted on January, 11th 2010. [http://developer.android.com/guide/developing/tools/emulator.html]

[62] Android Dev. Tools (other IDEs), Consulted on January, 11th 2010. [http://developer.android.com/guide/developing/other-ide.html]

[63] Android Dev. Community, Consulted on January, 11th 2010. [http://developer.android.com/community/index.html]

[64] Android Dev. Publisihing, Consulted on January, 11th 2010. [http://developer.android.com/guide/publishing/app-signing.html]

[65] Wikipedia: AR, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/Augmented_reality]

[66] Vered AR, Consulted on January, 11th 2010. [http://vered.rose.utoronto.ca/people/paul_dir/IEICE94/ieice.html]

[67] SE AR, Consulted on January, 11th 2010. [http://www.se.rit.edu/~jrv/research/ar/introduction.html]

[68] Blasttheory, Consulted on January, 11th 2010. [http://www.blasttheory.co.uk/bt/work_cysmn.html]

[69] Youtube: ARLatGT, Consulted on January, 11th 2010. [http://www.youtube.com/user/AELatGT]

[70] ARquake, Consulted on January, 11th 2010. [http://wearables.unisa.edu.au/arquake/]

[71] Youtube: BMW AR, Consulted on January, 11th 2010. [http://www.youtube.com/watch?v=P9KPJlA5yds]

[72] FIAT AR, Consulted on January, 11th 2010. [http://www.fiat500masterpiece.com/site/video.html]

[73] Layar, Consulted on January, 11th 2010. [http://layar.com/]

[74] Qrcodes AR, Consulted on January, 11th 2010. [http://2d-code.co.uk/qr-code-augmented-reality/

[75] Wikitude, Consulted on January, 11th 2010. [http://www.wikitude.org/]

[76] Alec Jeong, “Hologram Science Projects”, 2006. http://www.holokits.com/a-hologram_science_project.htm]

[77] Wikipedia: Holodeck, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/Holodeck]

[78] Youtube: ATV, Consulted on January, 11th 2010. [http://www.youtube.com/watch?v=AqlwZK9e5UI&feature=player_embedded]

[79] James, R. Vallino, “Interactive Augmented Reality”, 1998. http://www.se.rit.edu/~jrv/publications/VallinoThesis.pdf]

[80] Paul Milgram and Fumio Kishino, “A taxonomy of mixed reality visual displays”, IEICE Transactions on Information Systems, Vol E77-D, No.12 December 1994. http://vered.rose.utoronto.ca/people/paul_dir/IEICE94/ieice.html]

[81] Wikipedia: ARToolkit, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/ARToolKit]

[82] ARToolkit homepage, Consulted on January, 11th 2010. [http://www.hitl.washington.edu/artoolkit/documentation/userarwork.htm]

[83] ATOMIC_Authoring_Tool definition, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/ATOMIC_Authoring_Tool]

[84] ATOMIC_Authoring_Tool homepage, Consulted on January, 11th 2010. http://www.sologicolibre.org/projects/atomic/en/index.php?page=Documentation]

[85] Wikipedia: Augmented Reality definition, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/Augmented_reality]

[86] International Symposium on Mixed and Augmented Reality, Consulted on January, 11th 2010. [http://www.ismar09.org/]

[87] PTAM homepage, Consulted on January, 11th 2010. [http://www.robots.ox.ac.uk/~gk/PTAM/]

[88] Chen, G., Kotz, D.: A survey for context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College (2001). http://www.cs.dartmouth.edu/reports/TR2000-381.pdf

[89] Kevin Mullete. The Essence of Effective Rich Internet Applications . Macromedia Experience Design Team. November 2003.http://www.uiresourcecenter.com/rich-internet-applications/whitepapers/TheEssenceOfEffectiveRichInternetApplications.pdf

[90] Johanson, B., Fox, A., Winograd, T.: The interactive workspaces project: Experiences with ubiquitous computing rooms. IEEE Pervasive Computing (2002) 67–74.

[91] Román, M., Hess, C.K., Cerqueira, R., Ranganathan, A., Campbell, R.H., Nahrstedt, K.: Gaia: A middleware infrastructure to enable active spaces. IEEE Pervasive Computing (2002) 74–83

[92] Kindberg, T., Barton, J.: A web-based nomadic computing system. Computer Networks 35 (2001) 443–456.

[93] Yura, J., Nakazawa, J., Tokuda, H.: Galaxy ds: Directory service for service composition based on smart space structure. In: Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05). (2005).

[94] Chen, H.: An Intelligent Broker Architecture for Pervasive Context-Aware Systems". PhD thesis, University of Maryland, Baltimore County (2004).

[95] Marsá, I. M.: SETH: A Hierarchical, Agent-based Architecture for Smart Spaces, Universidad de Alcala (2006).

[96] Yahoo! Pipes official page, Consulted on January, 11th 2010. [http://pipes.yahoo.com/pipes/]

[97] Wikipedia: Yahoo! Pipes, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/Yahoo!_Pipes]

[98] Arduino Home page, Consulted on January, 11th 2010. [http://www.arduino.cc/en/Guide/Introduction]

[99] Wikipedia: Touch Screen, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/Touch-screen]

[100] Apple Inc. Multi-Touch screen, Consulted on January, 11th 2010. [http://www.apple.com/iphone/iphone-3gs/high-technology.html]

[101] Wikipedia: Multi-Touch, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/Multi-touch]

[102] Wikipedia: TuxDroid, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/Tux_Droid]

[103] Tux Doc., Consulted on January, 11th 2010. [http://www.tuxisalive.com/documentation/tutorial/tux-api-documentation]

[104] Tux Gadget Management, Consulted on January, 11th 2010. [http://www.tuxisalive.com/documentation/how-to/tux-gadget-manager/view]

[105] Tux Main Page, Consulted on January, 11th 2010. [http://wiki.tuxisalive.com/index.php/Main_Page]

[106] Mindstorms Main Page, Consulted on January, 11th 2010. [http://mindstorms.lego.com/en-us/default.aspx]

[107] Mindstorms, Consulted on January, 11th 2010. [http://www.dmoz.org/Kids_and_Teens/Sports_and_Hobbies/Toys/Lego/Mindstorms//]

[108] Wikipedia: Mindstorms, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/Lego_Mindstorms_NXT]

[109] Wikipedia: Mindstorms Programming Languages, Consulted on January, 11th 2010. [http://en.wikipedia.org/wiki/Lego_Mindstorms#Programming_languages_2]

[110] Mindstorms Community, Consulted on January, 11th 2010. [http://mindstorms.lego.com/en-us/community/default.aspx]

[111] T. Strang and C. Linnhoff. "A context modelling in 1st Int'l Workshop on Advanced Context Modelling, Reasoning and Management, 2004, pp. 34-41.

[112] Y. Zhiwen, Z. Xingshe, Z. Daqing, C. Chung-Yau, W. Xiaohang and M. Ji "Supporting Context-Aware Media Recommendations for Smart phones. IEEE Pervasive Computing, vol. 5 pp. 68-75, 2006.

[113] Suparna De and Klaus Moessner. Ontology-based Context Inference and Query for Mobile Devices.

[114] A. Gangemi, “Ontology Design Patterns for Semantic Web Content,” in Proc. of ISWC 2005, Galway, Ireland, November 6-10, 2005, ser. LNCS, vol. 3729. Springer, 2005, pp. 262–276.

[115] Department of Defense of the United States, Biometrics Task Force

[116] The Biometric Consortium website: www.biometics.org

[117] A. K. Jain, A. Ross, S. Prabhakar, An Introduction to Biometric Recognition, IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Imageand Video-Based Biometrics 14 (1) (2004) 4–20.

[118] “Biometrics Foundation Documents”. National Science and Technology Council, Committee on Technology, Committee on Homeland and National Security, Subcomittee on biometrics. USA.

[119] Wikipedia: Retina, Consulted on January, 11th 2010. [http://en.wikipedia.org/w/index.php?title=Retina&oldid=330176559]

[120] Mitra, S. & Acharya, T. (2007). Gesture Recognition: A Survey, IEEE Tran. on Systems, Man, and Cybernetics, Part C, Vol. 37, No. 3, 311-324.

[121] [WANG] J.-G. Wang, E. Sung, and R. Venkateswarlu, “Eye gaze estimation from a single image of one eye,” ICCV, pp. 136-143, 2003.

[122] [SMITH] P. Smith, M. Shah, and N.d.V. Lobo, “Determining driver visual zttention with one camera,” IEEE Trans. on Intelligent Transportation Systems, 4(4), 2003.

[123] HEIS] R. Heishman, Z. Duric, and H. Wechsler, “Using eye region biometrics to reveal affective and cognitive states,” CVPR Workshop on Face Processing in Video, 2004.

[124] [SIB] L.E. Sibert and R.J.K. Jacob, “Evaluation of eye gaze interaction,” Conf. Human-Factors in Computing Syst., pp. 281-288, 2000.

[125] Dodson, S. (2008). Forward: A Tale of Two Cities The Internet of Things: A Critique of Ambient Technology and the All-Seeing Network of RFID. Amsterdam: Institute of Network Cultures.

[126] Franke, N., Keinz, P., & Schreier, M. (2008). Complementing Mass Customization Toolkits with User Communities: How Peer Input Improves Customer Self-Design. Journal of Product Innovation Management, 25(6), 546-559.

[127] Franke, N., & Schreier, M. (2002). Entrepreneurial Opportunities with Toolkits for User Innovation and Design. International Journal on Media Management, 4(4), 225 - 234.

[128] Franke, N., & von Hippel, E. (2003). Satisfying Heterogeneous User Needs via Innovation Toolkits: The Case of Apache Security Software. Research Policy, 32(7), 1199-1215.

[129] Prügl, R., & Schreier, M. (2006). Learning From Leading-Edge Customers at The Sims: Opening Up the Innovation Process Using Toolkits. R&D Management, 36(3), 237-250.

[130] Thomke, S., & von Hippel, E. (2002). Customers as Innovators: A New Way to Create Value. Harvard Business Review, 80(4), 74-81.

[131] Von Hippel, E. (2001). User Toolkits for Innovation. Journal of Product Innovation Management, 18(4), 247-257.

[132] WSMX Main Page, Consulted on January, 11th 2010. [http://www.wsmx.org/]

[133] WSMO org., Consulted on January, 11th 2010. [www.wsmo.org]

[134] WSMO studio, Consulted on January, 11th 2010. [http://www.wsmostudio.org/]

[135] Pieter De Leenheer, Christophe Debruyne: DOGMA-MESS: A Tool for Fact-Oriented Collaborative Ontology Evolution. OTM Workshops 2008: 797-806

[136] Nonaka, I., Takeuchi, H.: The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation. Oxford University Press, Oxford (1995)

[137] De Leenheer, P., de Moor, A., Meersman, R.: Context dependency management in ontology engineering: a formal approach. LNCS Journal on Data Semantics 8, 26–56 (2007)

[138] J. Wang, Z. Ding, C. Jiang: GAOM: Genetic Algorithm based Ontology Matching In Proceedings of APSCC, 2006

[139] Stamou, s., and Ntoulas, A. (2009): Search personalization through query and page topical analysis, journal of User modeling and User-Adapted Interaction, Springer Netherlands, ISSN 0924-1868, Volume 19, Number 1-2/Feb. 2009

[140] Das, S., Chong, E. I., Eadon, G., and Srinivasan, J. (2004): Supporting ontology based semantic matching in RDBMS, in RDBMS. Proc. Of 30th VLDB Conference

[141] Fellbaum, C. (1999): WordNet: an electronic lexical database, Massachusetts Institute of Technology, ISBN 0-262-06197

[142] Spyns, P., Tang, Y. & Meersman, R. (2008), An ontology engineering methodology for DOGMA, Journal of Applied Ontology, Volume 3, Issue 1-2, p.13-39

[143] De Moor, A., De Leenheer, P., and Meersman (2006), DOGMA-MESS: A Meaning Evolution Support System for Interorganizational Ontology Engineering, in proc. Of 14th International Conference on Conceptual Structures (ICCS 2006), Volume 4068, Aalborg, Denmark, p.189-203

[144] Tang, Y., De Baer, P., Zhao, G., and Meersman, R. (2009): On Constructing, Grouping and Using Topical Ontology for Semantic Matching, the 5th international IFIP workshop on Semantic Web and Web Semantics (SWWS'09), proc. Of On the Move to Meaningful Internet Systems: OTM 2009 Workshops, Springer, LNCS 5872, ISBN -978-3-642-05289-7, pp 816-825, Vilamoura, Portugal, Nov. 1 ~ Nov. 6, 2009

[145] De Baer, P., Kerremans, K., and Temmerman, R. (2008): Constructing Ontology-underpinned Terminological Resources, A Categorization Framework API, Proceedings of the 8th International Conference on Terminology and Knowledge Engineering, Copenhagen

[146] De Baer, P., Kerremans, K., and Temmerman, R. (2006): Facilitating Ontology (Re)use by Means of a Categorization Framework. In: Meersman, R., Tari, Z. (eds.) On the Move to Meaningful Internet Systems 2006. Proceedings of the AWeSOMe workshop, pp. 126-135.

[147] Yan Tang and Robert Meersman, Use Semantic Decision Tables to Improve Meaning Evolution Support Systems, special issue of the Inderscience International Journal of Autonomous and Adaptive Communications Systems (IJAACS), in, Frode Eika Sandnes, Yan Zhang et. al., (eds.) ISSN (Online): 1754-8640, ISSN (Print): 1754-8632,

[148] Tang, Y. (2009): On Semantic Decision Tables, PhD dissertation, VUB STARLab

[149] Yan Tang and Robert Meersman, SDRule Markup Language: Towards Modeling and Interchanging Ontological Commitments for Semantic Decision Making, Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches, IGI Publishing, ISBN: 1-60566-402-2, USA, 2009

[150] Yan Tang and Robert Meersman, Towards Building Semantic Decision Tables with Domain Ontologies, in book "Challenges in Information Technology Management", Man-Chung Chan, Ronnie Cheung & James N K Liu (eds), ISBN 978-981-281-906-2, 981-281-906-1, World Scientific, 2008

[151] Kallio, S., Kela, J., Korpipää, P. & Mäntyjärvi, J. (2006). User independent gesture interaction for small handheld devices. *Special Issue on Intelligent Mobile and Embedded Systems of IJPRAI,* 20(4), pp. 505-524.

[152] Kela, J., Korpipää, P., Mäntyjärvi, J., Kallio, S., Savino, G., Jozzo, L. & Di Marca, S. (2006). Accelerometer based gesture control for a design environment. *Personal and Ubiquitous Computing*, Online First Springer, pp. 1-15.

[153] Mäntyjärvi, J., Kela, J., Korpipää, P. & Kallio, S. (2004). Enabling fast and effortless customization in accelerometer based gesture interaction. In *Proc. International Conference on Mobile and Ubiquitous Multimedia (MUM)*, ACM, pp. 25–31.

[154] Feldman, A., Tapia, E.M., Sadi, S., Maes, P. & Schmandt, C. (2005). ReachMedia: on-the-move interaction with everyday objects, In *Proc. IEEE International Symposium on Wearable Computers (ISWC'05)*, pp. 52-59.

[155] Linjama, J. & Kaaresoja, T. (2004). Novel, minimalist haptic gesture interaction for mobile devices, In *Proc. NordiCHI2004*, ACM Press., pp. 457-458.

[156] Rekimoto, J. (2001). GestureWrist and GesturePad: Unobtrusive wearable interaction devices, In *Proc. Fifth International Symposium on Wearable Computers (ISWC),* pp. 21-27.

[157] Ronkainen, S., Häkkilä, J., Kaleva, S., Colley, A. & Linjama, J. (2007). Tap Input as an Embedded Interaction Method for Mobile Devices. In *Proc. Tangible and Embedded Interaction*, In Press.

[158] Levin, G. & Yarin, P. (1999). Bringing sketching tools to keychain computers with an acceleration-based interface, In *Proc. CHI 98*, ACM, New York, pp. 268-269.

[159] Sawada, H., Uta, S. & Hashimoto, S. (1999). Gesture recognition for human-friendly interface in designer - consumer cooperate design system. In *Proc. IEEE International Workshop on Robot and Human Interaction*, Pisa, Italy, pp. 400-405.

[160] Rekimoto, J. (1996). Tilting operations for small screen interfaces. In *Proc. 9th Annual ACM Symposium on User Interface Software and Technology,* pp. 167-168.

[161] Oakley, I., Ängeslevä, J., Hughes, S. & O'Modhrain, S. (2004). Tilt and Feel: Scrolling with Vibrotactile Display. In *Proc. Eurohaptics*, pp. 316-323.

[162] Hinckley, K., Pierce, J., Horvitz, E. & Sinclair, M. (2005). Foreground and background interaction with sensor-enhanced mobile devices. *ACM Transactions on Computer-Human Interaction (TOCHI),* 12(1), pp. 31-52.

[163] Nokia Corporation. 5500 phone. (2006)., Consulted on January, 11th 2010. [http://europe.nokia.com/link?cid=EDITORIAL_8657]

[164] Shari Trewin, Gottfried Zimmermann and Gregg Vanderheiden: Abstract User Interface Representations:How Well do they Support Universal Access.

[165] Charles Goldfarb, "HyTime: A standard for structured hypermedia interchange". IEEE computer magazine, vol. 24, iss. 8 (Aug. 1991), pp. 81–84

[166] MHEG ISO Standard, Consulted on January, 11th 2010. [http://www.mheg.org]

[167] P. Hoschka, S. Bugaj, D. Bulterman, et al. Synchronized Multimedia Integration Language - W3C Working Draft 2-February-98. W3C, URL: http://www.w3.org/TR/1998/WD-smil-0202, Februar 1998

[168] Guido van Rossum and al, "CMIFed: A Presentation Environment for Portable Hypermedia Documents", ACM Multimedia '93, Anaheim, Aug '93, 183 - 188.

[169] Nabil Layaïda, "Madeus: Système d'édition et de présentation de documents structurés multimédia ", Thèse, Université Joseph Fourier, juin 1997.

[170] Susanne Boll and Wolfgang Klas, "ZYX- A Semantic Model For Multimedia Documents and Presentations", Proceedings of the 8th IFIP Conference on Data Semantics (DS-8), Rotorua, New Zeeland, 5-8 January 1999.

[171] Thomas D. C. Little and Arif Ghafoor:, "Synchronization and Storage Models for Multimedia Objects", IEEE Journal on Selected Areas of Communications, vol. 8, no. 3, pp. 413-427, April 1990.

[172] Augusto Celentano and Ombretta Gaggi, "A Synchronization Model for Hypermedia Documents Navigation", ACM Symposium on Applied Computing 2000

[173] Romain Deltour and Cecile Roisin, "The LimSee3 Multimedia Authoring Model", ACM Symposium on Document Engineering, 10-13 October 2006, Amsterdam, The Netherlands, pp. 173-175

[174] Susanne Boll, Wolfgang Klas, Utz Westermann, "A Comparison of Multimedia Document Models Concerning Advanced Requirements", Technical Report, 1999. Available from: http://www.cs.univie.ac.at/publication.php?pid=254

[175] Paternò, Fabio, "Model-based design and evaluation of interactive applications", Springer, 1999, 208 p.

[176] Paternò, F., Santoro, C., Mäntyjärvi, J., Mori, G. and Sansone, S., (2008) "Authoring pervasive multimodal user interfaces", Int. Journal on Web Engineering and Technology, Vol. 4, No. 2, pp. 235-261

[177] Szekely, Pedro, "Retrospective and Challenges for Model-Based Interface Development", 2nd International Workshop on Computer-Aided Design of User Interfaces, Namur: Namur University Press.

[178] Oasis-open: Committees, Consulted on January, 11th 2010. [http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=uiml]

[179] UIML Home page, Consulted on January, 11th 2010. [http://www.uiml.org

[180] Abrams, M., Phanouriou, C., Batongbacal, A. L., Williams, S. M., and Shuster, J. E. 1999. UIML: an appliance-independent XML user interface language. Computer Networks 31, 11-16 (May. 1999), 1695-1708. DOI= http://dx.doi.org/10.1016/S1389-1286(99)00044-4

[181] Plomp, Johan; Mayora-Ibarra, Oscar, "A Generic widget vocabulary for the generation of graphical and speech-driven user interfaces", International Journal of Speech Technology (2002) No: 5, 39 – 47

[182] USI XML Home page, Consulted on January, 11th 2010. [http://www.usixml.org]

[183] Limbourg, Q., Vanderdonckt, J., Michotte, B., Bouillon, L., López-Jaquero, V., "USIXML: A Language Supporting Multi-path Development of User Interfaces", in Engineering Human Computer Interaction and Interactive Systems, LNCS 3425, Springer, 2005, pp. 200-220.

[184] XUL Home page, Consulted on January, 11th 2010. [http://developer.mozilla.org/xul]

[185] Wikibooks "Evolution of Operating Systems Designs", http://en.wikibooks.org/wiki/Evolution_of_Operating_Systems_Designs

[186] Buxton B., (2007), "Sketching User Experiences", Morgan Kaufmann, 448pp

[187] Saffer D., (2008), "Designing Gestural Interfaces", O'Reilly Media, 268pp

[188] Iwata H., "Haptic inerfaces" in The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications, CRC press, p229-246

[189] Ishii I., "Tangible User Interfaces" in The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications, CRC press, p469-489

[190] A.S. Taylor, S.P. Wyche, J.Kaye., (2008), "Pottering by design" in Proceedings of the 5th Nordic conference on Human-computer interaction

[191] T. Van Cutsem, S. Mostinckx, E. Gonzalez Boix, J. Dedecker, and W. De Meuter. "AmbientTalk: Object-oriented event-driven programming in mobile ad hoc networks". In Proceedings of the 26th International Conference of the Chilean Computer Science Society (SCCC 2007), November 2007, Iquique, Chile.

[192] J. Dedecker, T. Van Cutsem, S. Mostinckx, T. D'Hondt, and W. De Meuter. "Ambient-oriented programming in Ambienttalk". In Proceedings of the 20th European Conference on Object-oriented Programming (ECOOP 2006), volume 4067 of Lecture Notes in Computer Science, pages 230–254. Springer, 2006.

[193] AmbientTalk Source Code project, consulted on March 31st 2010. [http://code.google.com/p/ambienttalk/]

[194] T. Stanley, T. Close, M.S. Miller, (2009) "Causeway: A message-oriented distributed debugger", HPL-2009-78. [http://www.hpl.hp.com/techreports/2009/HPL-2009-78.html].