



**An online Facial Expression Dictionary as a first step in  
the creation of a complete Nonverbal Dictionary**



Master's thesis of Edwin J. de Jongh

---

Delft University of Technology  
Faculty of Information Technology and Systems  
June 2002



Graduation Committee:

Dr. drs. L.J.M. Rothkrantz  
Prof. dr. ir. E.J.H. Kerckhoffs  
Ir. A. Wojdel  
Prof. dr. H. Koppelaar (chairman)

de Jongh, Edwin J. ([E.J.deJongh@kbs.twi.tudelft.nl](mailto:E.J.deJongh@kbs.twi.tudelft.nl) or [edwin\\_de\\_jongh@hotmail.com](mailto:edwin_de_jongh@hotmail.com))

Master's thesis, June 2002

*“ FED : An online Facial Expression Dictionary as a first step in the creation of a complete Nonverbal Dictionary ”*

Delft University of Technology, The Netherlands  
Faculty of Information Technology and Systems  
Knowledge Based Systems Group

Keywords: Artificial intelligence, Facial expressions, Automatic facial expression recognition, Nonverbal communication, Nonverbal dictionary

---

Picture on cover: FED logo designed by D. Broekens

© Copyright 2002, Edwin de Jongh

## Preface

This thesis describes the research that I have done for my graduation project at the Knowledge Based Systems (KBS) group of the Delft University of Technology, headed by Prof. dr. H. Koppelaar. One of the main research focuses of the KBS group is the field of automatic facial expression recognition. This report describes the development of an application that makes use of techniques from this research field: an online Facial Expression Dictionary.

## Project

The first idea for a suitable graduation project occurred to me in the second year of my study when I saw the movie ‘True romance’. In this movie, a mafia gangster is able to determine whether or not someone is telling the truth by carefully looking at certain facial characteristics. The idea for the graduation project was to carefully measure these facial features with a computer, and build a foul-prove lie detector. Unfortunately, I later found out that this had been tried before and failed, due to the fact that determining the relevant facial features was something that was extremely difficult.

Still, I was grasped by the idea of computers ‘seeing’ human facial expressions. When my supervisor Leon Rothkrantz told me about an idea he had for the development of an online Facial Expression Dictionary, or FED for short, which would involve automatic facial expression recognition, I was immediately enthusiastic and quickly decided that this was what I wanted to do.

As a first step in the development, a tool was developed which would enable users to let FED determine the label of the facial expression shown in a picture. This labeling process depended on the entries present in FED. Logically, the next step in the development was the creation of a FED management tool, which was used to create the FED corpus. As the project progressed, additional ways of issuing a query into FED were identified and implemented. The results of all these activities are summarized in this report.

## Report overview

After the introduction, chapters 2 and 3 describe the concept of an online Facial Expression Dictionary, and the relevant theoretical background is introduced. Chapter 4 describes the system design of the FED website. This includes the website architecture and database design. Chapters 5 and 6 describe the implementation of the FED website and the possible queries into FED respectively. In chapter 7, the performance of the FED website is analyzed. Chapter 8 is devoted to conclusions. Appendix C contains a paper that was written for the 14th Belgian-Dutch Conference on Artificial Intelligence (BNAIC'02), and serves as an extended abstract of this report.

## Acknowledgements

First of all, I want to thank Leon Rothkrantz for his guidance and advice during the project. Our meetings provided me with insight into the scope of the project and with ideas on what to do next. Furthermore, I want to thank Danou Broekens and Leo Buntjer for their help with the design of the FED website, and Boi Sletterink for answering all my questions on Java and UNIX. Finally, I would like to thank all my family, friends and colleagues of the KBS group for their support.



## Summary

A verbal dictionary can be used to look up the spelling of a word, sometimes the phoneme representation, the meaning in different contexts and rules of transformation. The goal of this graduation project was to develop a prototype of an online Facial Expression Dictionary, or FED for short, as a first step in the creation of a complete Nonverbal Dictionary. A complete Nonverbal Dictionary would contain information about all the ways people communicate with each other nonverbally. Instead of words, FED contains information about facial expressions.

FED had to become available as a website. The first step in the creation of FED was the definition of an entry in FED. The choice was made to base each entry in FED on a facial expression picture generated by a facial expression generation tool called FaceShop. The next logical step was to implement management facilities, with which the FED entries could be managed. Subsequently, the FED database was filled with 56 facial expressions.

Essential for a Nonverbal Dictionary is the possibility to issue a *nonverbal query* through (multimodal) content. With FED, issuing a nonverbal query is done through uploading a picture containing a facial expression, after which the user semi-automatically determines the location of the face and the coordinates of the 30 Facial Characteristics Points or FCP's of the face model defined by Kobayashi and Hara. FED then determines the label of the unknown facial expression by comparing the FCP coordinates to the FCP coordinates of all entries present in the database. Also, it is possible to let FED determine the label of a facial expression sketched with FaceShop. Other query possibilities have been implemented as well. It is also possible to look for entries in FED on facial expression label, active Action Units or specific geometrical features. Finally, it is possible to look for entry incrementally, were the user iteratively selects the facial expression that resembles the facial expression he is looking for the closest.

The concept of FED as an online Facial Expression Dictionary was tested and found to be a viable approach. The FED system is easy to use, adapt, extend and manage. The query possibilities have been tested by a group of 30 students, and although there is room for some improvement, the results are satisfactory. The approach taken with FED could be used to create a complete Nonverbal Dictionary.



# Table of Contents

<b>PREFACE.....</b>	<b>I</b>
<b>SUMMARY .....</b>	<b>III</b>
<b>CHAPTER 1: INTRODUCTION.....</b>	<b>1</b>
1.1 PROBLEM SETTING.....	1
1.2 ONLINE NONVERBAL DICTIONARY .....	2
1.3 THE GOAL OF THIS THESIS: FED - AN ONLINE FACIAL EXPRESSION DICTIONARY .....	3
1.3.1 Approach.....	3
1.3.2 Goals .....	4
1.3.3 Challenges.....	5
1.3.4 Applications.....	6
1.4 THESIS OVERVIEW.....	7
<b>CHAPTER 2: THEORETIC BACKGROUND.....</b>	<b>8</b>
2.1 FACIAL EXPRESSIONS.....	8
2.1.1 Universality of facial expressions.....	8
2.1.2 Cultural display rules.....	9
2.1.3 Representation of facial expressions.....	9
2.1.4 Action Units: a universal representation of facial expressions.....	10
2.2 AUTOMATIC FACIAL EXPRESSION RECOGNITION .....	11
2.2.1 Modeling facial expressions.....	11
2.2.2 Face detection.....	11
2.2.3 Facial features extraction.....	11
2.2.4 Classification.....	12
2.3 AUTOMATIC FACIAL EXPRESSION GENERATION .....	12
2.4 THE METHOD OF KOBAYASHI AND HARA.....	13
2.4.1 Face Model.....	13
2.4.2 Determining the FCP's .....	13
2.4.3 Training and Classification.....	13
2.5 INFORMATION RETRIEVAL.....	14
<b>CHAPTER 3: FED CONCEPTUAL DESIGN .....</b>	<b>17</b>
3.1 AN ENTRY IN A FACIAL EXPRESSION DICTIONARY .....	17
3.2 AN ENTRY IN FED .....	17
3.3 POSSIBLE QUERIES IN FED .....	18
3.4 FACE MODEL .....	19
3.4.1 Advantages of the face model of Kobayashi and Hara.....	19
3.4.2 Disadvantages of the face model of Kobayashi and Hara.....	20
3.5 FED ENTRY REPRESENTATION: CLUSTERS VS. TEMPLATES.....	20
3.6 CREATING AN ORDERING BETWEEN ENTRIES IN FED .....	20
<b>CHAPTER 4: FED SYSTEM DESIGN .....</b>	<b>22</b>
4.1 REQUIREMENTS.....	22
4.2 GLOBAL DESIGN .....	22
4.3 THE FED DATABASE.....	24
4.4 VISUAL DATA INSPECTION TOOL .....	27
4.5 ISSUING A QUERY INTO FED .....	28
4.5.1 Label Query.....	29
4.5.2 Action Unit Query.....	29
4.5.3 Geometry Query .....	31
4.5.4 Incremental Query .....	32

## Table of Contents

---

4.5.5	Picture Query.....	33
4.5.6	FaceShop Query.....	34
4.6	FED MANAGEMENT .....	34
<b>CHAPTER 5: FED WEBSITE IMPLEMENTATION.....</b>		<b>37</b>
5.1	DEVELOPMENT TOOLS.....	37
5.1.1	Java.....	37
5.1.2	PostGreSQL .....	38
5.1.3	FaceShop .....	39
5.1.4	Other tools used during development.....	40
5.2	THE FED WEBSITE.....	40
5.3	DATA ACQUISITION.....	41
5.4	USER MANAGEMENT .....	46
5.5	LOGGING.....	46
5.5.1	Query log.....	46
5.5.2	Admin Log .....	48
<b>CHAPTER 6: FED QUERY IMPLEMENTATION.....</b>		<b>50</b>
6.1	VISUAL DATA INSPECTION: DETERMINING THE MOST DISCRIMINATIVE FACIAL FEATURES.....	50
6.2	LABEL QUERY.....	53
6.3	ACTION UNIT QUERY .....	53
6.4	GEOMETRY QUERY.....	55
6.4.1	Geometrical feature detection.....	56
6.4.2	Geometry QPM Implementation.....	56
6.5	INCREMENTAL QUERY.....	60
6.5.1	Determining the likeness of two facial expressions.....	62
6.5.2	Determining the minimum and maximum facial expression of a set.....	64
6.5.3	Incremental QPM implementation.....	66
6.6	PICTURE QUERY .....	68
6.6.1	Face Detection.....	69
6.6.2	Feature Extraction.....	69
6.6.3	Classification – determining the label of the unknown facial expression.....	76
6.7	FACE SHOP QUERY .....	80
<b>CHAPTER 7: RESULTS .....</b>		<b>82</b>
7.1	WEBSITE PERFORMANCE .....	82
7.1.1	Usability.....	82
7.1.2	Speed.....	82
7.1.3	Scalability.....	83
7.1.4	Extendibility / Adaptibility.....	83
7.1.5	Security.....	83
7.2	FED CORPUS.....	84
7.3	SEARCH MODALITIES.....	86
<b>CHAPTER 8: CONCLUSIONS AND FUTURE WORK.....</b>		<b>88</b>
8.1	WEBSITE ARCHITECTURE.....	88
8.2	FED CORPUS .....	88
8.3	SEARCH MODALITIES.....	89
8.4	FED : A VIABLE APPROACH?.....	89
8.5	FUTURE WORK .....	89
<b>BIBLIOGRAPHY .....</b>		<b>92</b>
<b>APPENDIX A: OVERVIEW OF JAVA CLASSES .....</b>		<b>94</b>
<b>APPENDIX B: FED CORPUS.....</b>		<b>100</b>
<b>APPENDIX C: PAPER .....</b>		<b>107</b>







## Chapter 1: Introduction

*The subject of this thesis is the development of an online Nonverbal Dictionary. Online verbal dictionaries are common, but at the moment, no online Nonverbal Dictionary has been developed. Section 1.1 describes the problem setting in which the development of the online Nonverbal Dictionary is set. Section 1.2 gives a motivation for the development of an online Nonverbal Dictionary and describes the main problems that are encountered.. As a first step, a dictionary of facial expressions was developed. The concept and features of an online Facial Expression Dictionary are described in section 1.3. In section 1.4 an overview of the structure of this report is given.*

### 1.1 Problem setting

Nowadays people can communicate with each other in all sorts of ways. Telephone, email, fax, and letters all provide efficient methods for remote communication. When people communicate with each other face to face, i.e. if they can see and/or hear the other person, the interpretation of what is being said does not depend on the meaning of the spoken words alone. Additional meaning and nuances can then be given to spoken words through *nonverbal communication*.

According to Harper (1978), nonverbal communication can be defined as *everything besides the words that comes across from one person to another in a social exchange*. There are two channels of nonverbal communication. The paralinguistic channel includes all other auditory characteristics besides the words and sentences itself: speech rate, loudness, pitch, tone of voice and the placing of inflections. The visible channel includes all aspects of communication that we can see: posture, gestures, eye movements and contact and facial expressions.

The importance of nonverbal communication can be illustrated with the following example. During the final debate between presidential candidates Michael Dukakis and George Bush in 1988, Dukakis was asked if he would still oppose the death penalty if his own wife was raped and murdered. Dukakis answered that he would, because there was no prove that capital punishment was an effective deterrent to capital crime, and furthermore, it would not bring his wife back. Despite the fact that this was a very fair, reasonable and perhaps wise answer, he displayed no emotions about the thought of his wife being raped and murdered through facial expressions or body gestures. People interpreted this as cold and it turned out to be crucial for his chances of winning the election. As an example of the opposite, Ronald Reagan, a former Hollywood actor, was known for his ability to give the impression of a compassionate, feeling and understanding man and was nicknamed 'the Great Communicator'. This despite the fact that the content of his speeches was not of a particularly high standard.

Several problems can arise when people communicate with each other through nonverbal communication. One of the reasons for this is that expression through nonverbal communication is not always universal. Research has shown that, for the most part, the ability to communicate nonverbally is something that has to be learned. It seems for example that people are born with the ability to generate and interpret only six facial expressions (*happiness, anger, disgust, fear, surprise and sadness*, Ekman and Friesen 1972). All other facial expressions and their meaning in different contexts have to be learned from the environment in which a person grows up. The meaning of facial expressions is thus largely context and culture dependant. Furthermore, facial expressions can be ambiguous, i.e. have several possible interpretations. Analogous problems occur with all other forms of nonverbal communication.

## 1.2 Online Nonverbal Dictionary

As with nonverbal communication, problems can arise when people communicate verbally. When neither person understands the language the other person is speaking, communication becomes very difficult. A tool that aims to provide a partial solution for this problem is the *dictionary*. A verbal dictionary can essentially exist in four different formats. It can exist as:

- a reference book containing words usually alphabetically arranged along with information about their forms, pronunciations, functions, etymologies, meanings, and syntactical and idiomatic uses.
- a reference book listing alphabetically terms or names important to a particular subject or activity along with discussion of their meanings and applications.
- a reference book giving for words of one language equivalents in another.
- a list (of items consisting of data or words) stored in a computer for reference (as for information retrieval or word processing).

The fourth type of dictionary can be made up of entries of any of the first three types and can be made available online as a website. A verbal dictionary can thus be of use when you want to communicate with people that speak a different language, or when you want to know the meaning of a certain word.

Analogous to a verbal dictionary, it is possible to define the concept of a *Nonverbal Dictionary*. The type of information that would need to be stored in a Nonverbal Dictionary differs from the type of information stored in a verbal dictionary. Instead of words, a Nonverbal Dictionary would contain information about all the ways people communicate with each other nonverbally: facial expressions, gestures, posture, eye movement and contact, speech rate, loudness, pitch, tone of voice and the placing of inflections. Then people could look up certain expressions of nonverbal communication or ask for an interpretation of a displayed nonverbal expression.

There are several problems that would arise when attempting to develop a Nonverbal Dictionary that would not occur when creating a verbal dictionary:

- Some expressions of nonverbal communication are universal, but most are not; this means that displaying and interpreting certain nonverbal expressions is only for a small part an inborn ability but for the most part something that has to be learned.
- Because expressions of nonverbal communication, for the most part, have to be learned from the environment in which someone grows up, the interpretation of certain nonverbal expressions varies per culture and context.
- Some expressions of nonverbal communication can have several possible interpretations, i.e. are ambiguous.
- Clear, scientifically agreed upon and complete descriptions of expressions of nonverbal communication are often not available. The description of a certain word is generally not a point of heavy debate.
- It is often difficult to determine which types of expression constitute a meaningful expression of nonverbal communication. It is for example not always clear which body postures are communicating something nonverbally and which body postures do not. This problem does not occur with a verbal dictionary, where the collection of words that means something is commonly agreed upon.

Filling a Nonverbal Dictionary with valid entries is thus more complex than creating the corpus of a verbal dictionary. Furthermore, issuing a query into a verbal dictionary is relatively simple to implement; it involves the matching of strings of text. With a Nonverbal Dictionary, it would

have to be possible to issue a *nonverbal query*, i.e. a query consisting of certain nonverbal information. An example of a nonverbal query would be when a user issued a query into a Nonverbal Dictionary using an audio clip exhibiting a certain tone of voice. The Nonverbal Dictionary would then have to be able to return a list of situations in which people are known to use that particular tone of voice. This would involve relatively complex analysis of the input audio signal and a comparison to audio signals stored in the dictionary. In general, processing nonverbal queries will involve more complex calculations than processing a query into a verbal dictionary.

Like a verbal dictionary, a Nonverbal Dictionary could be made available online as a website. Such an *online Nonverbal Dictionary* could help people communicate nonverbally and provide easy access to scientifically valid information on different types of nonverbal communication, which could be of interest to several different groups of people.

### 1.3 The goal of this thesis: FED - an online Facial Expression Dictionary

The previous sections described the importance of nonverbal communication and how an online Nonverbal Dictionary could help people communicate with each other. The goal of my graduation project was to design and implement a prototype of an *online Facial Expression Dictionary*, or *FED* for short, as a first step in the development of an online Nonverbal Dictionary. The insights gained during the development of FED can then be used to extend FED to a complete Nonverbal Dictionary.

The first subsection describes the approach that was taken in the development of FED. The second subsection describes the goals of the graduation project. In the third subsection, the main problems and challenges involved in creating an online Facial Expression Dictionary are discussed. Finally, in the last subsection, an overview of the possible applications of FED is given.

#### 1.3.1 Approach

Table 1.1 shows the similarities between a verbal dictionary and a Facial Expression Dictionary. As becomes clear from table 1.1, conceptually a Facial Expression Dictionary is quite similar to a verbal dictionary. The actual implementation of FED however will be quite different.

**Table 1.1**  
**Verbal Dictionary vs. Facial Expression Dictionary**

Verbal Dictionary	Facial Expression Dictionary
Words	Pictures
Spelling	Action Units / Facial characteristic points
Meaning in different contexts	Meaning in different contexts
Rules of transformation	Cultural display rules

A corpus-based approach was chosen to create FED. Ideally, FED would contain a 24-hour view of the facial expressions of all the people on the planet. Since this clearly is unattainable, a simpler approach had to be taken. The basis for all entries in FED is a picture of a facial expression. Given a picture, the facial expression displayed is generated by a facial expression generation tool. Then, for each picture, the coordinates of 30 so-called *facial characteristic points* (FCP's) and the *active action units* (AU's) are determined. FCP's and AU's both provide the possibility of relating facial expressions to each other and will be described in more detail in

chapters 2 and 3. *Cultural display rules* define how an expression is modified in a certain situation in a certain culture, and will also be explained in chapter 2. All FED entries will be stored in a database.

Analogous to an online verbal dictionary, users will have the possibility to issue various queries into FED online. The coordinates of the FCP's are used to label unknown facial expressions, which constitutes issuing a nonverbal query into FED. Furthermore, a FED administrator will have the possibility to manage the entries in FED. These management facilities can be used to create the FED corpus. Finally, FED will be set up in such a way that enables easy adaptations and extensions of FED in the future.

### 1.3.2 Goals

The goals of my graduation project can be divided into two categories: implementation goals and research goals. The implementation goals are determined by the requirements of the FED system. The research goals of the project stem from the fact that FED is essentially a first step in the creation of an online Nonverbal Dictionary. The implementation goals can be described as follows:

*Implementation goals:*

- Online means that FED will have to become available as a website.
- FED has to provide several ways of issuing a query. The most essential type of query is a nonverbal query where the FED system determines the label of an unknown facial expression displayed in a picture. Other types of queries that have to be implemented are a query on a facial expression label, geometric features, active Action Units and issuing a query incrementally. The latter means that the user incrementally selects the facial expression that resembles the facial expression he is looking for the closest.
- It has to be easy for an administrator to manage the entries in FED. Management functionality will therefore have to be implemented as well.
- It has to be possible to make adaptations and extensions to FED in the future.
- It has to be technically possible to extend FED to a complete online Nonverbal Dictionary, which besides information on facial expressions contains information about all the other ways people communicate with each other nonverbally.

There are several problems that have to be solved in order to meet these implementation goals:

- How to label facial expressions?
- How can you define relations between facial expressions (analogous to a verbal dictionary where the entries are order alphabetically)?
- How to provide users with the possibility to issue a nonverbal query into FED using a facial expression itself?
- How to classify unknown facial expressions?
- How to structure the database of an online Facial Expression Dictionary?

FED will initially be developed as a prototype. It aims to answer a number of questions regarding the viability of FED as an implementation of an online Facial Expression Dictionary and its suitability for extension to a complete Nonverbal Dictionary. This leads to a number of research goals:

*Research goals:*

- How good is the performance of the queries that can be issued into FED?
- To which extent is FED a viable implementation of an online Facial Expression Dictionary and what improvements and/or extensions can be made to improve the functionality of FED as an online Facial Expression Dictionary?
- To which extent is it possible to create an online Nonverbal Dictionary using the same methods that were used to create FED?
- To which extent can the FED implementation be used as a basis for the creation of an online Nonverbal Dictionary?

### 1.3.3 Challenges

FED contains information about facial expressions. There are several problems that need to be considered when trying to implement a Facial Expression Dictionary:

- As mentioned in the previous section, not all facial expressions are universal. The meaning of a facial expression can differ per culture and context.
- Not every possible facial expression can be labeled / has a meaning.
- People can display a mixture of several facial expressions
- Facial expressions are not always displayed to the same degree.
- Certain facial expressions can be displayed in a number of different ways. For example, the facial expression indicating ‘happiness’ can be displayed in a number of different ways.

As mentioned earlier, an essential feature of a Nonverbal Dictionary is that it has to be possible to issue a nonverbal query. For FED, this means that besides providing users with the possibility to issue a verbal query on a facial expression label, FED also has to provide users with the possibility to issue a query using a facial expression itself. This could be accomplished by either letting users supply a picture containing a facial expression, or by letting users generate a facial expression online. Subsequently, FED would then determine the label of the facial expression.

The labeling of an unknown facial expression involves techniques from the field of automatic facial expression recognition. Up until now, all automatic facial expression recognition systems are only operational as offline applications, and most of them exist only in the laboratory. Some systems have a very experimental character and the techniques used are of no use when developing an application that is to be made available as a website. As an example, consider the Active Appearance Model developed by Edwards (1998), where 122 facial points have to be localized manually.

Some of the current state-of-the-art automatic facial recognition systems can be described as fully automatic. All three steps of the automatic facial expression recognition, face detection, facial features extraction and classification, are performed without any user interaction. Ideally, FED would also be fully automatic. It is however difficult to realize a fully automatic system online because of the real-time restrictions. The challenge in developing an online automatic facial recognition system lies in automating as many aspects as possible, to enhance the user friendliness of the system, without crippling the performance of the system. It is also important to realize that the ideal facial recognition system does not yet exist. Pantic (2001) gives a description of the capabilities of the ideal facial expression recognition system (table 1.2). At the moment, no facial expression analyzer exists that implements all of these characteristics and can correctly handle all types of situations that can arise as mentioned in the beginning of this section.

**Table 1.2**  
**Properties of an ideal facial expression analyzer**

General Characteristics	
1	Automatic facial-image acquisition
2	Any possible subject
3	Deals with variation in lighting
4	Deals with partially occluded faces
5	No special markers/make-up required
6	Deals with rigid head motions
7	Automatic face detection
8	Automatic facial expression data extraction
9	Deals with inaccurate facial expression data
10	Automatic facial expression classification
11	Distinguishes all possible expressions
12	Deals with unilateral facial changes
13	Obeys anatomical rules
Characteristics required by Behavioral science research	
14	# of different AU's (of 44 in total)
15	Quantifies facial-action codes
Characteristics required by Multi-modal/media HCI	
16	Unlimited # of interpretation categories
17	Features adaptive learning facility
18	Assigns quantified interpretation labels
19	Assigns multiple interpretation labels
20	Features real-time processing

### 1.3.4 Applications

FED essentially performs the same function as a verbal dictionary: it enables people to determine the meaning of a certain facial expression in multiple ways. Several specific groups of users can be distinguished:

- *People who migrate to another country* can have problems interpreting the facial expressions of the indigenous people of that country, because of the differences in interpretation of facial expressions. FED could be used as a tool that allows people to learn the meaning of different facial expressions in different cultures.
- *Behavioral science researchers* often have a need for accurate descriptions of facial expressions.
- *Comic and cartoon artists* could benefit from a database containing examples of facial expressions. This can save time and increase the uniformity and quality of the resulting end product.
- *The general public* can make use of FED in the same way as they make use of a verbal dictionary. Examples are a student that needs scientifically valid examples of certain facial expressions for a school report on nonverbal communication, or someone that



wants to know the facial expression grandpa is displaying in the family picture taken in 1954.

## **1.4 Thesis overview**

After this introduction, the second chapter will give an overview of the theoretic background in which the development of FED is set. First, the basic psychological concepts currently accepted on facial expressions are briefly introduced. Also in this chapter are a brief overview of the fields of automatic facial expression recognition and automatic facial expression generation, and a description of the method of Kobayashi and Hara for automatic facial recognition. Kobayashi and Hara developed a face model based on 30 so-called Facial Characteristic Points (FCP's) that is used to model facial expressions in FED. Finally, the subject of information retrieval is briefly reviewed here.

Chapter 3 describes the conceptual design of FED. This includes the definition of an entry in FED and the method used to create an ordering between entries in FED. Chapter 4 is devoted to the system design of the FED website. This includes the requirements, global system design and design layout of the two main functionalities provided by FED: issuing a query and managing the dictionary. Also included here are the database design and the design of a Visual Data inspection Tool, that can be used to determine which facial features discriminate best between FED entries. Chapter 5 contains an overview of the tools used for the development of FED (JAVA, PostgreSQL DBMS and FaceShop) and the implementation details of the FED website. Also present is a description of the data acquisition process. Chapter 6 describes the implementation details of the various algorithms that were used to implement the FED queries.

Chapter 7 describes the results: the website performance, an analysis of the FED corpus and performance of the queries. Finally, chapter 8 is devoted to conclusions. This includes a discussion on to what extent the implementation goals are met and conclusions regarding the research goals based on the performance and characteristics of FED.

## Chapter 2: Theoretic Background

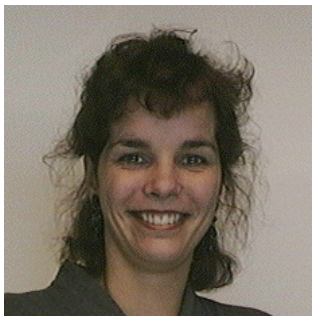
*This chapter briefly introduces the theories and technologies that have played a role in the development of FED. Section 2.1 gives a short introduction on the currently accepted views on facial expressions. Then, in section 2.2 and 2.3 respectively, the principles of automatic facial expression recognition and automatic facial expression generation are discussed, which is followed by a description of the method for automatic facial expression recognition developed by Kobayashi and Hara in section 2.4. Finally, sections 2.5 introduce the concept of information retrieval.*

### 2.1 Facial expressions

Facial expressions play an important role in nonverbal communication. Psychological research has shown that facial expressions are an indication of someone's emotional state.

#### 2.1.1 Universality of facial expressions

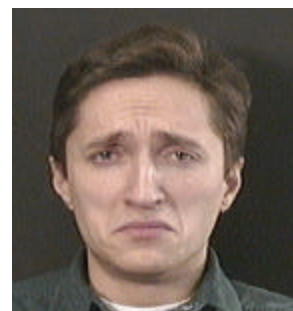
Cross-cultural psychological research on facial expressions indicates that there may be a small set of facial expressions that is *universal*. This was first suggested by Charles Darwin in his work *On the Origin of Species*. Psychologists Paul Ekman and Wallace Friesen (Ekman 1972), and independently, Carroll Izard (1971) conducted the first methodologically sound studies, and concluded that the emotions *Happiness*, *Anger*, *Sadness*, *Disgust*, *Surprise* and *Fear* are shown and interpreted in all human cultures in the same way. It should be noted that not all social psychologists accept these conclusions (e.g., J.A. Russell, 1994). Also, at the present time there are indications that the emotion *contempt* is also universal. Figure 2.1 shows the 7 basic emotions.



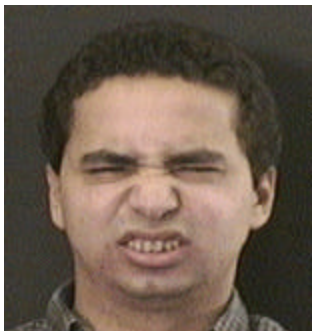
*Happiness*



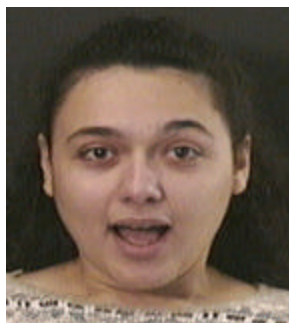
*Anger*



*Sadness*



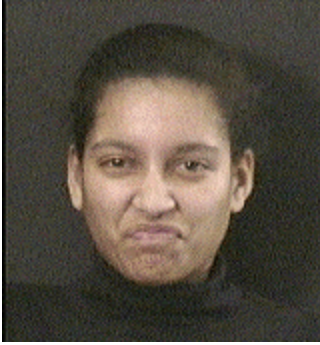
*Disgust*



*Surprise*



*Fear*



*Contempt*

**Figure 2.1: The seven universal expressions of facial emotion**

Ekman and Friesen called their 6 emotions the 6 basic emotions. Even though their research suggests that all humans are born with the ability to express and recognize these 6 basic emotions, it does not imply that emotions are displayed, interpreted and experienced in the same way across all cultures.

### 2.1.2 Cultural display rules

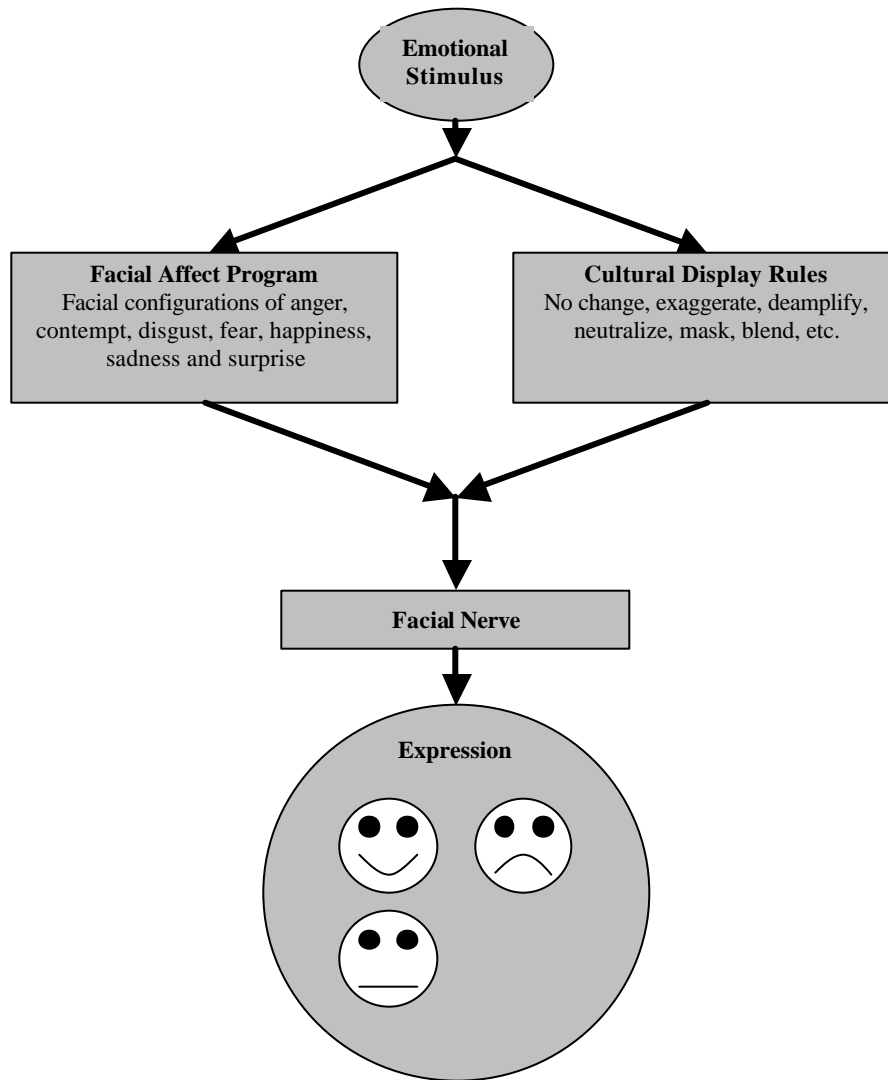
Ekman and Friesen explained this discrepancy by introducing the concept of cultural display rules. Cultural display rules determine how basic universal emotions are modified in a certain culture in certain social circumstances. They performed a study where American and Japanese subjects were shown highly stressful films while their facial expressions were being monitored. When watching the films alone, both American and Japanese subjects expressed negative feelings of disgust, anger, sadness and fear. When the experiment was done while an older, higher-status experimenter was with them in the room, the Japanese carefully hid their emotions, while the Americans continued to show their emotions. The conclusion is that facial expressions of emotion are influenced by universal, biological factors and also by culturally dependent learned display rules. Figure 2.2 gives a schematic overview of the principle of display rules. An emotional stimulus triggers an event where a universal facial expression is modified as a result of the relevant cultural display rule(s) to create the final facial expression displayed by a person.

### 2.1.3 Representation of facial expressions

When developing an online facial expression dictionary, it is important to realize that there are many possibilities that exist to represent a facial expression. Facial expressions can be represented through:

- pictures
- video
- cartoons
- smileys
- facial characteristic points
- active Action Units (see next section, 2.1.4)

As mentioned in the introduction, the basis of a facial expression entry in FED is a picture.



**Figure 2.2: The neurocultural theory of emotional expression – Cultural display rules**

#### 2.1.4 Action Units: a universal representation of facial expressions

The findings on the universality of 6 basic emotions inspired researchers to try and find a way to measure facial expressions, so that emotions could be objectively measured (instead of relying on the subjective interpretation of an observer). Probably the most prominent and most used technique to emerge is the Facial Action Coding System (FACS), developed by Ekman and Friesen in 1978. In their research they define 44 so-called Action Units (AU's). Each action unit describes the movement of certain muscle(s) of the face. Every facial expression can then be described in terms of which AU's are active, i.e. which muscles are flexed and which muscles are relaxed. FED also provides a representation of a facial expression entry in terms of active Action Units.

## 2.2 Automatic Facial Expression Recognition

Although the description of a facial expression in terms of AU's can give an objective description of emotion, an observer has to be highly trained to determine exactly which AU's are active. If this recognition process is automated it can have huge benefits for behavioral science research. Automatic facial expression recognition can also be of importance for applications in the field of human-computer interfaces, monitoring and education. Examples are a computer system interface which gives feedback to the user depending on the emotional state of the user, or a monitoring system in the cockpit of a plane, which alerts the people in the control tower when the pilot becomes stressed.

Because of the many possible applications, research on automatic facial expression recognition has been conducted since the 1970's. The first subsection describes how facial expressions are modeled in current automatic facial expression recognition systems. The next three sections describe the three steps that have to be performed by any automatic facial expression recognition system: face detection, facial features extraction and facial expression classification.

### 2.2.1 Modeling facial expressions

Basically all of the current state of the art automatic facial expression recognition systems use one of three methods to model / represent a facial expression. The facial expression is either represented in the system as a whole (holistic representation), as a set of facial characteristic points or contours describing the eyes, eyebrows and mouth (analytic representation) or as a combination of these (hybrid approach).

An example of the analytic representation is the method of Kobayashi and Hara (1992), where the face is modeled as a set of 30 facial characteristic points. Terzopoulos and Waters (1993) used the holistic approach and modeled the face as a 3D wire frame model with texture mappings. An example of the hybrid approach is provided by Thalmann (1998), who modeled the face as a wire frame model combined with a number of 3D facial characteristic points.

A new method for representing facial expressions, closely related to the representation through AU's, is representation of facial expressions through so-called *Facial Animator Parameters* or *FAP's*. The FAP's are based on a study of minimal perceptible actions. There are 68 FAP's, divided into 10 groups based on different parts of the face. Representation of facial expressions through FAP's is especially useful when trying to animate faces.

### 2.2.2 Face detection

The first step that has to be performed by an automatic facial expression recognition system is given an input image, determine the position of the face. Automatic detection of the position of the face is complex because of the fact that the size and orientation of the head may differ for different input images. The current state of the art systems generally assume input images where the face is visible in frontal view. In general, methods from the field of image processing are applied to detect the face in an input image.

### 2.2.3 Facial features extraction

The next step that has to be performed by an automatic facial expression recognition system is the automatic extraction of facial expression feature information. The method used is dependent on the representation method of the face and the kind of input images (static or dynamic). If the analytical approach is used to model the face, the relative positions and distances between the facial characteristic points are used for facial expression recognition. If the face is modeled as a

whole, any data structure that describes the face as a whole (a complete 2D array of intensity values of the image, a labeled graph) can be used to represent the facial expression information. If the hybrid approach is used, some facial characteristic points usually determine the initial position of a certain template.

### 2.2.4 Classification

The final task to be performed by an automatic facial expression recognition system is the classification of the facial expression displayed in the input image into a certain category. Successful classification is only possible if the input images are normalized in some way, so that images can be compared with each other. Kobayashi and Hara normalize all input images by defining their facial characteristic points in a Facial Coordinate System which is independent of the size of the facial expression in the input image and variations in the in-plane (the tilt of the head to the left or right).

Another important issue involved in the classification is defining a set of categories into which the input images are to be classified. This depends on the application domain of the facial recognition system. If the system is to be used as a tool for behavioral research, it may be desirable to determine and quantify the AU's present in the input image. In other cases it may be useful to classify the input image into an emotion category.

A number of categorization mechanisms can be used. With template-based classification, the unknown facial expression is compared with templates representing the classification categories (for example the 6/7 basic emotions). The image is classified into the category of the template to which it is closest. Neural networks can also be used as a classification method. Neural networks are an example of a black box approach. The neural network is trained by using a set of images that have been correctly classified as a certain emotion by a human expert. After training, the neural network can be used to correctly classify new images of which the corresponding emotions are unknown. Another method often used for classification in facial expression recognition systems is a rule-based expert system. The expert system contains a knowledge base with information about facial expression features stored in the form of logical if-then rules. The facial features of the unknown facial expression are given as input into the knowledge base and the facial expression label is determined through logical inference. There are several other techniques from the field of Artificial Intelligence that can be used to implement the classification component of an automatic facial expression recognition system, such as case based reasoning, fuzzy logic or genetic algorithms.

## 2.3 *Automatic facial expression generation*

Many applications in the fields of teleconferencing, human computer interface and computer animation require realistic reproduction of facial expressions. Automatic generation of facial expressions can be viewed as the inverse of automatic facial expression recognition. With facial expression recognition, the classifier is presented with the facial feature information (for example the coordinates of the 30 FCP's of the face model developed by Kobayashi and Hara), and tries to determine the facial expression label. With facial expression generation, the system has to determine the coordinates of the FCP's given a certain facial expression label. The most popular face model used today by researchers in this field is the face model based on FAP's (see section 2.2.1).

## 2.4 The method of Kobayashi and Hara

This section describes the method for automatic facial expression recognition developed by Kobayashi and Hara (1992). This includes a description of the face model, how the facial features are extracted and how classification of the input image was achieved. Kobayashi and Hara did not develop a method for detection of the face in the picture; they assume correct input images where a face is present in frontal view.

### 2.4.1 Face Model

Kobayashi and Hara model the face through 30 so called Facial Characteristic Points (FCP's). These 30 FCP's correspond to 30 of the 44 Action Units (AU's) of the FACS system, developed by Ekman and Friesen to objectively represent facial expression information. The 30 AU's chosen by Kobayashi and Hara are related to the contours of the eyes, eyebrows and mouth. They didn't use all 44 AU's, because experiments have shown that people only pay attention to the position and size of the eyes, eyebrows and mouth when classifying facial expressions. The other 14 AU's correspond to the movement of the cheek, chin and wrinkles. Fig. 2.3 shows the position of these 30 FCP's. The vertical lines in fig. 2.3 are the so-called *haralines*. The positions of the haralines are fixed and depend on the position of FCP's a1, a2, a3, a4 (corners of the eyes) and FCP a19 and a20 (inner corners of the eyebrows). The x-coordinates of all the other FCP's are fixed depending on the position of the haralines. This is a property of the face model of Kobayashi and Hara that provides the possibility of making the positioning of the FCP's semi-automatic. More details on the face model of Kobayashi and Hara will be described in chapter 4 on implementation.

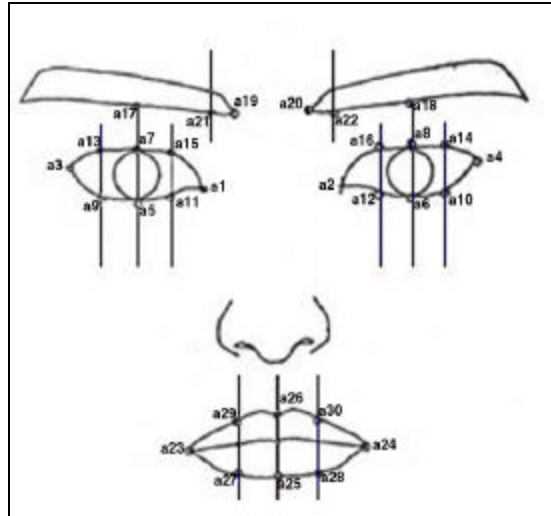
### 2.4.2 Determining the FCP's

The FCP's are determined manually by the user by clicking on the picture. Initially, the coordinates of the FCP's are given by the relative pixel position with respect to the origin of the coordinate system of the picture (usually the upperleft corner). Before the FCP feature vector can be used for facial expression recognition, the FCP's all have to be normalized to account for variations in size, position and in-plane orientation of the face. Kobayashi and Hara achieve this normalization by applying three transformations to the coordinates of the FCP's. These transformations are performed as soon as the two FCP's positioned at the inner corners of the eyes have been positioned by the user. In effect, the coordinates of the FCP's are transformed from one coordinate system, the *picture coordinate system* or *world coordinate system*, to another, the *facial coordinate system* (FCS).

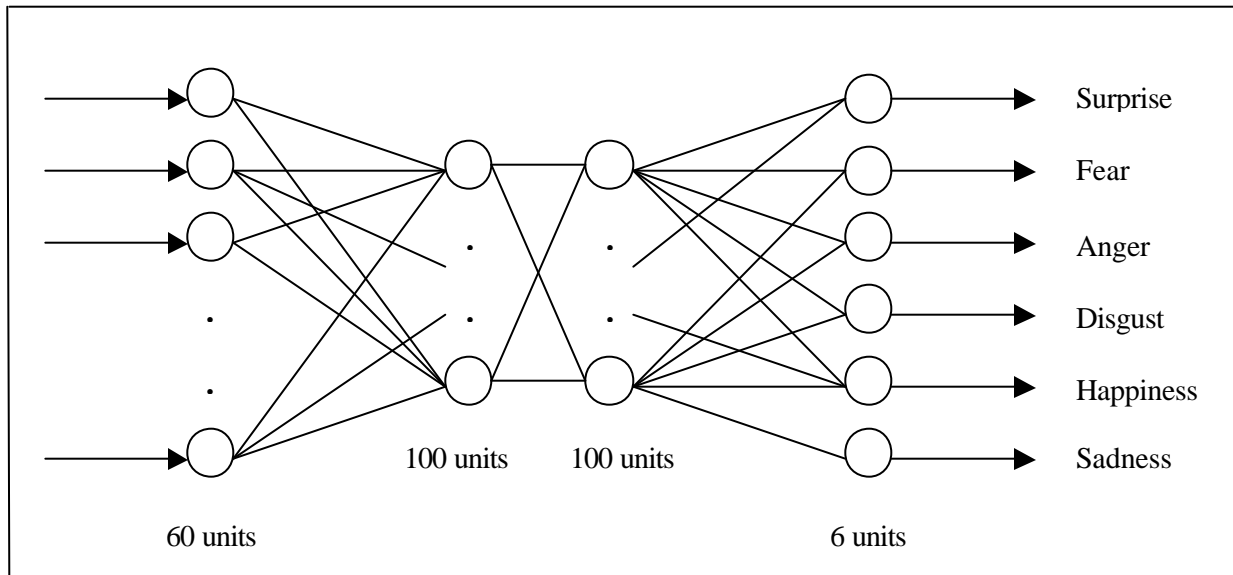
### 2.4.3 Training and Classification

Kobayashi and Hara used a 60-100-100-6 feedforward neural network to classify facial expressions. The 60 inputs correspond to the normalized coordinates of each of the 30 FCP's. The output layer contains one neuron for each of the six basic emotions *Happiness*, *Anger*, *Sadness*, *Disgust*, *Surprise* and *Fear* (see fig. 2.4). The neural network is trained using so called *ideal Ekman sets*. The ideal Ekman set consists of seven pictures of the same person. Six pictures show one of the basic emotions, and one shows the neutral expression. The FCS coordinates of the FCP's of the neutral expression are subtracted from the FCS coordinates of the FCP's of the basic emotions. These normalized feature vectors are given as input to the neural network along with

the correct output during training. After training, the neural network is capable of correctly classifying facial expressions of which the emotion category is unknown.



**Figure 2.3: the FCP's in the face model of Kobayashi and Hara**



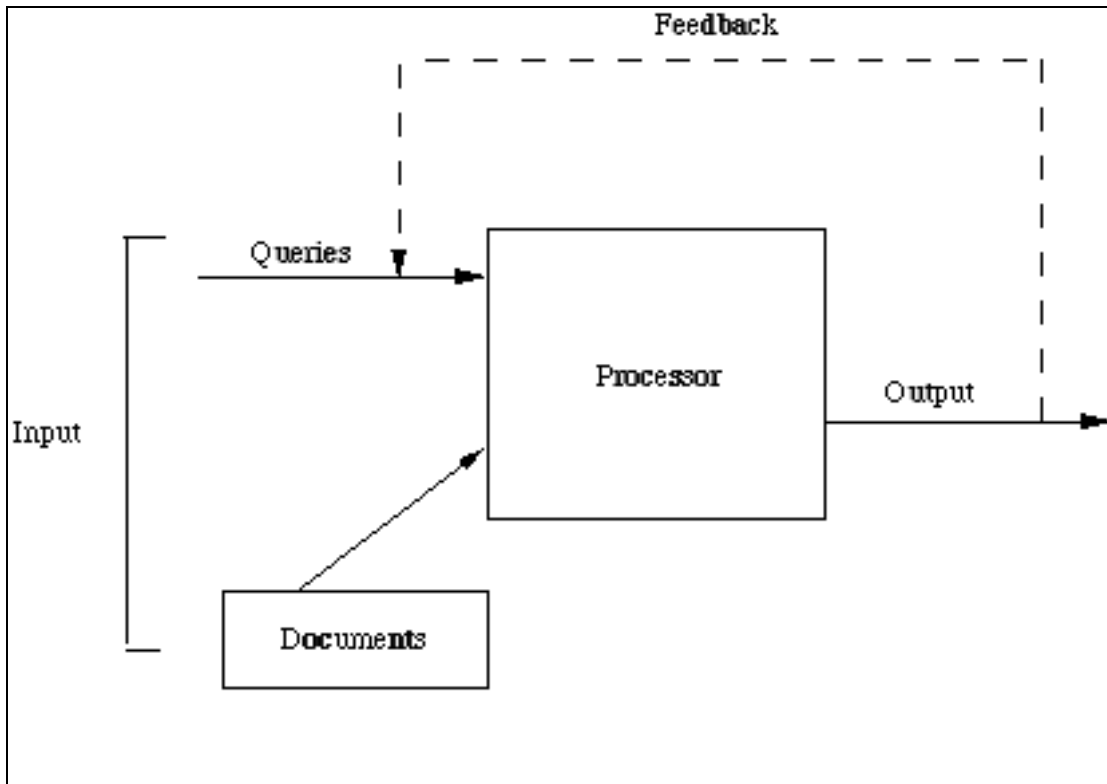
**Figure 2.4: the neural network used for facial expression recognition by Kobayashi and Hara**

## 2.5 Information Retrieval

There are vast amounts of information available in (online) computer systems. The total amount of information stored worldwide increases each year. Retrieving relevant information speedy and



accurately is becoming ever more difficult. Since the development of the computer, information retrieval systems have been created that aim to provide a solution. Although some advances have been made, the problem of retrieving all and only the relevant information is still largely unsolved. Figure 2.5 shows the general design of an information retrieval system.



**Figure 2.5 General design of an Information Retrieval System**

The actual implementation of an information retrieval system depends on the kind of information that is to be retrieved. The first information retrieval systems were used to search collections of text documents. Instead of parsing each document entirely, documents were usually characterized by a limited number of keywords. Advances in the field of natural language processing have made it possible to determine the relevancy of a document by looking at the entire text, but these methods are time-consuming and still far from perfect. Given below is an overview of the search techniques most commonly used by current information retrieval systems.

### *Boolean search*

A Boolean search strategy retrieves all information that evaluates as 'true' for the query. This formulation only makes sense if the queries are expressed in terms of index terms (or keywords) and combined by the usual logical connectives AND, OR, and NOT.

### *Matching functions*

A matching function calculates the degree of association between a query and a document or cluster profile. An example of a matching function is (With  $D$  the set of keywords representing the document, and  $Q$  the set representing the query):

$$M = \frac{2|D \cap Q|}{|D| + |Q|}$$

### *Serial Search*

With this type of search strategy, a matching function is used to calculate the degree of association between the query and a collection of documents. The documents can be ranked by degree of association, or a threshold can be used to filter out unlikely matches.

### *Cluster representatives*

A profile is defined for all identifiable clusters of documents in the total collection. The query is matched against these profiles, or cluster representatives. The cluster of documents corresponding to the best matching profile is returned.

### *Cluster-based retrieval*

All clusters are ordered in a tree structure. The search starts by evaluating the value of a matching function for the top cluster (node 0). The search then proceeds to evaluate the matching function for the immediate descendants of the first node. This pattern repeats itself down the tree. The search is directed by a decision rule, which on the basis of comparing the values of a matching function at each stage decides which node to expand further. For example, the node with the highest matching function value could be used. A stopping rule determines when to terminate the search and retrieve a cluster.

The above search strategy can be described as a top-down search strategy. It is also possible to traverse the tree through a bottom-up strategy, with the terminal nodes of the tree evaluated first.

### *Relevance feedback*

This search technique uses feedback provided by the user to iteratively improve the relevancy of the returned documents. Experiments have shown that relevance feedback can be very effective. A problem with implementing a relevance feedback system is that it is difficult for users to determine the relevance or non-relevance of a document

Except for text, computers are also used to store multimodal information. This can include images, music, video etc. Multimodal Information Retrieval (MMIR) differs from text-based information retrieval in a number of ways. Different search techniques are used to retrieve the relevant multimodal information corresponding to a query. The most commonly used techniques are based on relevance feedback algorithms. Also, often a query can be issued using multimodal content.

## Chapter 3: FED conceptual design

*This chapter describes the concept of FED as an online Facial Expression Dictionary. In section 3.1, a general description of the information associated with an entry in a Facial Expression Dictionary is given. Section 3.2 describes how an entry is defined in FED. This is followed by an overview of the possible queries into FED in section 3.3 and a motivation for the choice for the face model of Kobayashi and Hara in section 3.4. Section 3.5 describes how an entry is represented in FED. Finally, section 3.6 describes how an ordering between entries in FED is created.*

### 3.1 An entry in a Facial Expression Dictionary

An entry in a facial expression dictionary can contain all kinds of information. Graphical information can give an idea of how the facial expression actually looks like. Verbal information can be used to create a description of the facial expression in various ways. Information about the facial features of the facial expressions can be used to create an ordering between the entries in the facial expression dictionary.

#### *Graphical information*

This can include a 2-or 3D picture of the facial expression in frontal view or perspective. The picture can be taken from the real world, or generated by a facial expression generation tool.

#### *Verbal information*

The verbal information can include a facial expression label, synonyms of the label and a description. Because of the fact that facial expressions can have a different meaning in different contexts and cultures, all information about these variations in interpretation can be included in the description as well. Also, graphical items can be used to indicate the context in which the facial expressions can be encountered in the real-world. Smileys and cartoons can be used for this and function as a kind of abbreviation of the facial expression. Finally, the verbal information can include links to other sources of information about the facial expression (books, URL's, research papers).

#### *Facial Feature information*

Facial expressions can be modeled / described by their facial features. One example is the modeling of facial expressions by their active Action Units (AU's), as defined by Ekman and Friesen in their Facial Action Coding System (FACS, section 2.1.4). Many other methods exist. Other possibilities include defining the facial expression features in terms of certain facial characteristic points, templates, shading characteristics or by storing the facial expression information as a whole. The facial features can be used to define relations between the entries in a facial expression dictionary, and enable users to perform information retrieval through multimodal content (graphical representation of a facial expression or a certain facial feature).

### 3.2 An entry in FED

An entry in FED doesn't contain all the information described in the previous section. As mentioned in the introduction, a corpus-based approach was taken for FED, with a picture forming the basis of each entry. Analogous to a verbal dictionary, a facial expression label, label

synonyms and meaning are associated with each entry. Each entry also has a number of active AU's and the coordinates of the 30 FCP's of the face model developed by Kobayashi and Hara associated with it.

Table 3.1 shows the differences and similarities between an entry in a verbal dictionary and an entry in FED. As becomes clear from table 3.1, an entry in FED contains certain information that is also present in a verbal dictionary entry: it will be possible to issue a verbal query into FED, analogous to a verbal dictionary. The presence of additional information in a FED entry means that there are additional possibilities for issuing a query into FED.

**Table 3.1**  
**Verbal dictionary entry vs. FED entry**

Verbal Dictionary Entry	FED Entry
Word	Facial expression label
Synonyms	Label synonyms
Meaning / context description	Meaning / context description
-	Picture
-	FCP's
-	Active AU's

### **3.3 Possible queries in FED**

Issuing a query into FED is partly analogous to issuing a query into a verbal dictionary. It is possible to search for entries on facial expression label, label synonyms or words in the context description of a facial expression. This involves techniques from the field of natural language processing. However, FED also provides several ways of issuing a query that are not possible with a verbal dictionary.

It is also possible to issue a nonverbal query: determining the label of an unknown facial expression. The first step in labeling an unknown facial expression is accomplished by uploading a picture containing a facial expression. When the user has determined the position of the face in the picture, he has to determine the positions of the FCP's of the facial expression shown in the picture semi-automatically. The unknown facial expression is given the same label, synonyms and context description as the facial expression whose FCP coordinates best match the FCP coordinates of the unknown facial expression. If a user does not have a picture of the unknown facial expression, it is possible to generate a facial expression online using the FaceShop drawing tool. In that case, the positions of the FCP's are determined automatically.

Because of the fact that each facial expression entry in FED contains the coordinates of the FCP's, there are additional ways to issue a query that are not present in a (online) verbal dictionary. A user can also issue a query on geometric features, like for example issue a query for all facial expressions with the eyebrows raised or the mouth open. Also, it is possible to issue a query for all facial expressions with certain Action Units active.

With a verbal dictionary, it is possible to browse through the entire dictionary alphabetically. With FED, there is a similar query: finding an entry incrementally. With this query, the user incrementally selects the facial expression that resembles the expression he is looking for the closest from 4 facial expressions representative of 4 clusters covering a certain set of facial expressions. At each iteration, the 4 possibilities displayed to the user resemble the previously selected facial expression, and look more like the expression the user is looking for.

Table 3.2 shows the differences and similarities between queries into an online verbal dictionary and queries into FED.

**Table 3.2**  
**Verbal dictionary queries vs. FED queries**

Verbal Dictionary Queries	FED Queries
Keyword	Facial expression label
Synonyms	Label synonyms
Meaning / context description	Meaning / context description
Browse dictionary alphabetically	Search for entry incrementally
-	Active AU's
-	Geometric features
-	Label sketched facial expression
-	Label facial expression in picture

### 3.4 Face model

FED has to provide users with the possibility of determining the label of an unknown facial expression shown in a picture through a nonverbal query. In order to successfully label an unknown expression, techniques from the field of automatic facial expression recognition are used. The face model used in FED is the face model that was developed by Kobayashi and Hara (see section 2.4).

#### 3.4.1 Advantages of the face model of Kobayashi and Hara

There are several reasons why the method of Kobayashi and Hara is used for the implementation of the nonverbal query into FED where the label an unknown facial expression is determined. First of all, the real-time requirements of the FED website make it difficult to use a holistic approach for the face representation. Using a representation of the face as a whole, such as a complete 3D wireframe model with texture mappings, would result in unacceptable performance. With the analytical approach of Kobayashi and Hara just 30 2D points are used to represent the face.

Secondly, many facial expression techniques have an experimental character. They are developed for research purposes, not for use in practice, and often solve only a small subpart of the facial expression recognition problem correctly. This is undesirable, since FED is to be made available to the general public. This means that, besides good performance in terms of speed, good performance in terms of correct classification of facial expressions is desirable too. Kobayashi and Hara reached a correct classification rate of 90% of the input into one of the six basic emotion categories. Although FED will not use the same classification technique as Kobayashi and Hara used, their results indicate that their face model is a good representation of the facial features that determine the displayed facial expression.

Thirdly, a facial expression generation tool called *FaceShop* was already available. The FaceShop tool is also based on the face model of Kobayashi and Hara. By choosing the same model, FaceShop could be used to create the FED corpus and provide an additional way of issuing a query (labeling of a sketched facial expression).

Finally, describing each facial expression in FED by a set of facial characteristic points enables the creation of an ordering between entries in FED, as will be described in section 3.6.

### 3.4.2 Disadvantages of the face model of Kobayashi and Hara

A major disadvantage when using Kobayashi and Hara's method in FED, is that the 30 FCP's have to be determined manually. This can be a time-consuming, tedious and sometimes inaccurate process. To remedy this, one of the goals of FED was to make the feature extraction semi-automatic. This can be achieved in several ways, one of which is the use of the haralines mentioned in section 2.4.

Another possible disadvantage is that it is uncertain that the classification of unknown facial expressions can be just as successful when using different techniques than neural networks, when you are dealing with a much larger number of classification categories than just the 6 basic emotions of the experiment performed by Kobayashi and Hara.

### 3.5 *FED entry representation: Clusters vs. Templates*

In section 3.2, a FED entry was described as having six properties: a facial expression label, label synonyms, a description, an example picture, active AU's and the coordinates of the FCP's. For all facial expression, the first four properties are always fixed and unambiguous. The active AU's and the coordinates of the 30 FCP's however are not.

The reason for this is that there are usually several ways in which a facial expression can be displayed. For example, the facial expression described as '*happiness*' can in practice occur in a variety of different realizations. Furthermore, people are able to display mixtures of several facial expressions, for example a mixture of '*happiness*' and '*surprise*', and can display a facial expression to a certain degree.

With FED, it is possible to let the system determine the label of an unknown facial expression shown in a picture or the label of a facial expression sketched online. This is accomplished by comparing the FCP coordinates of the unknown facial expression to the FCP coordinates of the FED entries. The closest match determines the label of the unknown facial expression. The performance of these queries depends on the way a facial expression is represented in FED.

One way of representing a facial expression entry is through a cluster of different realizations of that facial expression. Each realization has its own FCP coordinates and active AU's. Another possibility is to represent a facial expression entry by just one template expression. The latter approach can be described as a first-order solution, whereas the first approach can be seen as an extension of this method or a second-order solution.

As a first approach, the choice was made to represent each facial expression by just one template. Given the fact that this will not cover all possible realizations of that facial expression, this is likely to affect the performance of FED queries to a certain extent. Implementing each entry through a cluster however would mean that there would be no time to implement all the basic functionality of FED.

Because it is not unlikely that the representation of facial expressions in FED will be extended to a cluster representation in the future, all successfully labeled facial expressions will be stored in the FED database. This data could then later be used to form clusters of different realizations of facial expressions.

### 3.6 *Creating an ordering between entries in FED*

In a verbal dictionary, words are presented in alphabetical order. It is possible to look up the spelling of a word, sometimes the phoneme representation, the meaning in different contexts and rules of transformation. The words in a verbal dictionary are ordered and related to each other by means of their spelling.

A facial expression dictionary such as FED is an example of a Nonverbal Dictionary. Instead of words, a Nonverbal Dictionary consists of *nonverbal words*, i.e. facial expressions in the case of FED. Each facial expression entry in FED has a facial expression label associated with it. In principle, these labels could be used to create an ordering in the entries of the dictionary in the same manner as with a verbal dictionary. This however would not be the most logical or intuitive approach. The labels of two facial expressions may be very closely related, while the facial expressions itself are very different.

In order to create a more logical ordering of entries in a facial expression dictionary, the concept of a *nonverbal alphabet* could be used. Instead of words, the characters of the nonverbal alphabet are the Action Units (AU's) as defined by Ekman and Friesen in their Facial Action Coding System (FACS, see section 2.1.4). A facial expression is 'spelled' as an ordered list of active AU's with corresponding intensities. Similar facial expressions would then have a similar representation in the nonverbal alphabet.

In the case of FED, using AU's as characters of a nonverbal alphabet to describe facial expressions is not appropriate, since the presence of AU's in facial expression entries in FED is determined manually, and no intensity value is associated with an active AU; it is either present or it is not. Describing a facial expression by a nonverbal word consisting of ordered AU's and corresponding intensities would require automatic detection of the presence and intensity of AU's with high accuracy.

The choice was therefore made to define relations between FED entries by comparing the coordinates of the FCP's. Although the coordinates of the FCP's are normalized to compensate for differences in scale, position and in-plane orientation, it would be insufficient to simply define the degree of similarity between facial expressions in terms of the Euclidian distance between the 30 FCP's. This is because of the fact that certain facial expression can have a number of different realizations, whereas FED contains only one template expression representative of all these different realizations.

Instead of using the total Euclidian distance between all 30 FCP's, the choice was made to determine the degree of similarity between two facial expressions by comparing the normalized values of certain facial features. This choice was made based on the idea that all possible valid realizations of a facial expression are alike, when only looking at the shape of the essential facial features. Which facial features are essential can differ per facial expression. The best approximation would be achieved when the facial features that discriminate the most between all facial expressions in the FED database are used. A Visual Data Inspection Tool was developed to determine these most discriminative facial features.

The implementation of the nonverbal query is based on these most discriminative facial features, as is the implementation of the query were the user looks for a facial expression incrementally.

## Chapter 4: FED System Design

*This chapter describes the system design of FED. In the section 4.1, the design requirements are presented. In section 4.2, the global design of the FED system is laid out. Section 4.3 describes the design of the FED database. The design of the Visual Data Inspection Tool, that was used to determine the most discriminative facial features, is described in section 4.4. The design layout used to handle FED query requests and the design of the individual query components is described in section 4.5. Finally, section 4.6 describes the design of the management part of the FED system.*

### 4.1 Requirements

Because of the fact that FED has to become available as a website available to the general public, there are certain specific requirements to be met. Also, it is likely that FED will be extended and modified in the future, which imposes certain requirements as well. Finally, the FED system has to be easily manageable. The main requirements are:

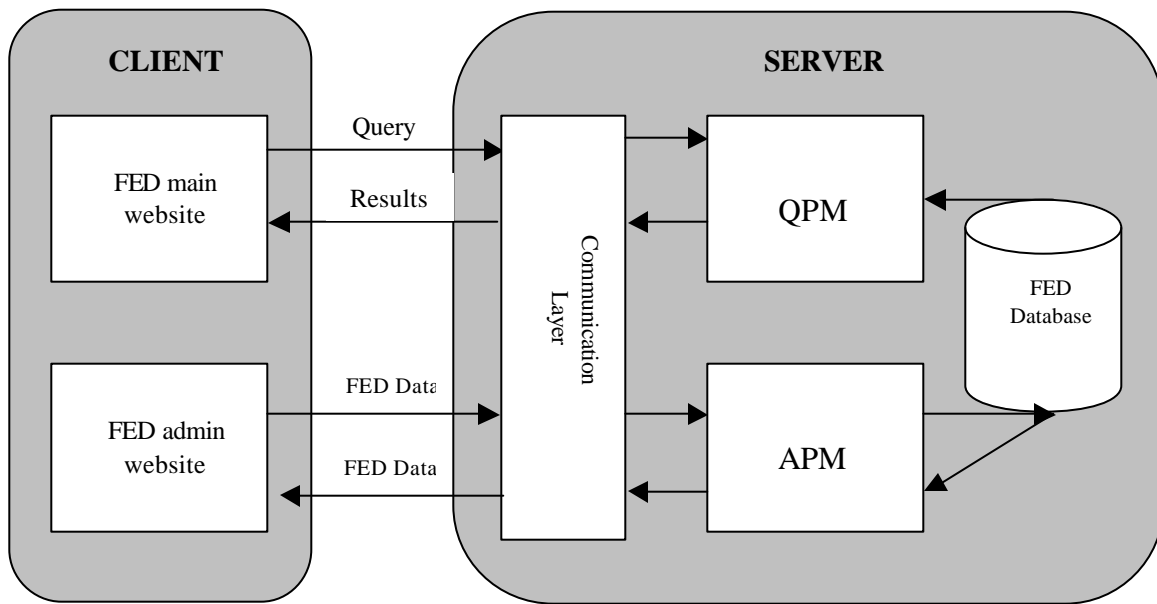
- *Real-time*: speed is of critical importance for a website. People who visit the FED website will not return there or recommend the site to other people, if they have to wait minutes for a query to be processed. It is therefore essential that query requests are processed in real-time.
- *Security*: a website is more vulnerable to threats like hackers or malice users than a stand-alone application. FED will have to be able to withstand these attempts to corrupt and/or crash the system.
- *Scalability*: if FED becomes a success, it is possible that hundreds of users will visit the site each day. The FED system will have to be able to handle all these visitors, or be easily upgradeable to handle this task.
- *Adaptability*: There will be not enough time to implement the most sophisticated techniques for all components of the FED system. Therefore, it is important that the FED system is set up modularly, so that future improvements can be implemented with relative ease.
- *Extendibility*: It is also possible that certain additions will be made to the FED website in the future. Additional query possibilities and management facilities could be implemented later. Again, a modular design is a good way to ensure that dependencies in the code are minimized and extending FED will not be too difficult.
- *Manageability*: it has to be easy to manage the entries in the facial expression dictionary. Because of the complexity of an entry in FED, it is extremely impractical to manage the FED entries manually, for example by issuing commands in the DBMS directly. An administrative mirror website is needed to provide easy management of the FED system (a mirror website is a website that enables manipulation of the data used in the main website; changes made to the data on the mirror website affect the content of the main website).

### 4.2 Global design

FED handles query requests via a client-server architecture. Figure 4.1 shows the global design of the FED system.



The individual components of the FED system can be described as follows:



**Figure 4.1: FED global design**

- The *FED main website* enables users to issue queries into FED. It consists of static HTML pages and Java applets. The HTML pages provide the user with information and are used to structure the layout of the website. The applets implement the GUI for issuing a query into FED. For each query, there exists an applet that implements the GUI for that query. When a user selects to issue a certain query, the appropriate applet is loaded from the server
- The *Communication Layer* of the FED system resides on the server and handles all data traffic between the client and the server. The communication layer consists of a collection of Java servlets.
- The *Query Processing Module* or *QPM* of the FED system also resides on the server and consists of several modules, each of which has the ability to process a specific type of query. Each module is implemented through one or more static Java classes.
- The *FED admin website* provides the GUI for the management part of the FED system. Like the main website, it consists of static HTML pages and Java applets. The differences with the main website are that the admin website is only accessible through user authentication, and that the applets don't provide the GUI for a certain query, but the GUI for a certain FED management function.
- The *Admin Processing Module* or *APM* implements the functionality needed to manage the FED system. Like the QPM, the APM consists of several modules that in turn consist of one or more static Java classes.
- The *FED Database* contains all the entries in the dictionary, admin user information, and log info. The PostgreSQL database management system is used to implement the database.

There are several advantages to this setup. Firstly, it ensures that the time to process a query is independent of the capacity of the computer of the user, because all calculations involved in processing a query are performed at the server, which is a high performance computer.

Secondly, this setup increases the security of the website. All database access takes place at the server. If database access took place from the applet on the client's computer, it would be relatively easy to obtain the database password and hack the FED system. There are several tools with which it is possible to obtain the source code of an applet by de-compiling the Java .class files.

A third advantage is that by separating the GUI, communication and processing modules, the FED system becomes more easily adaptable and extendible. If all query processing, communication and GUI code is located in one applet, the code can become very complex and thus difficult to adapt and/or extend.

Finally, by using Java servlets in the communication layer, it becomes easy to upgrade the capacity of FED, i.e. the number of simultaneous users the website can handle. This is because it is possible for the system administrator to set the amount of resources available to servlets.

### 4.3 The FED Database

Figure 4.2 shows the design of the PostgreSQL database of the FED system. The design is set up in such a way that it is possible to include the results of issued FaceShop queries and issued picture queries in FED in the future. These results can then be used to form clusters of different realizations of each facial expression entry in FED in the future, as mentioned in the previous section. The rest of this section describes the different tables of the design in more detail.

#### *tbl\_onvd\_entries*

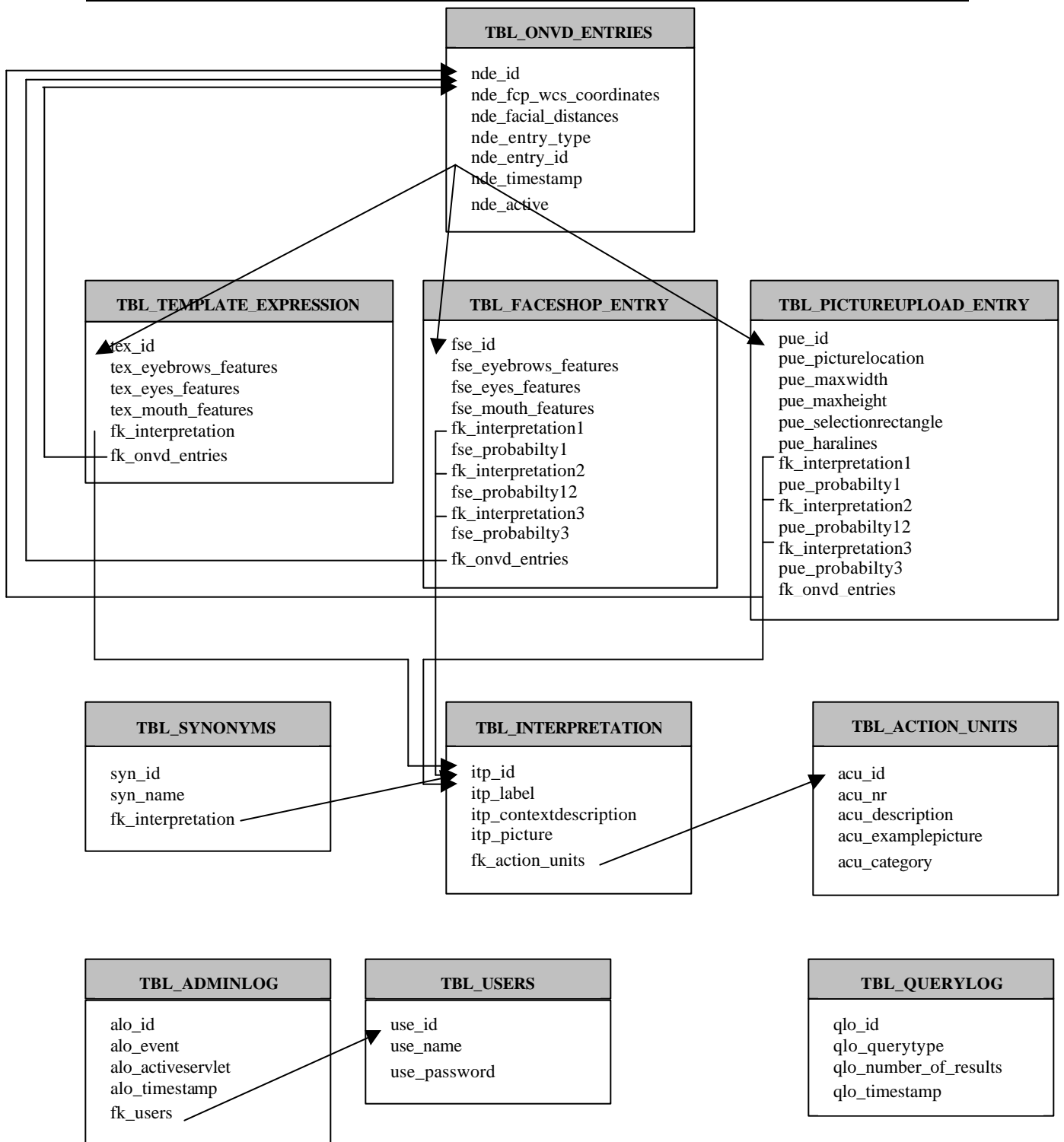
In the current version of FED, all queries issued into the FED system are processed using templates of facial expressions. For example, if a user sketches a facial expression with FaceShop, the features of the sketch are compared to the features of the template expressions in the database in order to obtain the best matching facial expression(s), and when a user issues a query on a keyword, the entered keyword is matched against the label, synonyms and description of the template expressions.

Because of the fact that template expressions, FaceShop query results and picture query results may be integrated into clusters in the future, the properties these database entry types have in common are stored in a single database table *tbl\_onvd\_entries*. This includes the coordinates of the FCP's and 21 so-called facial distances, which are a different representation of the FCP coordinates. This will make the implementation of the query processing modules when dealing with clusters easier.

Other information stored in *tbl\_onvd\_entries* includes a variable indicating if an entry is active, i.e. included when processing a query issued from the main website, a timestamp, and variables indicating the table and record id that contains the data specific to the FED entry.

#### *tbl\_template\_expression*

For each possible facial expression such as 'happiness', 'anger', 'sadness' etc., there exists exactly one entry in *tbl\_template\_expression*. In this table the values of the sliders of the FaceShop tool are stored. These are needed in order to be able to reconstruct a picture of the facial expression in FaceShop. Also present are a link to the template expression feature information in *tbl\_onvd\_entries* and a link to the interpretation data of the template expression: the facial expression label, label synonyms, description, example picture and active AU's. The

**Figure 4.2: FED database design**

interpretation data is stored in tables *tbl\_interpretation*, *tbl\_synonyms* and *tbl\_actionunits* which will be described in more detail later in this section.

*tbl\_faceshop\_entry*

When a user issues a query on the main website by sketching a facial expression with FaceShop, the FCP coordinates of the sketch are compared to the FCP coordinates of the entries in *tbl\_template\_expression*. to determine the closest matching facial expression(s).

If one or more matches are found, the query results are stored in *tbl\_faceshop\_entry*.

Like *tbl\_template\_expression*, *tbl\_faceshop\_entry* contains the values of the FaceShop sliders and a link to the facial expression feature information stored in *tbl\_onvd\_entries*.

Also stored are links to the interpretation records associated with the template expression(s) to which the sketch matched, and corresponding probabilities.

### *tbl\_pictureupload\_entry*

When a succesfull picture query was issued, the query results are stored in *tbl\_pictureupload\_entry*. This table contains the location of the uploaded picture on the server and data that is needed to reconstruct a picture upload FED entry: the maximum width and height at the moment the picture was cropped to fit the query GUI (which can vary depending on the screen resolution of the user), the coordinates of the subpart of the picture that was selected by the user in the face detection step and the coordinates of the so-called haralines, which will be discussed later in this chapter.

Like *tbl\_faceshop\_entry* and *tbl\_template\_expression*, *tbl\_pictureupload\_entry* contains a link to the facial expression feature information stored in *tbl\_onvd\_entries*. The matching results are stored in the same way as in *tbl\_faceshop\_entry*, namely through links to *tbl\_interterpretation* and corresponding probabilities.

### *tbl\_interpretation*, *tbl\_synonyms* and *tbl\_actionunits*

Each template facial expression has a certain interpretation. This interpretation includes the facial expression label, label synonyms, description, active AU's and an example picture showing the template facial expression. Tables *tbl\_interpretation*, *tbl\_synonyms* and *tbl\_actionunits* are used to store this information.

The facial expression label, label synonyms, description and the location of the example picture are stored in table *tbl\_interpretation*. Each synonym associated with the entry is represented through a record in *tbl\_synonyms*. Each record in *tbl\_synonyms* contains a reference to the record in *tbl\_interpretation* it is associated with. Finally, each record in *tbl\_interpretation* contains an array with references to records in *tbl\_action\_units* that represent the active action units of the facial expression.

### *tbl\_users*

*Tbl\_users* is used to store the login name and password of users of the administrative part of the FED system. When someone wants to gain access to the administrative part of the FED system, he has to supply a valid login name and password.

### *tbl\_adminlog* and *tbl\_querylog*

Two tables are used to store log information: *tbl\_adminlog* and *tbl\_querylog*. *Tbl\_adminlog* contains records of actions taken at the admin site. Each record stores the type of action, the servlet at which the action was performed, a timestamp and the user responsible (*alo\_event*, *alo\_activeservlet*, *alo\_timestamp* and *fk\_users*).

*tbl\_querylog* stores data on each query issued from the main website. This includes the type of query issued, the number of results the query returned and a timestamp (*qlo\_querytype*, *qlo\_number\_of\_results* and *qlo\_timestamp*).

#### 4.4 Visual Data Inspection Tool

In order to test a hypothesis about certain properties of a data set, *data mining* or *explorative data analysis* can be performed. In the case of FED, an example of a hypothesis would be ‘entries in FED where the mouth is not smiling also have closed eyes’. A problem can occur if a researcher has no idea about which hypothesis would be interesting to investigate. In that case, these techniques can also help a researcher to find interesting hypothesis.

A *Visual Data Inspection Tool (VDI Tool)* was developed as part of the FED administrative backend website. With this tool, an authenticated user has the possibility to define two facial features as a mathematical expression and view a scatter plot of the values of these features for all active facial expressions present in the FED database. Instead of defining a facial feature manually, it is also possible to select one of several predefined facial features for analysis. This tool can help a researcher test or find a hypothesis about the facial features of entries in FED. Several of the possible queries into FED are processed through determining the extent to which two facial expressions resemble each other. As mentioned in section 3.5, the choice was made to create an ordering between facial expressions in FED based on the values of the most discriminative facial features of entries in FED. The VDI tool can be used to find these most discriminative facial features.

Figure 4.3 shows the design of the VDI Tool. The entered features are sent to a parser for evaluation. If no parsing errors occur, the feature values of all facial expressions present in the FED database are calculated and passed along to a plotting canvas. Here the plotting coordinates of the origin are determined, and a scatterplot is generated with the first feature on the x-axis and the second feature on the y-axis. Finally, the scatterplot is displayed to the user. The user can select any point on the scatterplot to view the details of the facial expression(s) associated with the selected point.

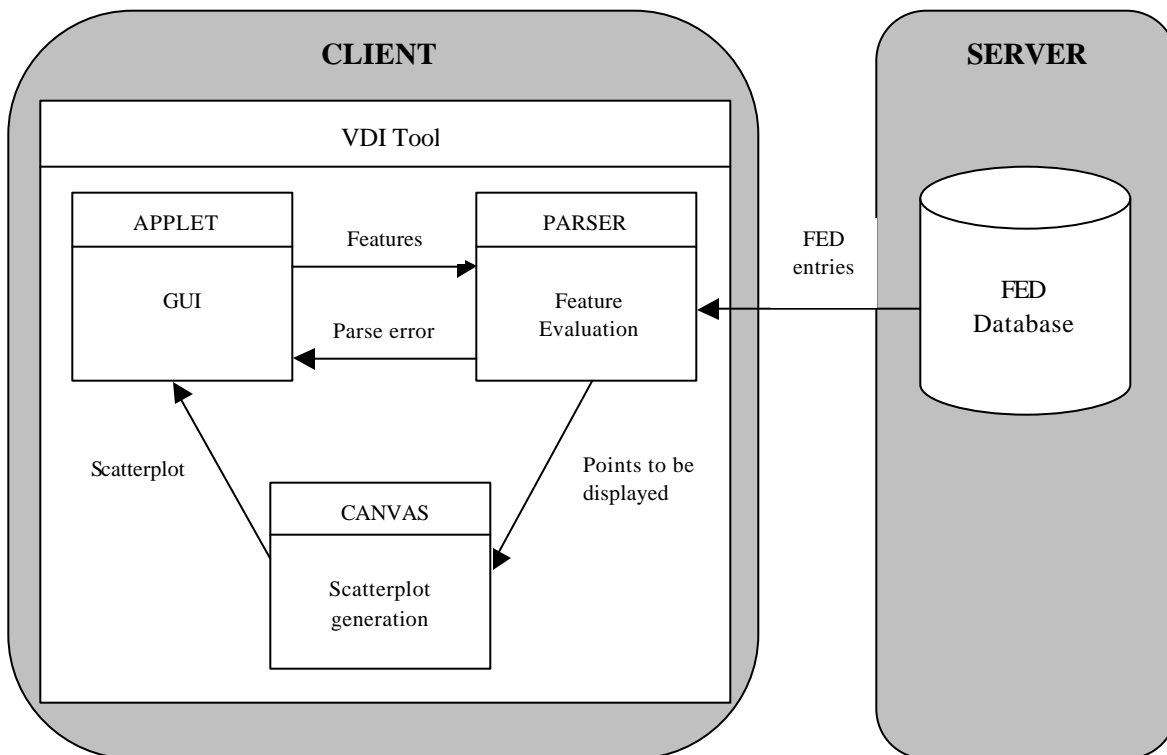


Figure 4.3: Visual Data Inspection Tool design

### 4.5 Issuing a query into FED

As mentioned in section 3.3, there are essentially six possible queries into FED. It's possible to issue a query on a facial expression label, active AU's, geometry features, issue a query incrementally, label a sketched facial expression and label an unknown facial expression shown in a picture. When issuing a query on a facial expression label, the entered query is also matched on label synonyms and words in the description of FED entries.

The design of the part of the FED system that handles query requests is set up modularly. The GUI, communication with the server and actual processing of each type of query request is implemented in separate components. This ensures that adapting the code will usually involve the adaptation of just one component, as long as the interface with the connected components doesn't change. Also, this makes it easier to add a new query possibility to FED: there are no dependencies between the components of different queries.

As shown in figure 4.4, the QPM of figure 4.1 actually consists of five separate Query Processing Modules. The labeling of an unknown facial expression, either sketched with FaceShop or shown in an uploaded picture, is done by the same QPM. The 6<sup>th</sup> component of the QPM is a PostgreSQL-to-Java conversion layer, which is needed to convert the FED entries from the database to a Java representation. The calculations that have to be performed on the FED entries are often too complex to be performed by basic SQL commands directly into the FED database. The 7<sup>th</sup> component of the QPM is a Java-to-PostgreSQL conversion layer. This layer is needed to insert the results of successfully labeled facial expressions in the FED database. Analogous to the QPM, the main website contains six applets, each implementing the GUI of one query type, and the communication layer contains six separate servlets for handling the data traffic involved in each query type.

Figure 4.5 shows the general design of how a query possibility is implemented in FED. When a user selects to issue a certain query, the applet that contains the GUI for this query is loaded from the server. When the user subsequently enters a query and selects to process it, the entered query is sent to the appropriate servlet of the communication layer on the server, which in turn invokes the QPM module of the selected query. The query QPM retrieves all FED entries from the database and processes the query. The results are returned to the communication layer, which in turn sends them to the applet where they are presented to the user.

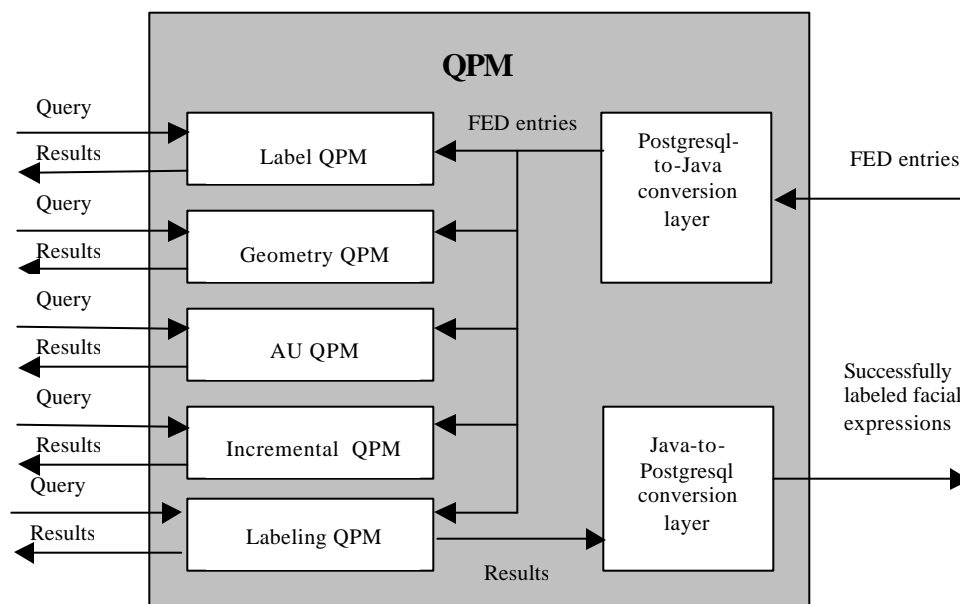


Figure 4.4: QPM design

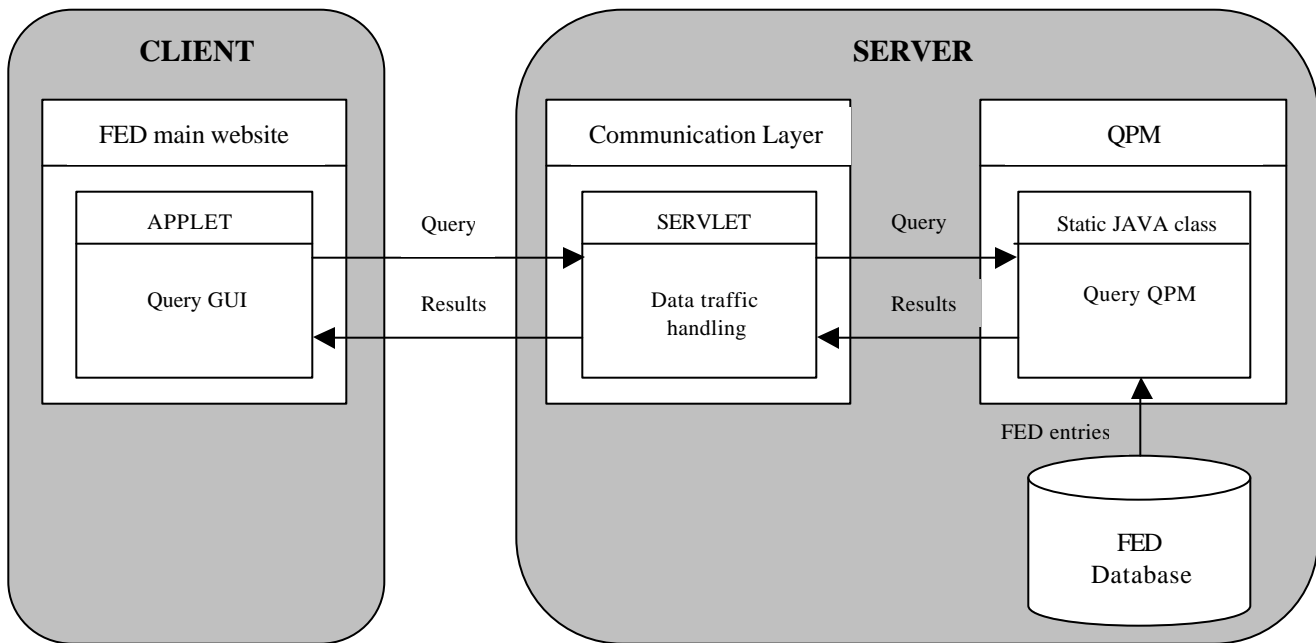


Figure 4.5: FED Query handling

#### 4.5.1 Label Query

Users can try to find an entry in FED by specifying the facial expression label. When a user selects to issue a label query at the FED main website, the applet that contains the label query GUI is loaded from the server, and the user is asked to enter the label of the facial expression he is looking for. Examples of label queries are 'happiness', 'stressed', 'surprise' etc. When the user subsequently selects to process the query, the entered keyword is send to the communication layer on the server, which in turn invokes the *Label Query Processing Module*, or *Label QPM* for short.

Figure 4.6 shows the design of the Label QPM. The Label QPM retrieves all FED entries from the database and will try to match the entered keyword on 3 parts of each FED entry: the facial expression label, the label synonyms and the description. If there are one or more matching entries found, they are send back to the applet as results. If no matching entry is found, a *Spelling Correction Module* is used to attempt to correct the entered keyword. It checks to see if the keyword matches a FED entry when allowing for one insertion ('happiness'), deletion ('hapiness') or substitution ('happuness'). If this is the case, the corrected keyword is send back to the applet as a spelling suggestion. If the corrected keyword still results in zero matches, no results are returned. All results are ordered by relevance, with FED entries matching on facial expression label appearing before entries matching on synonyms, which in turn appear before entries that match on a word in the description.

#### 4.5.2 Action Unit Query

A user can look for entries in FED by specifying which action units are active with the facial expressions he is looking for. Analogous to the Label Query described in the previous section, an applet containing the query GUI is loaded from the server when the user selects to issue an

Action Unit Query. This applet contains an enumeration of 30 AU's and corresponding checkboxes. The user can issue a query by selecting the appropriate AU's.

Figure 4.7 shows the design of the *Action Unit QPM*. It receives a query from the GUI applet through the communication layer. The entered query consists of a list of AU's. The Action Unit QPM retrieves all the FED entries from the database, and will then determine which entries have one or more active AU's in common with the AU's specified in the query. These entries are ordered by relevance and returned as query results (matching entries that have the most AU's in common with the query appear first).

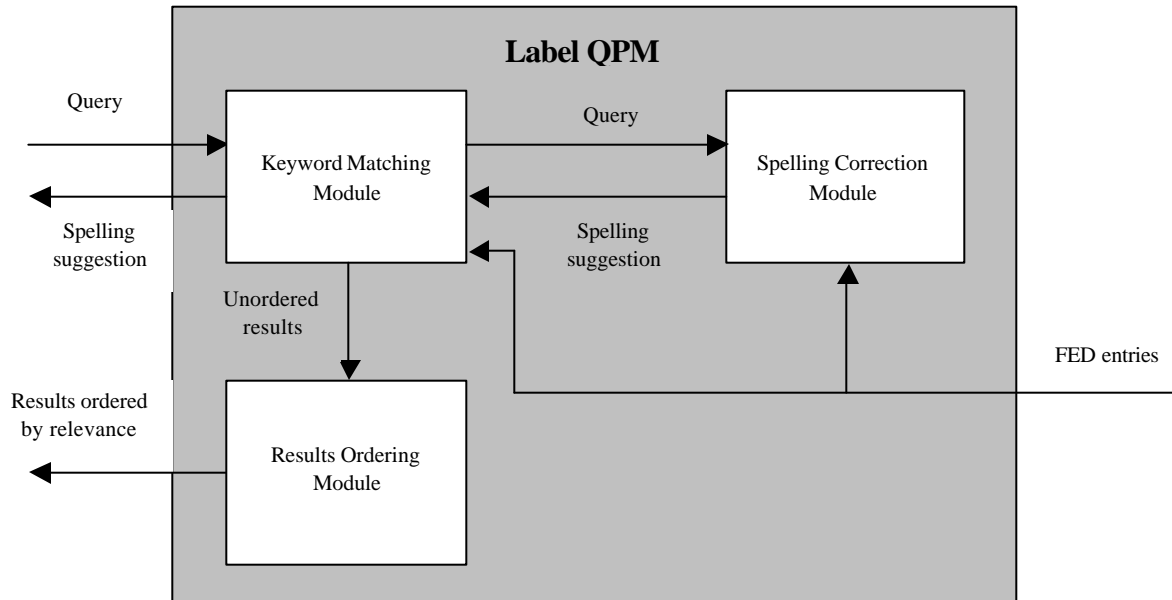


Figure 4.6: Label QPM design

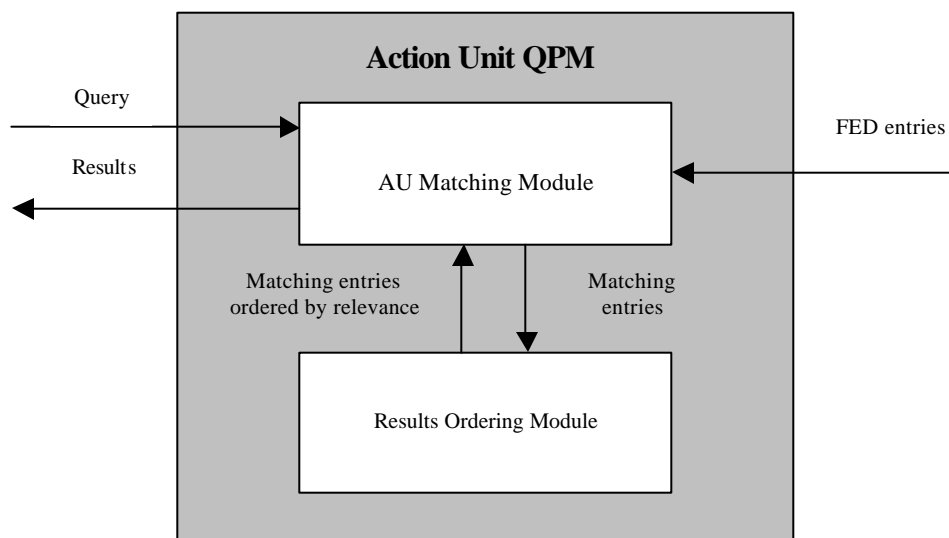


Figure 4.7: Action Unit QPM design



### 4.5.3 Geometry Query

Because of the fact that each entry in FED contains the coordinates of the 30 FCP's, it is possible to issue a query for all entries in FED that exhibit certain geometrical features. The user is restricted to enter queries that confirm to a certain format. The reason for this is that successfully parsing a query could become very difficult if users are allowed to enter a geometrical query in any format they like. A query such as 'well I would like to see all entries where the person has his eyes closed' would be very difficult to parse correctly. It's possible to issue a so-called *basic geometry query* on 16 possible facial features (see table 4.1). When a query is submitted for processing, the coordinates of the FCP's of all FED entries are examined to determine which facial expressions exhibit the indicated geometrical feature.

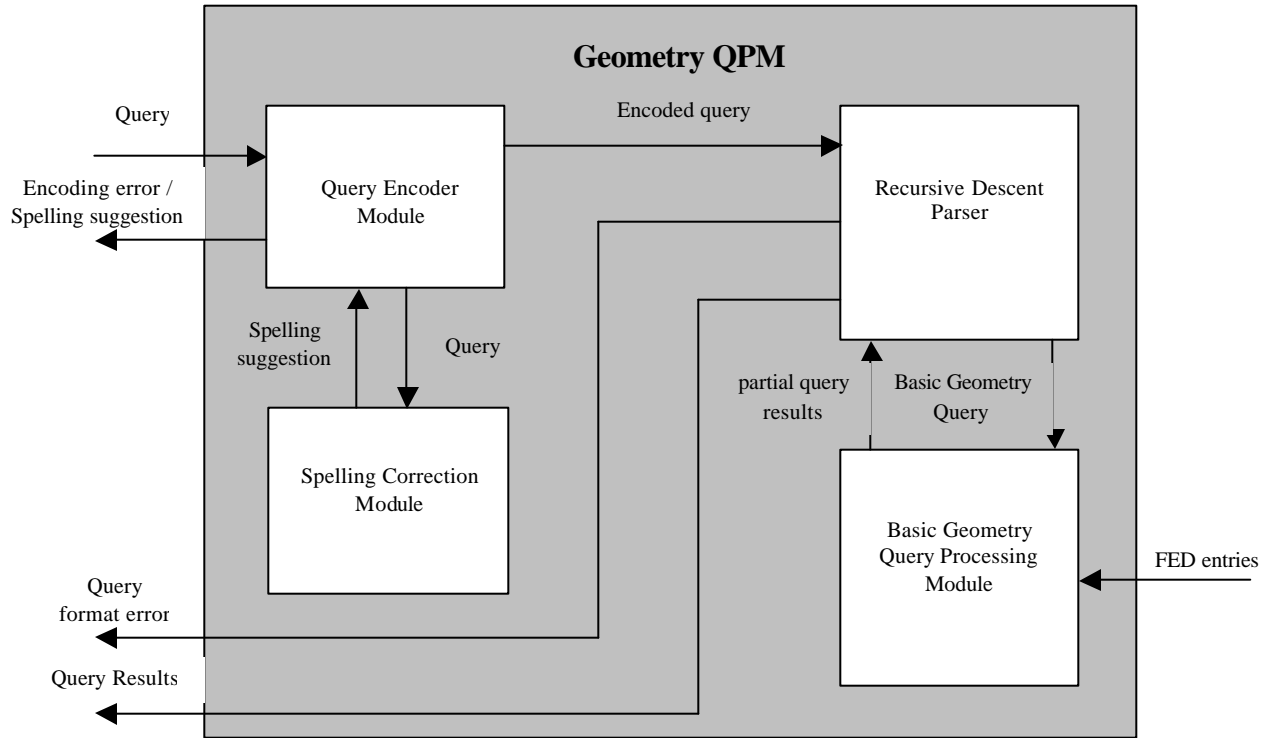
**Table 4.1:**  
**Basic Geometry Queries**

<b>Facial Feature</b>	<b>Possible query</b>
Eyebrows	raised
	frowned
	close together
	far apart
Eyes	wide open
	open
	slit
	closed
	slanting up
Mouth	slanting down
	wide open
	open
	smiling
	not smiling
	stretched horizontally
	stretched vertically

It is also possible to issue an aggregate query using the logical AND ('&') and OR ('+') operators. For example, it is possible to issue a query such as 'mouth open & eyes slit', which will returns all facial expressions that exhibit both of these features. Furthermore, a query can be made infinitely complex by using the bracket operators '(' and ')'. Using brackets, queries of the form '(mouth open & eyes slit) + eyebrows raised' can be issued. This particular query will return all FED entries with either the eyebrows raised or with both the mouth open and the eyes slit.

Figure 4.8 shows the design of the *Geometry QPM*. The *Query Encoder Module* first encodes the query to a simplified form. This reduces the complexity of the implementation of the *Recursive Descent Parser*. While encoding the query, the syntax of each encountered basic geometry query is checked for validity. Again, as with the Label Query, a spelling suggestion is generated if a basic geometry query is invalid but matches a valid basic geometry query when allowing for one insertion, deletion or substitution. When the encoding of the query is successful, the encoded query is send to the Recursive Descent Parser, which will execute all basic geometry queries and eventually return all FED entries that meet the restrictions of the entered query. The parser makes

use of a *Basic Geometry Query Processing Module* that is able to retrieve all FED entries with a certain geometrical feature present.



**Figure 4.8: Geometry QPM design**

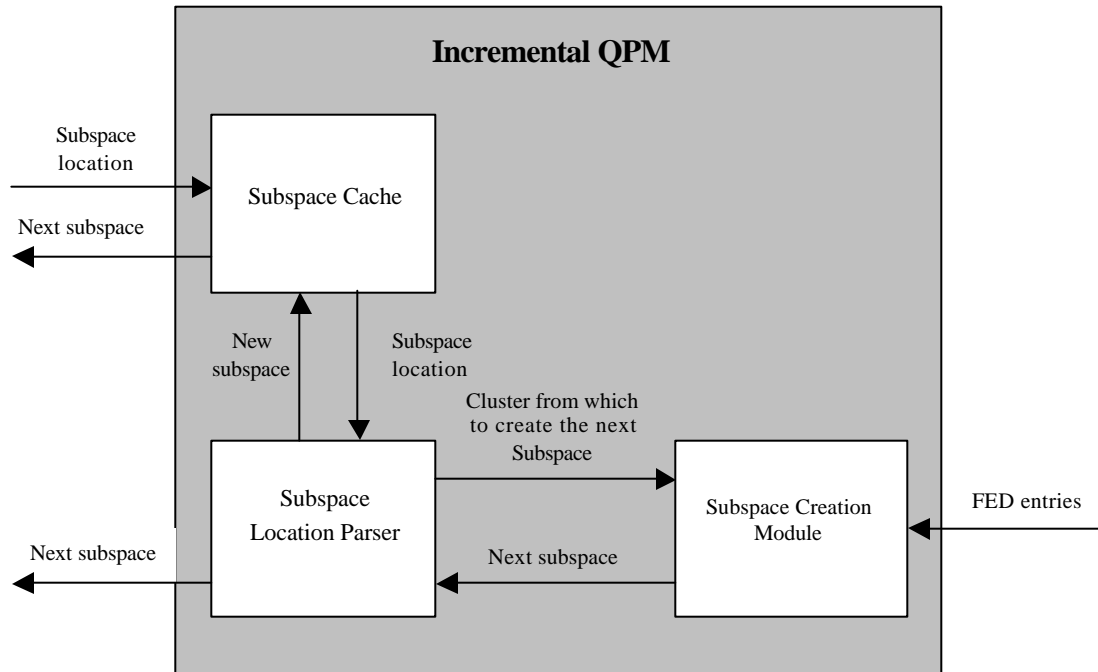
#### 4.5.4 Incremental Query

When a user doesn't know the label, active AU's or geometrical features exactly, the Incremental Query allows him to find the facial expression he is looking for incrementally. Initially, the user is presented with four facial expressions (*pivot expressions*) representative of four clusters covering the complete FED database.

The user now has to select the pivot expression that resembles the facial expression he is looking for the closest. Subsequently, the cluster corresponding to the selected pivot is again divided into four clusters with corresponding pivot expressions, which are again displayed to the user. These new pivot expressions more or less resemble the selected pivot expression, and are more alike each other than the pivot expressions of the previous iteration. This process can be repeated until a selected cluster consists of just one facial expression. If the facial expression that the user was looking for appears as a pivot, the user can view the details of that particular FED entry. In order to successfully divide a collection of facial expressions into four equal clusters, the most discriminative facial features of all facial expressions of FED were used to determine the extent to which two facial expressions are alike.

Figure 4.9 show the design of the *Incremental QPM*. A subspace in the context of the Incremental Query is defined as the four clusters and corresponding pivot expressions that equally divide a certain set of facial expressions from the FED database. The state of the Incremental Query is determined by the subspace that is shown to the user. The state is encoded in the form of a *subspace location string* that consists of integers indicating the next cluster from which the next subspace is to be created.

Firstly, the Incremental QPM checks to see if the subspace corresponding to the subspace location string is stored in the *Subspace Cache*. If this is the case, the next four clusters and pivots can be returned immediately. If this is not the case, the *Subspace Location Parser* iteratively parses the subspace location string, using the *Subspace Creation Module* to determine the next



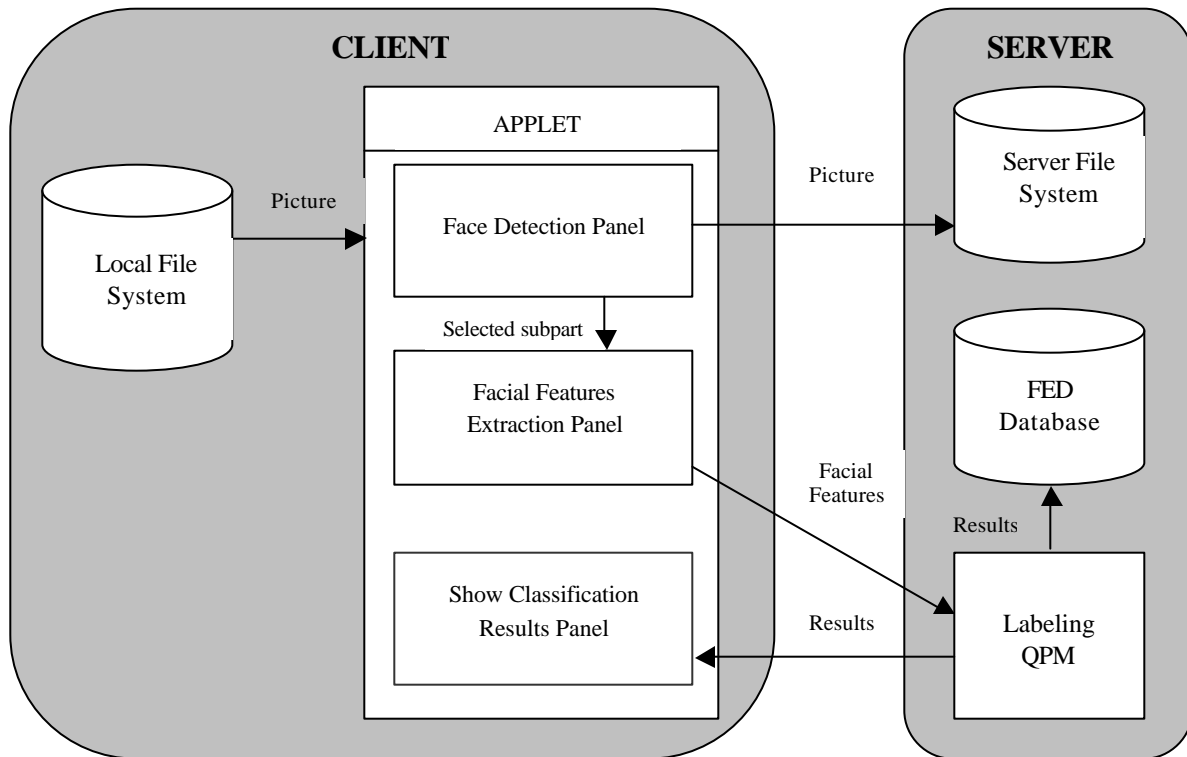
**Figure 4.9: Incremental QPM design**

subspace at each iteration. A subspace is constructed by ordering all facial expressions of the set that is to be divided by feature values from low to high. The facial features used in this algorithm are determined through visual data inspection and are the most discriminative between all facial expressions in the FED database. Each subspace that is created is stored in the Subspace Cache, if it is not already present. The Subspace Cache is cleared whenever a FED entry is added, edited or deleted from the FED administrative website, to avoid outdated subspaces with data no longer accurate.

#### 4.5.5 Picture Query

The Picture Query allows a user to determine the label of an unknown facial expression. Implementation of this type of query requires techniques from the field of automatic facial expression recognition. As mentioned in section 2.2, there are three steps that have to be performed by any automatic facial expression recognition system: face detection, facial features extraction and classification. In the case of FED, the user performs the face detection step manually. The feature extraction takes place semi-automatically.

Figure 4.10 shows the design for the Picture Query implementation. Each step in the recognition process is implemented in a different panel. An applet serves as a container for these three panels. The user has to select a picture from his local file system, which is uploaded to the server and displayed on the first panel of the applet. Here the user can make a selection of the picture containing the face. Once he has made a selection and selects to continue, the selected subpart is scaled and displayed on the second panel, where he has to determine the position of the 30 FCP's semi-automatically. When all FCP's have been positioned, the user can select to process. The



**Figure 4.10: Picture Query design**

facial features are then send to the *Labeling QPM*, where the closest matching facial expression(s) are determined using the most discriminative facial features (identical to the facial features used for the Incremental QPM, determined by visual data inspection) . If the labeling process was successful, the results are stored in the FED database. The third and final panel is used to display the results of the facial expression labeling process.

#### 4.5.6 FaceShop Query

The FaceShop Query also allows a user to determine the label of an unknown facial expression. Instead of uploading a picture and determining the facial features semi-automatically however, the user only has to sketch a facial expression with FaceShop. The coordinates of the FCP's are determined automatically. Once the user has sketched a facial expression, he can select to process. As with the Picture Query, The labeling process is also performed by the Labeling QPM. Again, the classification results are stored in the FED database if the labeling was successful.

### 4.6 FED Management

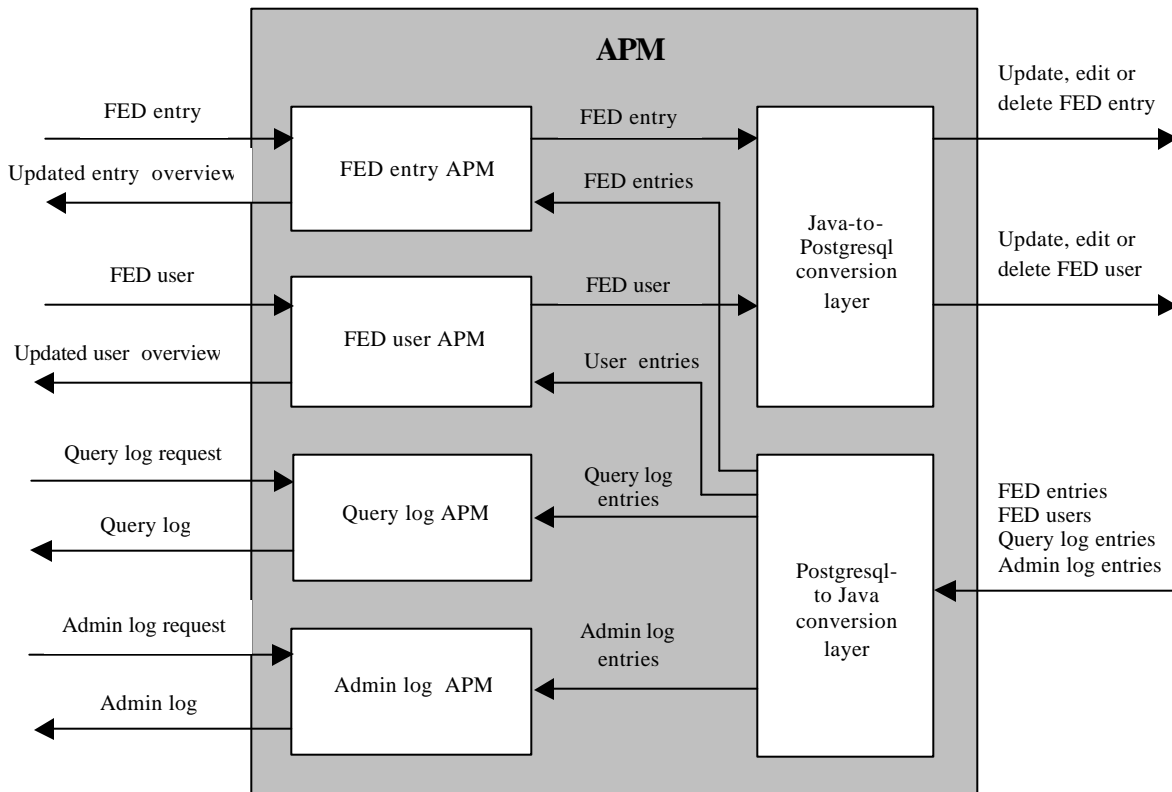
The administrative part of the FED system is restricted to authenticated users. This means that people can only gain access to the admin website by supplying a username and a password. Once a user has successfully logged in, the admin site is loaded. The admin site provides functionality to manage FED entries FED and admin users, which can be added, edited or deleted. Also, a log of the queries issued from the main website and a log of the actions taken by users at the admin site can be viewed here.

Again, a modular design is used. The GUI for each admin function is implemented in a specific applet, the communication with the server is accomplished through a specific servlet and the

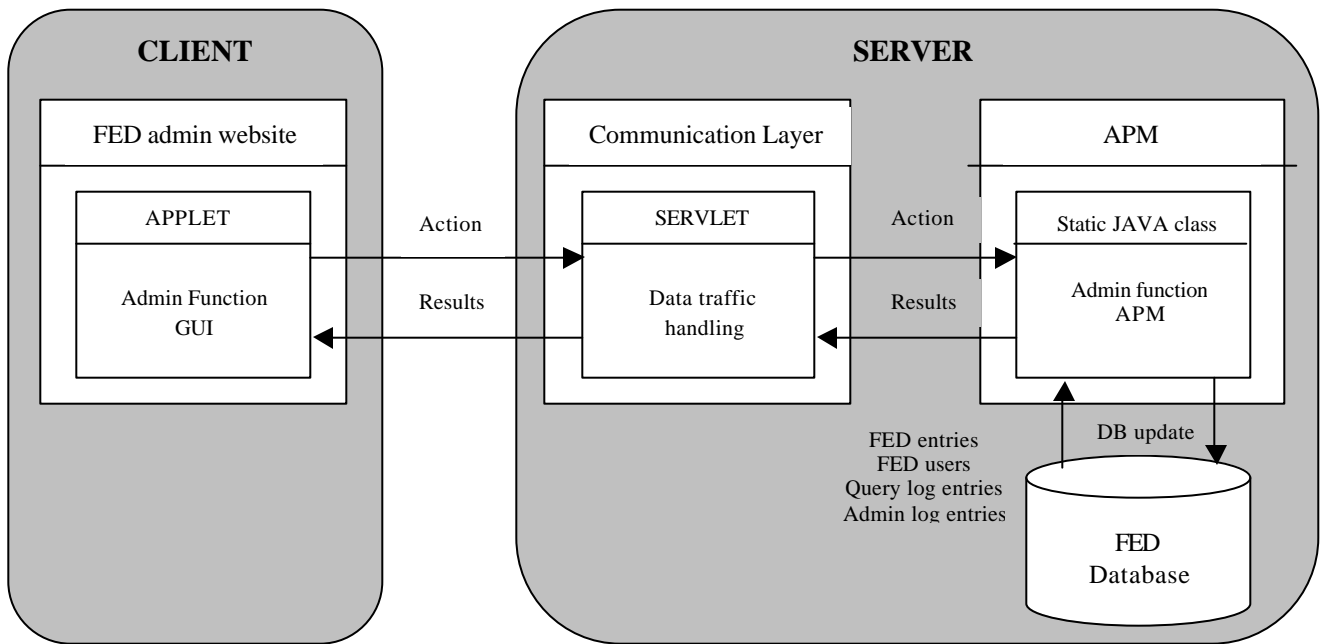
actual admin operations are performed by a specific Admin Processing Module. Just like the QPM, the APM thus consists of several APM modules. Figure 4.11 shows the design of the APM. Besides four admin processing modules, also present are a PostgreSQL-to-Java and a Java-To-PostgreSQL conversion layer. These have the same function as with the QPM of figure 4.4. Figure 4.12 shows the general design of how an admin function is implemented in FED.

When the user selects to manage the FED entries, an applet that gives an overview of all the FED entries is loaded from the server. When the users selects to add, edit or delete a FED entry, the entry is send to the appropriate servlet in the communication layer, which in turn invokes the FED entry APM. The APM performs the requested update of the FED entries in the database, and returns an updated overview of the FED entries to the applet, where they are again presented to the user. Managing admin users is analogous to managing FED entries.

When the user selects to view the query log or the admin log, this request is also send to the Query log APM or Admin log APM via the communication layer. The specified log is constructed from the log entries in the database and is send back to the applet.



**Figure 4.11: APM design**



**Figure 4.12: FED Management**

## Chapter 5: FED Website Implementation

*This chapter describes the development details of FED. In section 5.1, the tools used during the development are described. The implementation details of the FED website are presented in section 5.2. Section 5.3 gives insight into the data acquisition process of FED. Finally, section's 5.4 and 5.5 handle the implementation details of FED user management and logging.*

### 5.1 Development tools

In this section the different tools and techniques used in the development of FED are described. The first two subsections give a description of the two main tools used to implement FED: the Java programming language and the relational database management system PostgreSQL. A comparison with similar tools and the motivation for using Java and PostgreSQL is given here too. In the third subsection, the FaceShop tool for generating facial expressions is described. For completeness, the last subsection gives an overview of all the other tools used in the development of FED.

#### 5.1.1 Java

Information on the internet can be presented through the use of Hypertext Markup Language (HTML). HTML is a commonly agreed upon standard (defined by the IETF in RFC 1866), and all internet browsers are able to parse and display HTML pages correctly (although some browsers allow proprietary HTML extensions that others do not). HTML is platform independent (all you need is an internet browser) and has been in use since 1990.

When the internet and its popularity expanded, the need for more dynamic websites developed. Many so-called scripting languages, which allow for the creation of dynamic websites, have been developed in response to this need. Two types of scripting can be distinguished: client-side scripting and server-side scripting.

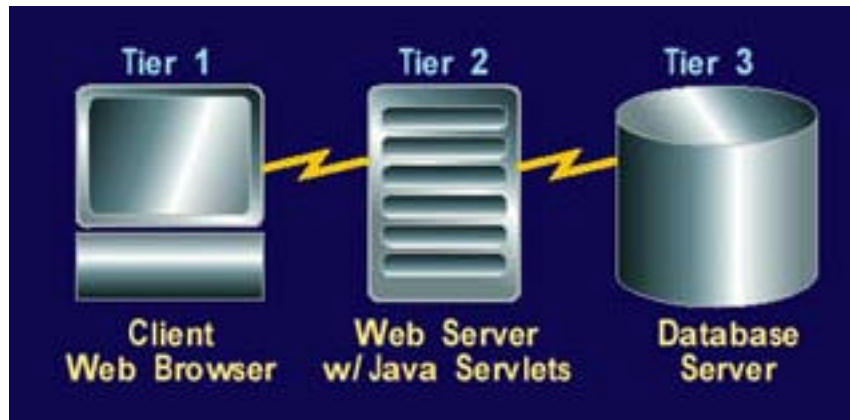
Client-side scripts are contained within HTML pages and are therefore sent to the client whenever he requests a HTML page from a server. Client-side scripting is often used to allow more dynamic user interaction. An example of client-side scripting is JavaScript, which can be used to change the markup properties and content of a HTML page (i.e. color, size) depending on actions performed by the user.

Server-side script runs at the server and allows for dynamic generation of HTML pages and interaction with databases residing on the server. An example of server-side scripting is ASP (Active Server Pages). Instead of issuing a request for a HTML page, a client can also make a request to a server for an ASP page. The server compiles the requested .asp file depending on the values of parameters being passed along in the request. The result of this compilation is a plain HTML page, which is eventually sent back to the client. This HTML page can contain client-side script again, such as JavaScript.

Since 1994 there exists another possibility for creating websites. This came with the introduction of the Java programming language. With Java it is possible to develop both applications and websites. Compiling Java code results in files consisting of *java bytecode* (class files). In order to run applications or websites developed in Java, the *Java Virtual Machine* (JVM) is needed, which has the ability to parse and run Java bytecode. Since nowadays most browsers contain an implementation of the JVM, websites developed in Java can be categorized as platform independent, just as websites consisting of HTML pages. The main difference between using HTML and Java is that HTML is essentially a language used for representing information on the web, whereas Java focuses on the creation of web applications.

Java provides the possibility of developing the client-side part of a website through applets. Applets are Java programs that can run in a web browser and are sent to the client as a whole. Applets provide greater possibilities for creating fast, dynamic user interaction than HTML pages containing client-side scripts such as JavaScript.

One way Java provides server-side ‘scripting’ is through so-called servlets. Java servlets run at the server and can, just as applets, make use of all the functionality that the Java language has to offer, except for the fact that servlets have no graphical user interface. Servlets are considered an extension of the capabilities of the web server and can be used to perform any task: perform calculations, manage database access or dynamically generate HTML pages. In comparison with most server-side scripting, servlets have far greater possibilities because of the fact that Java is a full-fledged programming language. Another advantage of servlets is that they make a website more easily scalable and help modularize the design. Figure 5.1 shows a so-called three-tier design, where the GUI, website logic and data storage are all placed in a different layer, with servlets handling the data processing and data traffic between the GUI and the database of the website.



**Figure 5.1: Three-tier website design**

FED is a highly interactive website. Also, relatively complex operations have to be carried out because of the involved image processing and facial expression classification. Because of this, I chose to develop FED with Java applets and servlets. Java applets provide better possibilities than HTML in combination with JavaScript for creating a highly dynamic, interactive graphical user interface and performing image processing operations. Java servlets are better suited for performing the facial expression classification than scripting languages, because servlets provide you with all the functionality of the Java programming language. It would be possible to use a scripting language to do this as well, but usually only limited libraries are provided, making the realization of complicated functionality difficult and time-consuming. A possible disadvantage of using Java comes from the fact that the performance may not be too good with real-time applications. Efficient programming is thus important.

### 5.1.2 PostgreSQL

FED uses a database to store the feature vectors and corresponding facial expression information of uploaded pictures (to improve performance, the pictures themselves are saved as files on the file system of the server). The classification algorithm of the used in labeling unknown facial



expressions needs this data in order to classify the facial expression shown in an uploaded picture. Also, all other queries use the information in the database to retrieve the correct results.

There are many possible database management systems (DBMS) that could be used for this purpose. Because I had worked with SQL Server for Windows, a similar database management system had my preference. My first choice was MySQL, a less sophisticated version of SQL Server that provides all the functionality needed. Also, a free UNIX version can be downloaded from the internet. However, at the Faculty of Knowledge Based Systems, there already was another DBMS installed, PostgreSQL.

PostgreSQL is an open source object-relational database management system developed at the University of California at Berkeley. It would be convenient to use PostgreSQL, because the whole DBMS was already in place. Upon examining the functionality provided by PostgreSQL, it turned out that it was similar to the DBMS's I had worked with and could also be used to develop the FED backend. It supports most SQL (Structured Query Language) constructs, allows the creation of user-defined types and functions and very large amounts of data can be stored (see table 4.1). This is important since FED needs to store large amounts of data per entry.

**Table 5.1**  
**Limitations of PostgreSQL 7.1**

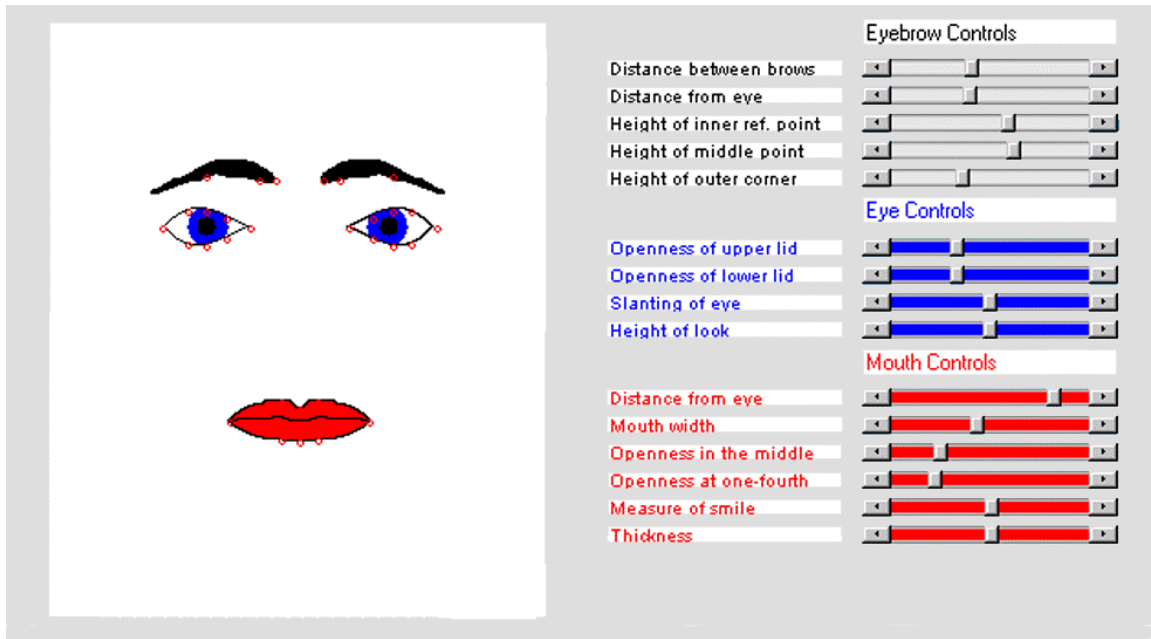
Feature	Limitation
Maximum size for a database	Unlimited (60GB databases exist)
Maximum size for a table	64 TB on all operating systems
Maximum size for a row	Unlimited
Maximum size for a field	1GB
Maximum number of rows in a table	Unlimited
Maximum number of columns in a table	1600
Maximum number of indexes on a table	Unlimited
Of course, these are not actually unlimited, but limited to available disk space and memory/swap space. Performance may suffer when these values get unusually large.	

### 5.1.3 FaceShop

When creating a dictionary, it is desirable to fill the dictionary with as many entries as possible. For each entry in FED, the coordinates of the 30 FCP's as defined in the face model of Kobayashi and Hara have to be stored as well. Otherwise, a number of queries could not be implemented. In principle, it is possible to determine the FCP coordinates of FED entries manually. However, this can be a time consuming and possibly inaccurate process. For this reason, FED was filled with entries using FaceShop.

FaceShop is a drawing tool for generating facial expressions developed at the Knowledge Based Systems group of the TU Delft. It uses 3 types of control sets that allow a user to draw a facial expression: an eyebrow, eye and mouth control set. Each control set contains a number of facial expression features that can be manipulated. Each individual feature can be manipulated by changing the value of a slider. The major advantage of using FaceShop is that the coordinates of the FCP's can be determined automatically from the values of the sliders.

Figure 5.2 shows the FaceShop tool. The red dots indicate the position of the FCP's. Whenever the user changes the value of one of the sliders, the coordinates of the FCP's are recalculated and the red dots will move to the new coordinates.



**Figure 5.2: The FaceShop tool**

#### 5.1.4 Other tools used during development

An overview of all the other tools used during the development and deployment of the FED website is given in table 5.2. Emacs is a text editor available for Unix that allows for integration with the Java compiler. Apache is a free web server that also provides the possibility of running Java Servlets through a program called Tomcat.

**Table 5.2**  
**Tools used during the development of the Nonverbal Dictionary**

Tool	Type	Version
Operating System	SUN OS	6
Programming language	Java	JDK 1.2
DBMS	PostgreSQL	7.1.3
Text editor	Emacs	20.4.1
Servlet Engine – development	Java Servlet Development Kit (JSDK)	2.1
Servlet Engine – livesite	JSDK (to be migrated to Tomcat)	2.1
Web server	Apache	1.3.14

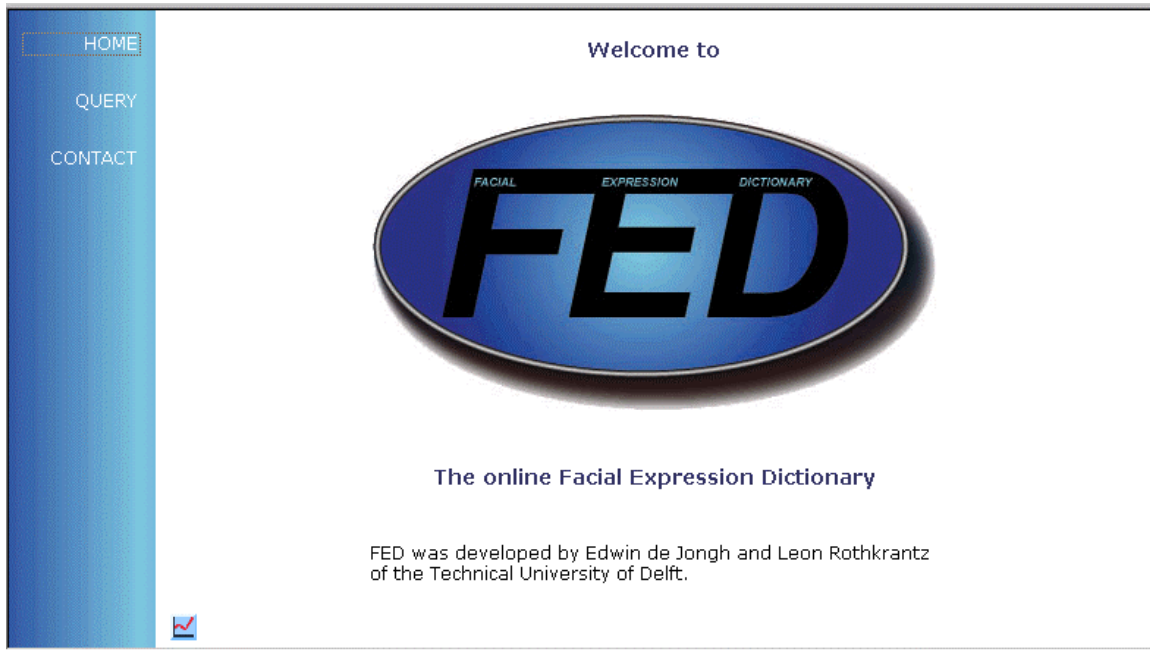
## 5.2 The FED website

Figure 5.3 shows the architecture of the website that is used to implement FED. As mentioned in chapter three, the FED main website provides users with the possibility to issue a query into FED. Static HTML pages are used to structure the website, and for each possible query, a Java applet implements the GUI for that query. Furthermore, query requests are processed at the server. Java servlets handle all the communication between the client and the server, and static

Java classes implement the code that processes the queries and performs all operations on the FED database.

The FED admin website provides authenticated users with the possibility to manage the FED system. As becomes clear from figure 5.3, the login information supplied by a user is verified by a servlet using the user information stored in the FED database. Only if the user has supplied a valid login name and password will the admin website be loaded. The HTML pages of the admin website are always generated by a dedicated servlet. This ensures that malice users are unable to load certain HTML pages belonging to the admin website directly. The setup of the admin website is the same as with the main website: static HTML pages, applets, servlets and static java classes work together to provide the FED management functionalities.

Figure 5.4 shows the starting page of the main website. The user is presented with a welcome text explaining the FED website. Users can choose to issue a query by selecting this option in the menu on the left. Figure 5.5 shows the main query page that is then displayed. This page gives an explanation about the six possible queries into FED. When a user selects one of the query links, a HTML page explaining the selected query in detail is displayed. From this page, the user can start the query. The appropriate applet containing the GUI of the query will then be displayed to the user.



**Figure 5.4: FED website start page**

### **5.3 Data acquisition**

Scientists have determined that there exist approximately 7000 different facial expressions. The number of possible combinations of 44 action units is of course much larger, but not every combination can be realized as a facial expression or is a meaningful facial expression.

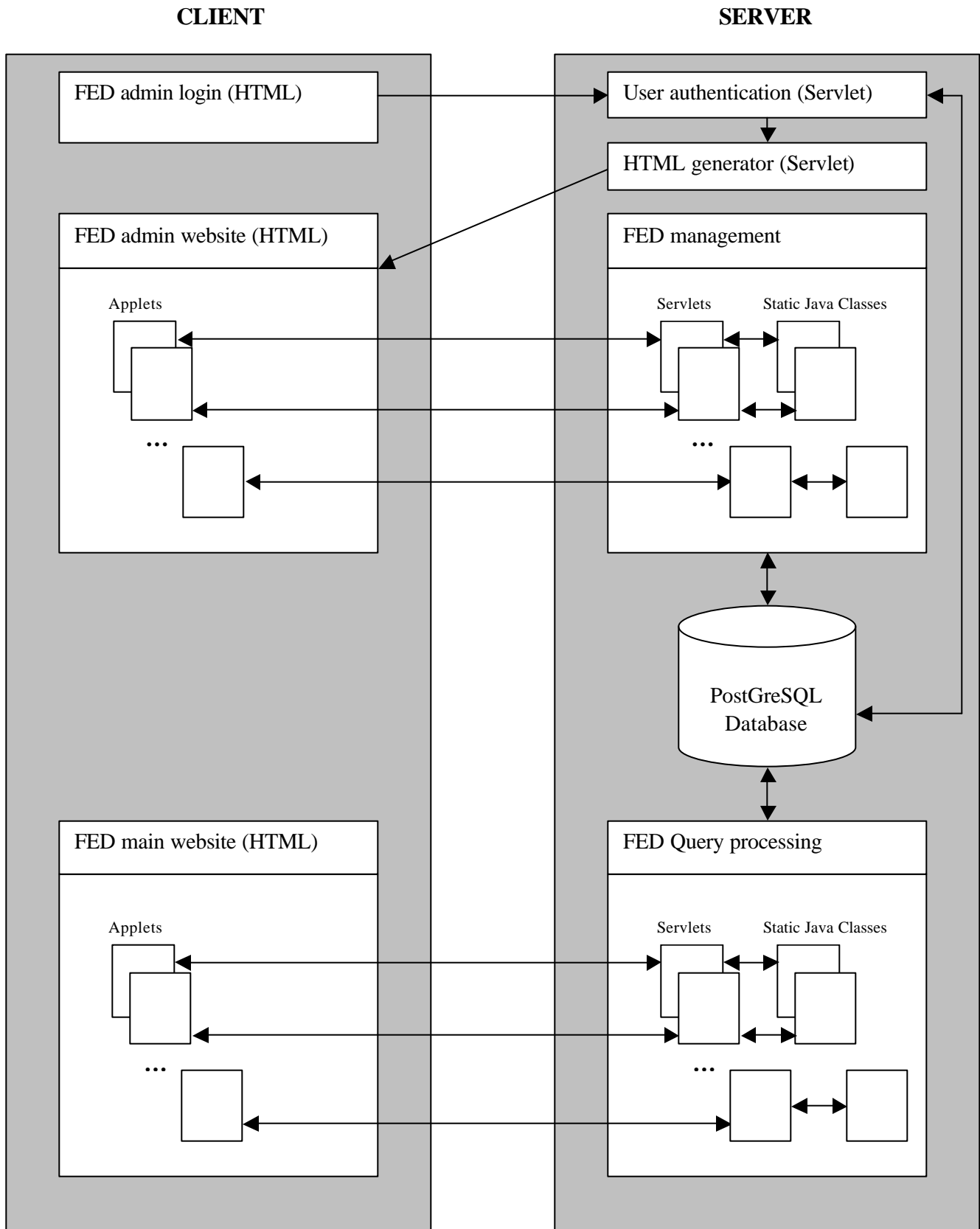
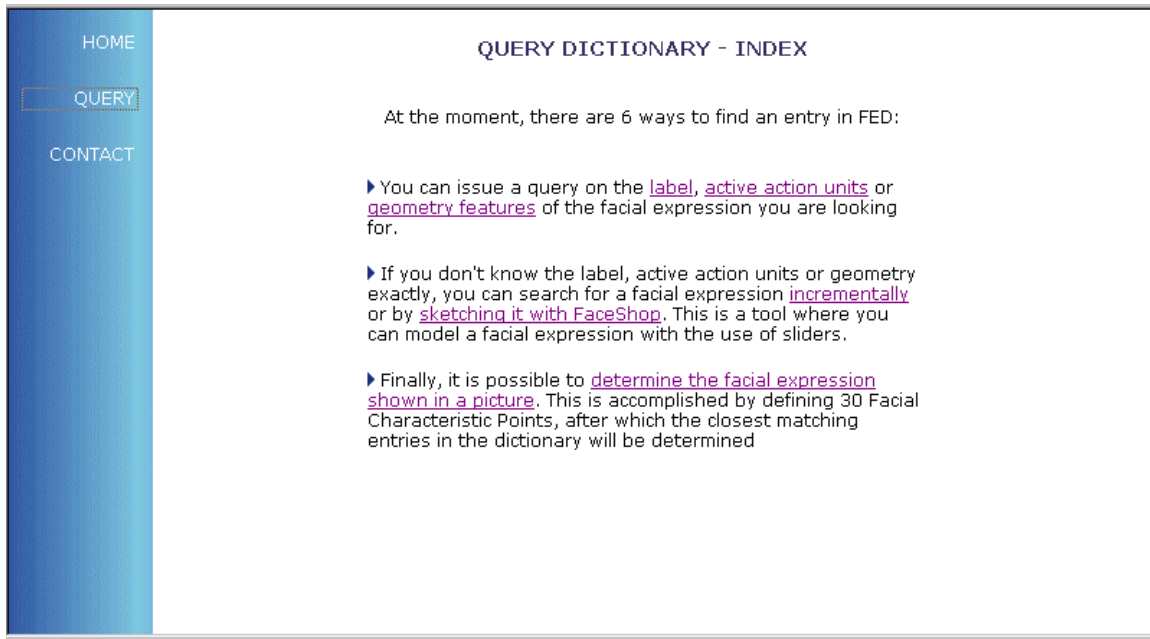


Figure 5.3: FED website architecture

Ideally, FED would contain all 7000 possible facial expressions, each with a scientifically valid interpretation. The primary goal for this graduation project however is to develop a prototype. This means that it is more important that the functional requirements of FED are met than that the data in FED is complete and scientifically valid. Also, filling the database with all 7000 entries is a graduation project in itself, and ensuring the scientific validity of each entry and its interpretation is best left to psychologists.

For this reason, the goal was to fill the FED database with approximately 60 entries. This is a sufficient number of facial expressions when trying to determine if the FED system functions correctly and if it is a viable implementation of an online facial expression dictionary.

The remainder of this section describes the two methods used to add entries to FED: adding entries through the FED admin site and adding entries through the AddTemplateTool.



**Figure 5.5: FED main query screen**

### 5.3.1.1 Adding entries through the FED Admin site

As mentioned in section 5.1.3, FaceShop is used to create a facial expression that is to be inserted into the FED database. The other information that makes up a FED entry includes the facial expression label, the label synonyms, a description, the active AU's and an example picture, as indicated in section 3.2.

The primary method for adding an entry to FED is by using the management facilities of the administrative backend website. Here an authenticated user can use FaceShop to sketch the new facial expression and enter all additional properties of the FED entry. Figure 5.6 shows a screenshot of the main screen of the FED entry management tool. Here the user can select to add or edit a FED entry. The 'next 10 templates' and 'previous 10 templates' buttons can be used to browse thru the FED entries. The 'Write TE Data to file' button writes the FCP coordinate data of all FED entries to a file that can be used for statistical analysis.

Figure 5.7 shows a screenshot of the screen used to add a FED entry. The user can add the new entry by pressing the 'save' button or return to the main screen by pressing 'cancel'. The 'active'

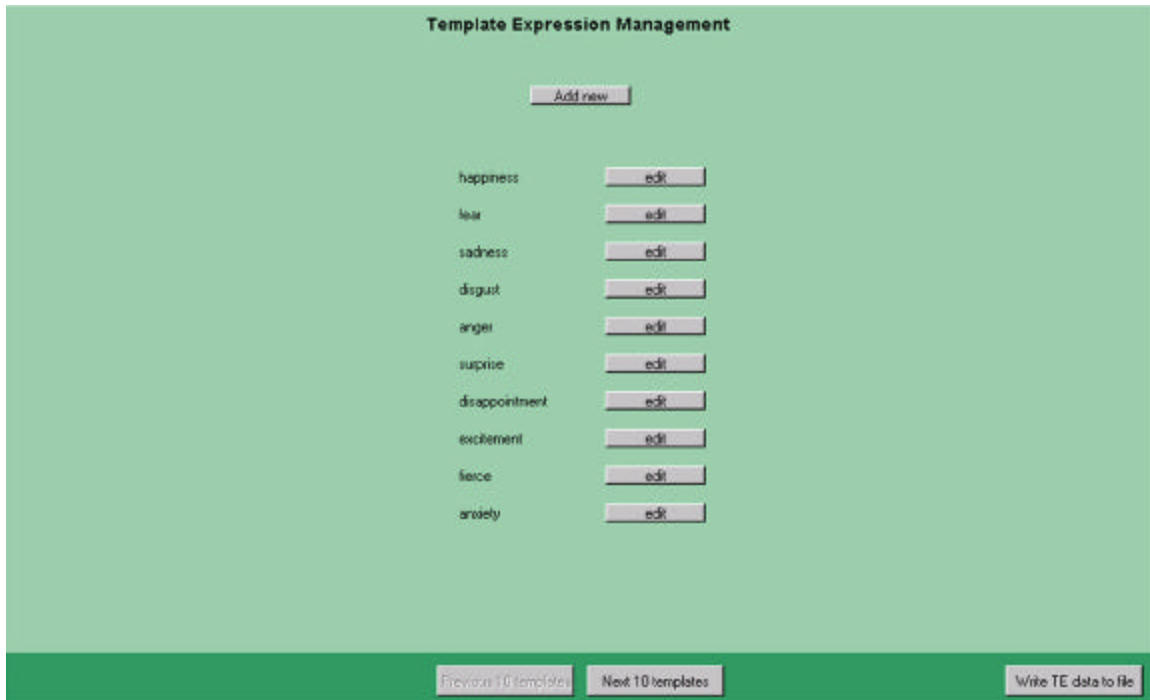


Figure 5.6: FED entry management main screen

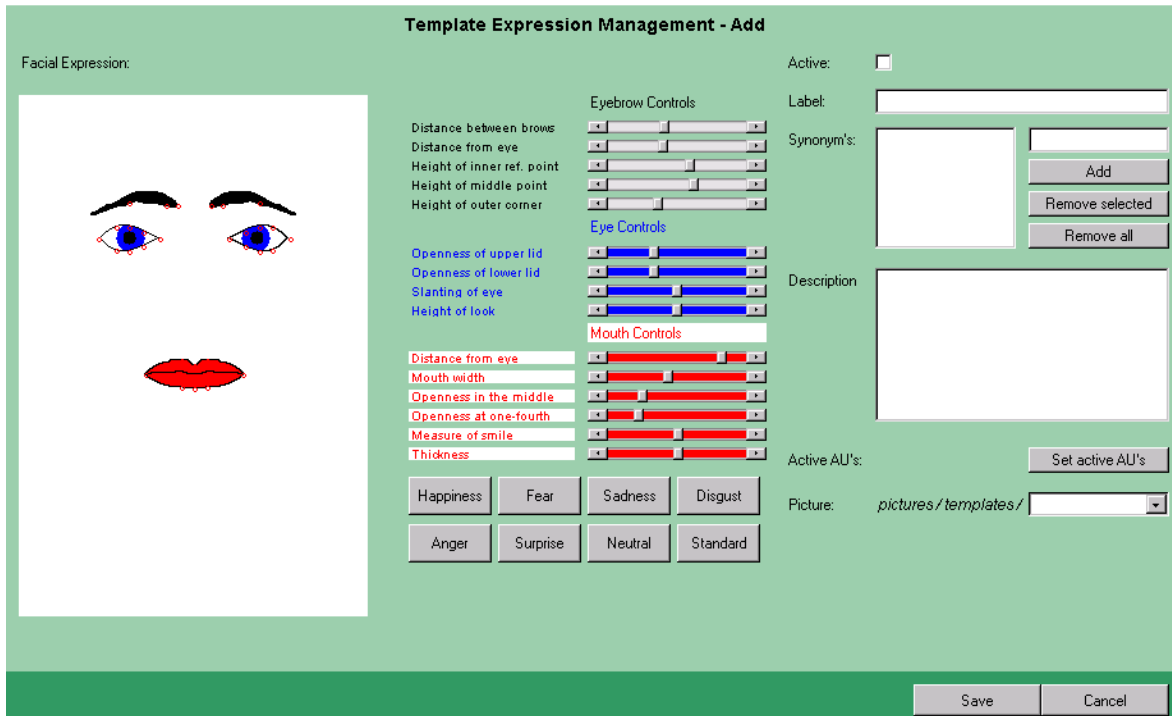


Figure 5.7: Adding a new FED entry

**Figure 5.8: Editing an existing FED entry**

checkbox indicates if the entry is to be used when processing a query issued from the FED main website. Figure 5.8 shows a screenshot of the screen used to edit a FED entry, which appears if the user selects to edit a FED entry from the main screen. Here the facial expression and/or the interpretation data of the FED entry can be modified and saved to the FED database by pressing 'save changes'. The FED entry can be deleted by pressing 'delete' or the changes made can be cancelled by pressing 'cancel'. The user can scroll to the next or previous FED entry with the 'next>>' and '<<previous' buttons.

### 5.3.1.2 Adding entries through the AddTemplateTool

As mentioned earlier, filling the FED database with entries can be time consuming. For this reason, a tool was developed that allows outside users to add new entries to FED: the AddTemplateTool. This tool is available as a website and allows users to sketch a facial expression with FaceShop, enter the facial expression label, and save the information to the FED database.

In the FED entry management screen, the entered information appears as a new inactive entry, with just the facial expression and label properties set. An authenticated user can select the best of these new entries to be included into the FED system. This is accomplished by adding additional information such as the label synonyms, description, active AU's and example picture and by setting the 'active' property of the FED entry to *true*. Figure 5.9 shows a screenshot of the AddTemplateTool.

The AddTemplateTool was but to practice by a number of students participating in a course by Leon Rothkrantz of the Knowledge Based Systems Group. They were given the assignment to model 10 facial expressions and save them to the FED database. This is why temporarily the field 'Group' was included in the AddTemplateTool as well.

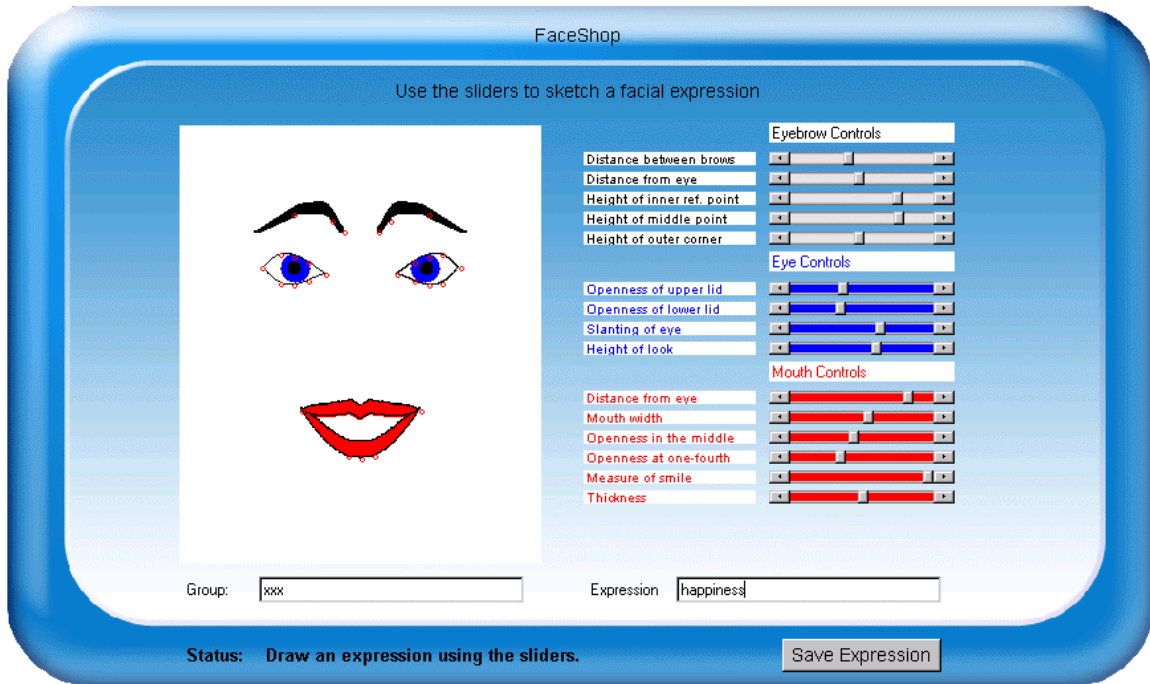


Figure 5.9: AddTemplateTool

## 5.4 User management

The administrative backend website of FED is only accessible for authenticated users. When a user wants to gain access to the admin site, he has to supply a valid login name and password. At the admin site, it is possible to add, edit or delete the authenticated users. The GUI of this admin function resembles the GUI of the FED entry management admin function closely. Also present are a main screen displaying all the users, from where the authenticated user can select to add a new user or edit an existing user, and screens for adding and editing users. Figure 5.10, 5.11 and 5.12 show screenshots of these screens.

## 5.5 Logging

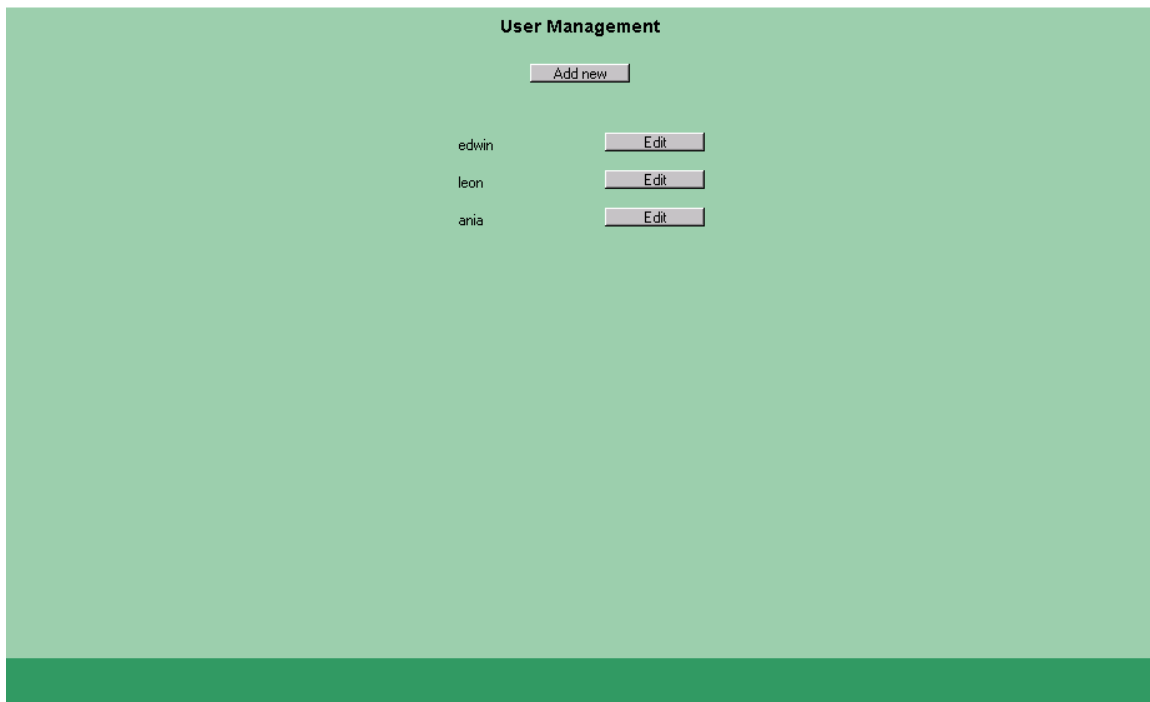
The main functionality provided by FED is the possibility to issue a query into the facial expression dictionary. It may be of interest to see how many queries were issued in a certain time period, and how many of these were actually successful. An authenticated user has the possibility to view a log of the queries issued from the main website at the administrative backend website.

Besides this *query log*, authenticated users can also view the *admin log*. This log contains records of all the actions taken by authenticated users at the admin site. The next two subsections describe these two logs in more detail.

### 5.5.1 Query log

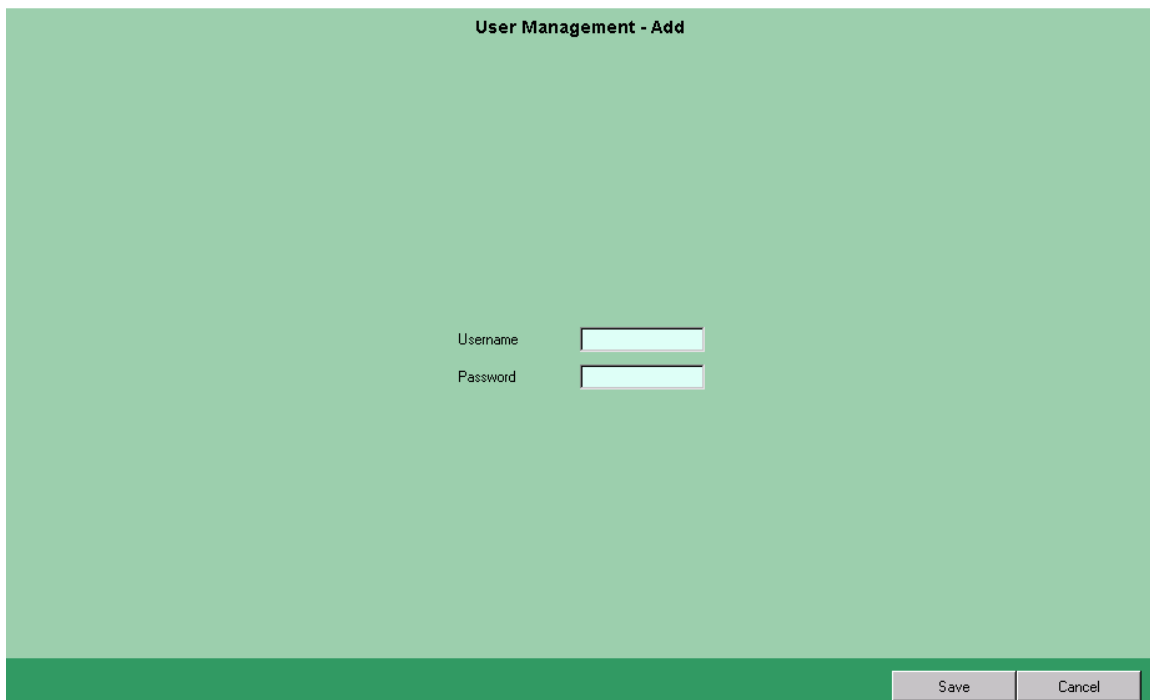
The query log shows the query statistics of five of the six possible queries into FED. The incremental query is not included, because of the fact that with this query, there are no relevant statistics regarding success or failure.





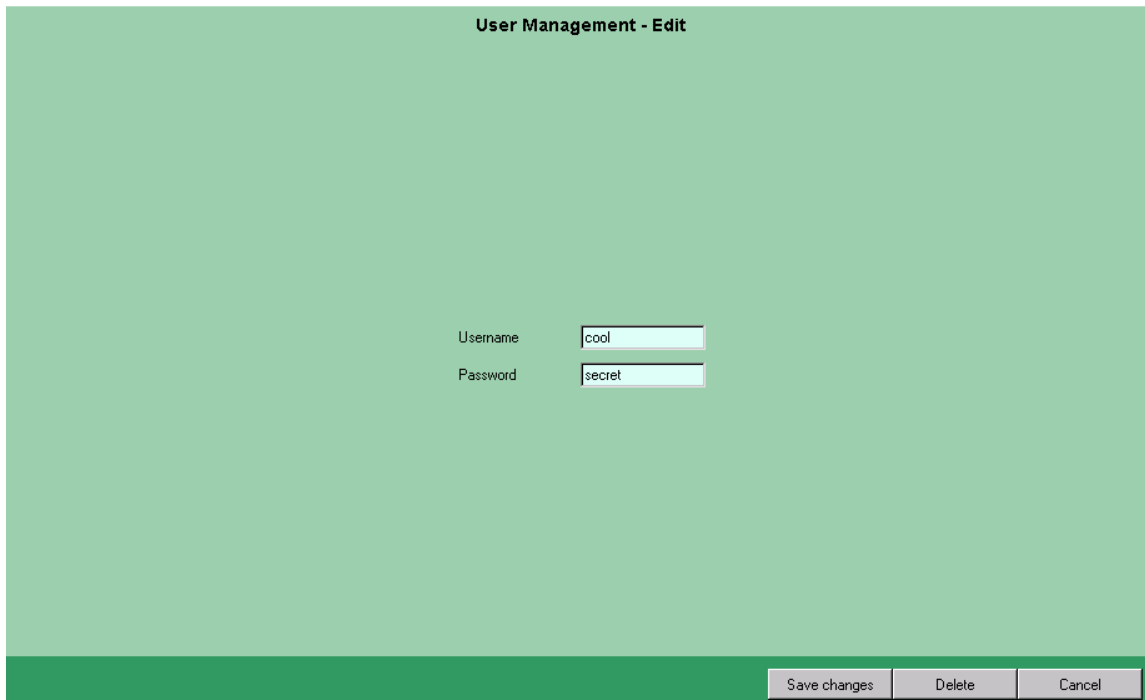
**Figure 5.10: User management main screen**

---



**Figure 5.11: Adding a new FED admin user**

---



User Management - Edit

Username

Password

Save changes Delete Cancel

**Figure 5.12: Editing an existing FED admin user**

Each time a query is issued from the FED main website, a log entry is added to *tbl\_querylog* of the FED database. When an authenticated user selects to view the query log at the admin site there are four statistics displayed per query type: the number of issued queries, the number of successful queries, the hitrate ( $\# \text{ successful queries} / \# \text{ queries}$ ) and the average number of results. A query is considered successful if there was at least one result returned by the query. Figure 5.13 shows a screenshot of the query log. The toolbar at the bottom enables users to view the query log over any possible time period.

## 5.5.2 Admin Log

There are essentially four possible actions that can be performed by authenticated users at the admin site. There exists the possibility to manage FED entries, manage FED admin users and view the query and admin logs.

Each time a user adds, edits or deletes a FED entry or adds, edits or deletes a FED admin user, a log entry is added to the *tbl\_adminlog* of the FED database. Also logged is when a user logs in to the FED admin site. When users view the query or admin log, this is not logged because of the fact that this has no effect on the operation of the FED system.

When an authenticated user selects to view the admin log, all log entries are read from *tbl\_adminlog* of the FED database and displayed to the user. This includes the time the log entry was made, the FED admin user that performed the action, if applicable, the Java servlet that performed the action and a short description of the action. As with the query log, it is possible to view the admin log of any given time period by using the controls at the bottom of the screen. Figure 5.14 shows a screenshot of the admin log.

Query Log				
Query type	# Queries	# Succesfull Queries	Hitrates	Average # results
ACTIONUNITS	72	64	0.88	8.6
FACESHOP	244	164	0.67	0.9
GEOMETRY	181	119	0.65	11.5
KEYWORD	383	260	0.67	1.1
PICTUREUPLOAD	60	29	0.48	1.1

Displaying log from 1 1 2002 to 8 5 2002 Display Today Last week Last month Total

Figure 5.13: The query log

Admin Log			
Date / Time	User	Servlet	Event
02-05-2002 13:47:38		AdminAddTEServlet	Added TE MMVR01-SICK
02-05-2002 13:56:07		AdminAddTEServlet	Added TE MMVR01-YAWNING
02-05-2002 14:05:03		AdminAddTEServlet	Added TE MMVR01-DISAPPOINTED
02-05-2002 14:39:27		AdminAddTEServlet	Added TE MMVR01-PANIC
02-05-2002 16:32:14	edwin	AdminLoginServlet	User EDWIN logged in
02-05-2002 20:33:53		AdminAddTEServlet	Added TE 1014692-AFGRIJZEN
03-05-2002 10:27:00	edwin	AdminLoginServlet	User EDWIN logged in
03-05-2002 11:45:10		AdminAddTEServlet	Added TE IN4017TU_GROEP11-ALERT
03-05-2002 11:46:27		AdminAddTEServlet	Added TE IN4017TU_GROEP11-VERLEGEN
03-05-2002 11:47:17		AdminAddTEServlet	Added TE IN4017TU_GROEP11-MISSELUK
03-05-2002 11:47:56		AdminAddTEServlet	Added TE IN4017TU_GROEP11-SERIEEN
03-05-2002 11:48:30		AdminAddTEServlet	Added TE IN4017TU_GROEP11-VERWACHTINGSVOL
03-05-2002 12:36:05		AdminAddTEServlet	Added TE IN4017TU_GROEP11-GEINTERESSEERD
03-05-2002 12:36:48		AdminAddTEServlet	Added TE IN4017TU_GROEP11-AFGRIJZEN
03-05-2002 12:37:29		AdminAddTEServlet	Added TE IN4017TU_GROEP11-NEERSLACHTIG
03-05-2002 12:38:20		AdminAddTEServlet	Added TE IN4017TU_GROEP11-VENJUNG
03-05-2002 13:41:28		AdminAddTEServlet	Added TE IN4017TU_GROEP11-BEWONDERING
03-05-2002 17:44:08	edwin	AdminLoginServlet	User EDWIN logged in
05-05-2002 17:53:53		AdminAddTEServlet	Added TE MMVR01-SELF CONFIDENT
05-05-2002 17:54:11		AdminAddTEServlet	Added TE MMVR01-SELF CONFIDENT
05-05-2002 17:54:30		AdminAddTEServlet	Added TE MMVR01-SELF-CONFIDENT
05-05-2002 17:55:15		AdminAddTEServlet	Added TE MMVR01-SELF-CONFIDENT
05-05-2002 18:05:43		AdminAddTEServlet	Added TE MMVR01-MELANCHOLIC
06-05-2002 11:05:37	edwin	AdminLoginServlet	User EDWIN logged in
07-05-2002 11:00:20	edwin	AdminLoginServlet	User EDWIN logged in
08-05-2002 09:02:22	edwin	AdminLoginServlet	User EDWIN logged in
08-05-2002 13:24:11	edwin	AdminManageUsersServlet	Added user COOL

Displaying log from 1 1 2002 to 8 5 2002 Display Today Last week Last month Total

Figure 5.14: The admin log

## Chapter 6: FED Query Implementation

*This chapter describes the implementation details of the various query possibilities into FED. Section 6.1 describes the implementation details of the Visual Data Inspection Tool and how the most discriminative facial features of entries in FED were determined. Sections 6.2 through 6.7 describe the implementations of the Label, Action Unit, Geometry, Incremental, Picture and FaceShop query respectively.*

### 6.1 Visual Data Inspection: determining the most discriminative facial features

The implementations of four of the six possible queries into FED are based on the evaluation of certain facial features. The Geometry, Incremental, FaceShop and Picture Query all process a query by comparing the values of certain facial features of the facial expressions present in the FED database with the entered query (Geometry Query), each other (Incremental Query), the facial features of the sketched facial expression (FaceShop Query) or the facial features present in a picture of a facial expression (Picture Query).

In order to determine which facial features discriminate best, a *Visual Data Inspection Tool (VDI Tool)* was developed as part of the FED administrative backend website. With this tool, an authenticated user has the possibility to define two facial features as a mathematical expression and view a scatter plot of the values of these features for all active facial expressions present in the FED database. Figure 6.1 shows a screenshot of the VDI tool feature definition screen. It is also possible to use one of the predefined features, instead of creating one through a mathematical expression.

To illustrate how the most discriminative facial features were found, consider the scatter plots of figure 6.2 and 6.3. Figure 6.2 shows the resulting scatter plot of the features ‘distance between the eyebrows’ and ‘mouth distance to eyes’. All points are located roughly in the same area, and several facial expressions have exactly the same distance between the eyebrows and distance from the mouth to the eyes. The selected point even represents 5 facial expressions with identical feature values. Clearly, these features do not appear to discriminate facial expressions very well.

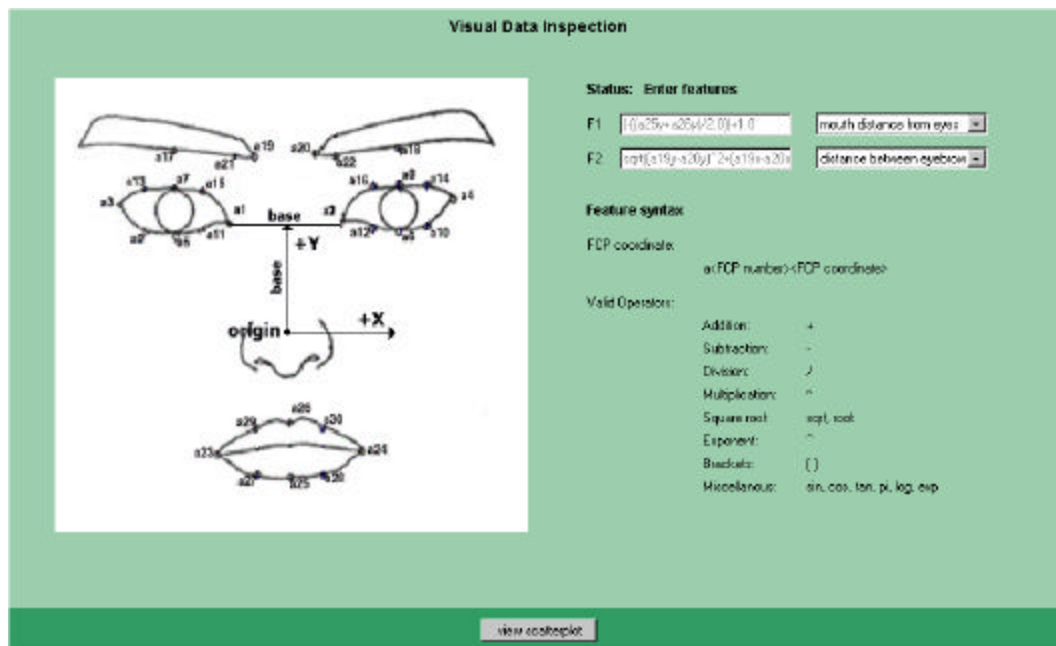
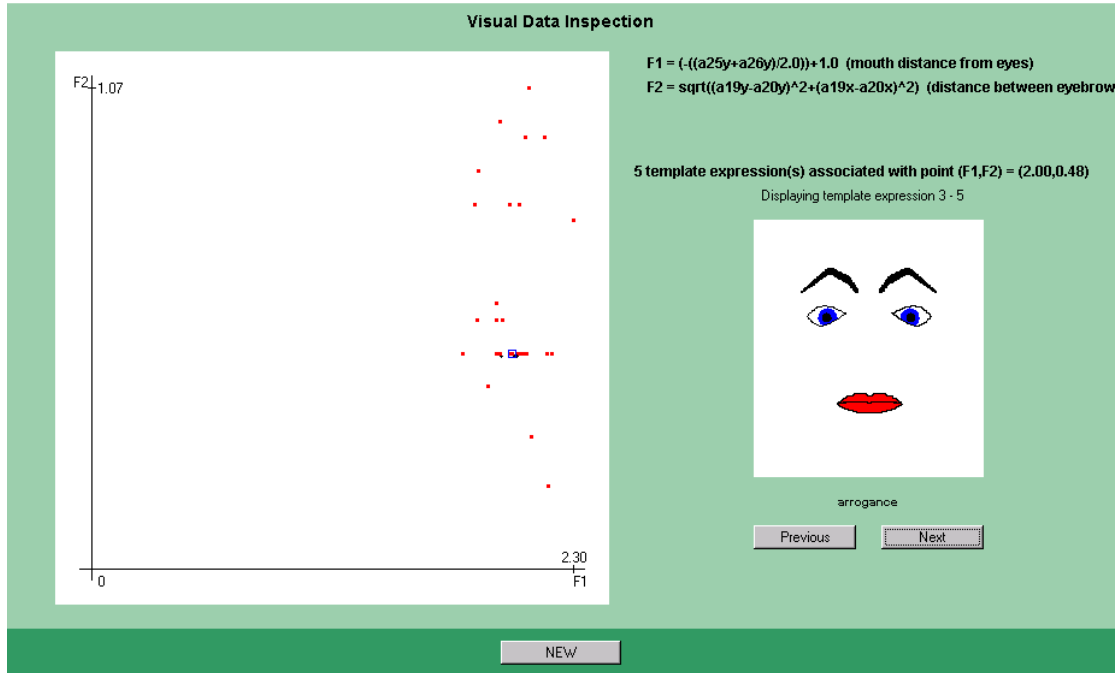
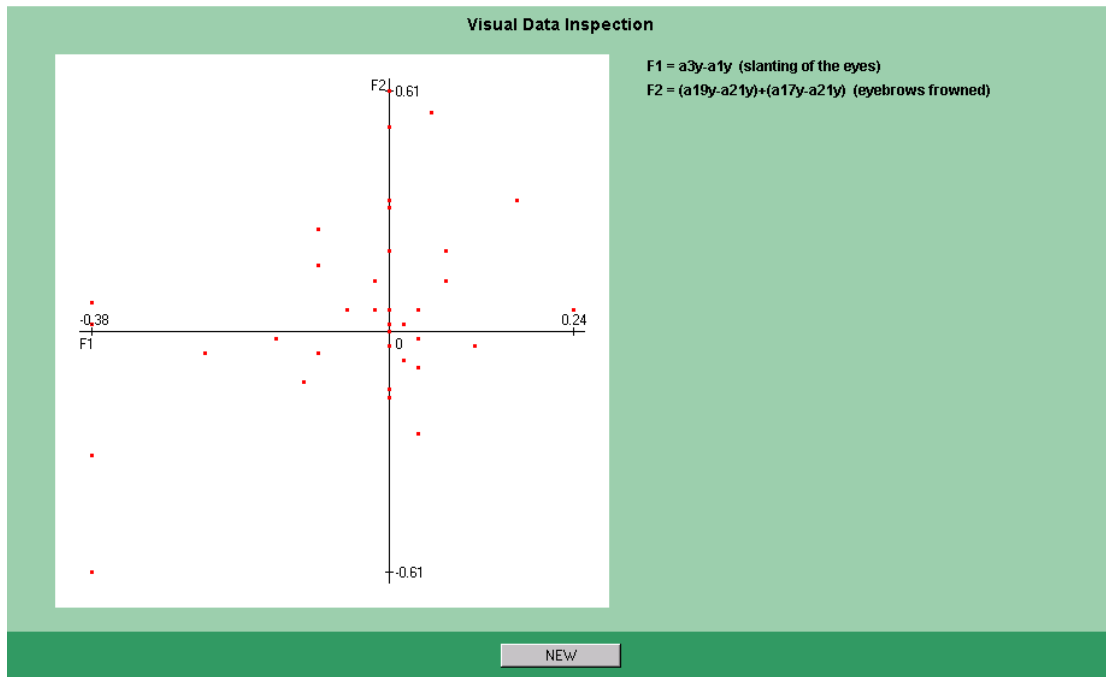


Figure 6.1: VDI Feature definition



**Figure 6.2: Scatter plot of two features that discriminate between FED entries badly**

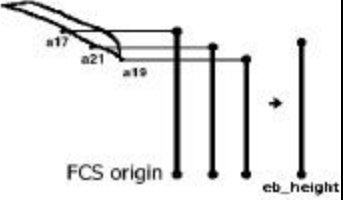
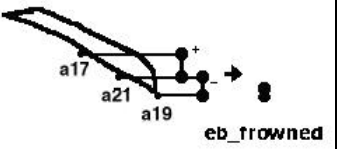

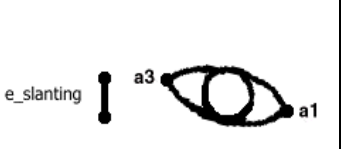
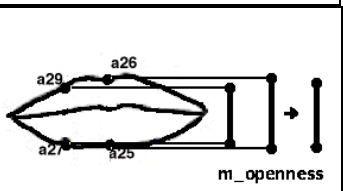
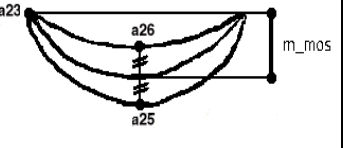


**Figure 6.3: Scatter plot of two features that discriminate between FED entries well**

Figure 6.3 shows the scatter plot of the features ‘slanting of the eyes’ and ‘eyebrows frowned’. In this case, all points in the scatter plot are spread fairly equal, and every point corresponds to exactly one facial expression. These features do seem to discriminate adequately between the facial expressions in the FED database. In an explorative way, the facial features of table 6.1 were

found to be promising candidates for the most discriminative facial features of all active facial expressions entries in the FED database. In principle, statistical analysis should be employed to verify if features that appear to discriminate badly (such as the features of figure 6.2) in fact do so, and that the features of table 6.1 are in effect discriminating well. But unfortunately, because of the limited number of available data points, methods such as Principal Component Analysis will not return useful results at this moment. Because of the symmetry of the FaceShop facial expressions, only one side of the facial expression is used to calculate feature values.

**Table 6.1**  
**Features used to determine the likeness of two facial expressions**

Feature	Numerical	Graphical
<i>eb_height</i> – Height of the eyebrows	$(a17.y + a19.y + a21.y) / 3.0$	
<i>eb_frowned</i> – The extent to which the eyebrows are frowned	$(a19.y - a21.y) + (a17.y - a21.y)$	
<i>e_openness</i> – Openness of the eyes	$a7.y - a5.y$	
<i>e_slanting</i> – Slanting of the eyes	$a3.y - a1.y$	
<i>m_openness</i> – Openness of the mouth	$((- (a25.y - a26.y) - (a27.y - a29.y)) / 2.0)$	
<i>m_mos</i> – Measure of smile	$a23.y - (a26.y + ((a25.y - a26.y) / 2.0))$	

## 6.2 Label Query

Figure 6.4 shows a screenshot of the KWQApplet, which contains the GUI for issuing a Label Query. The user is given some examples of valid queries. A status field informs the user to issue a query in the 'Keyword' textfield. If the user selects to process the entered query, it is sent to the QPKeywordServlet of the communication layer. The QPKeywordServlet invokes the Label QPM, which tries to match the entered query to a facial expression label, label synonyms or words in the description of all FED entries. If one or more matches are found, the corresponding FED entries are sent back to the KWQApplet and displayed to the user (See figure 6.5). If no matches are found but the query matches an entry when allowing for one insertion, deletion or substitution, the relevant keyword is sent back to the KWQApplet as a spelling suggestion.

## 6.3 Action Unit Query

The AUQApplet contains the GUI for issuing an Action Unit Query. The AUQApplet contains an enumeration of 30 AU's and corresponding checkboxes. The user can issue a query by selecting the AU's and press 'process query'. Figure 6.6 shows a screenshot of the AUQApplet. The selected AU's are sent to the QPActiveAUServlet, which in turn will invoke the AU QPM. There the selected AU's are matched against the active AU's of all FED entries. All matches are ordered by relevance: entries with the highest number of AU's in common with the query are presented to the user first. Matches with the same number of AU's in common are separated by the number of AU's they do not have in common with the query. The results are sent back to the AUQApplet and presented to the user in the same manner as with the KWQApplet (figure 6.7).



**Figure 6.4: Screenshot of the KWQApplet, used to issue a Label Query**



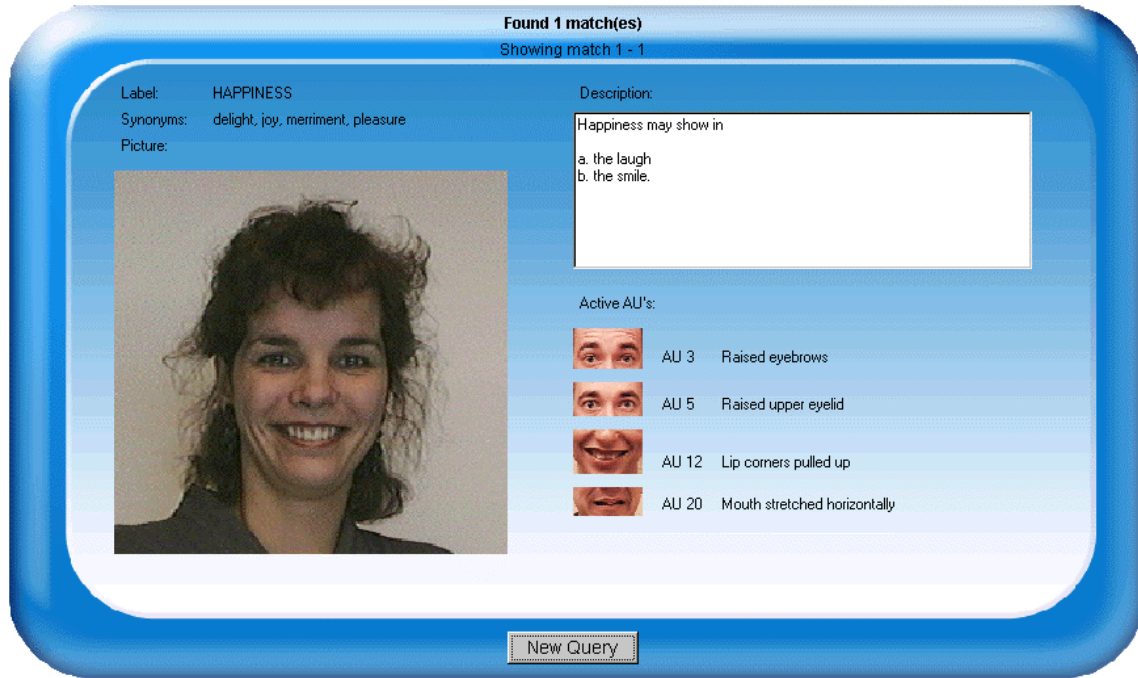


Figure 6.5: Label Query results displayed to the user in the KWQApplet



Figure 6.6: screenshot of the AUQApplet, used to issue a Action Unit Query





Figure 6.7: Action Unit Query results displayed to the user in the AUQApplet

#### 6.4 Geometry Query

The GEOMQApplet provides users with the possibility to issue a query for facial expressions with certain geometrical features present (figure 6.8). The next two subsections describe how the presence of a certain geometrical feature is detected and how a Geometry Query is processed.




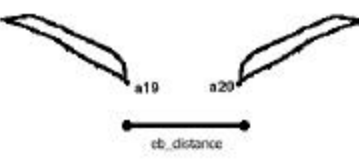
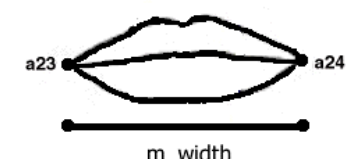
Figure 6.8: screenshot of the GEOMQApplet, used to issue a Geometry Query

### 6.4.1 Geometrical feature detection

Table 6.3 shows all possible Basic Geometry Queries and the metrics that are used to determine if a certain geometrical feature is present. The variables in the table (i.e. 'eb\_frowned') are the metrics from table 6.1 and the additional metrics of table 6.2. Because of the fact that each facial expression entry in FED was created with the FaceShop tool, the left and right parts of a facial expression are always symmetrical. This means that when determining a certain facial feature, only the FCP-coordinates of one of the sides of the facial expression have to be examined; there would be no added value in calculating the feature value using the complete facial expression. Also, the x-coordinate of the eyebrows and the eyes are identical, so calculating the distance between two symmetrical points is equivalent to calculating the distance between the x-coordinates.

As mentioned in section 4.5.3, it is also possible to issue an aggregate query using the logical AND ('&') and OR ('+') operators. This is accomplished by applying the logical operators on sets of facial expressions that are the result of a basic geometry query.

**Table 6.2**  
Additional features used to determine the presence of a geometrical feature

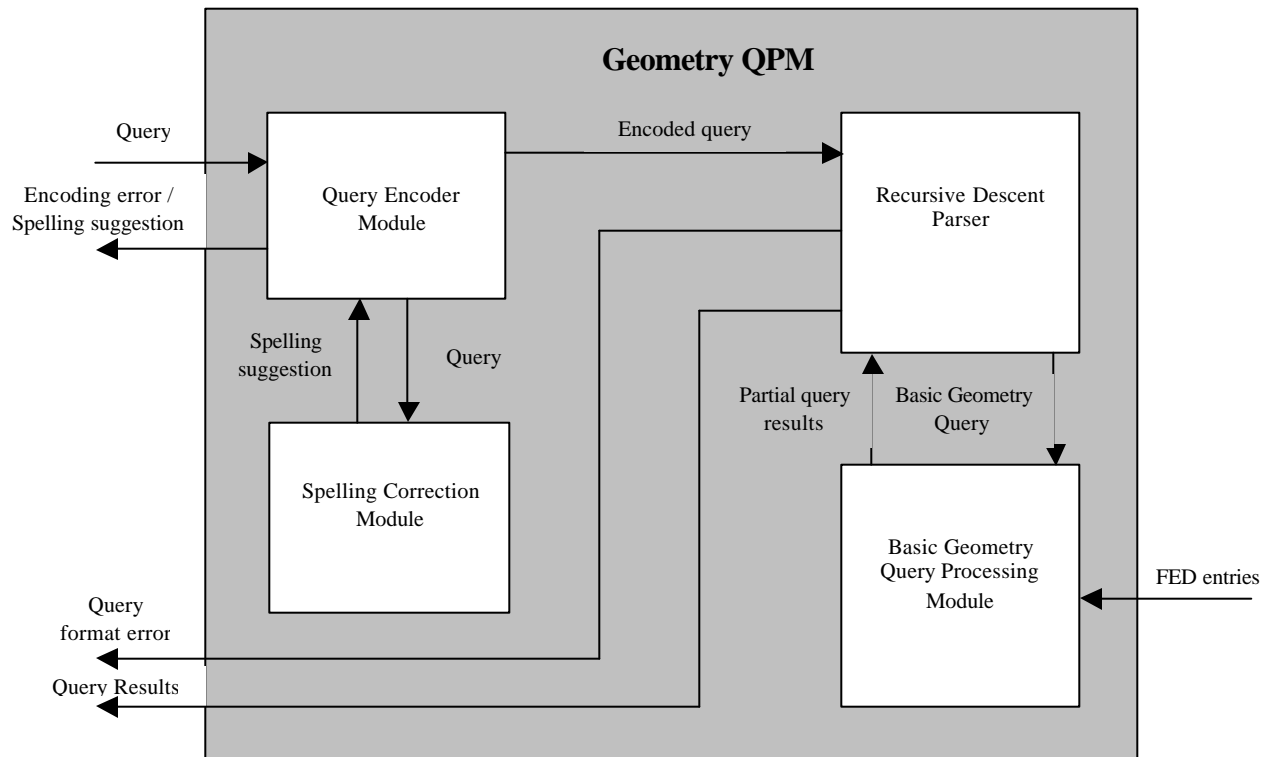
Feature	Numerical	Graphical
<i>eb_slanting</i> – Slanting of the eyebrows	$(a17.y + a21.y) + (a21.y - 19.y)$	
<i>eb_distance</i> – Distance between the eyebrows	$a20.x - a19.x$	
<i>m_width</i> – Width of the mouth	$a24.x - a23.x$	

### 6.4.2 Geometry QPM Implementation

Figure 6.9 shows the design of the Geometry QPM (see also figure 4.8). When a user has entered a query at the GEOMQApplet and selects to process, the entered query is sent to the QPGeometryServlet of the communication layer. Here the query is received and passed along to the Geometry QPM. This subsection will describe how the Geometry QPM processes a query by describing each step in detail.

**Table 6.3**  
Basic geometry queries and corresponding criteria used to determine if the feature is present

Facial Feature	Possible query	Criterion
Eyebrows	raised	$(eb\_slanting + (a19.y - 1.5)) > 0.0$
	frowned	$eb\_slanting < 0.0$
	close together	$eb\_distance < 0.4$
	far apart	$eb\_distance > 0.8$
Eyes	wide open	$e\_openness \geq 0.5$
	open	$e\_openness \geq 0.3$
	slit	$0.1 \geq e\_openness < 0.3$
	closed	$e\_openness < 0.1$
	slanting up	$e\_slanting \geq 0.15$
	slanting down	$e\_slanting \leq -0.15$
Mouth	wide open	$m\_openness \geq 0.75$
	open	$m\_openness \geq 0.4$
	smiling	$m\_mos \geq 0.15$
	not smiling	$m\_mos \leq -0.15$
	stretched horizontally	$m\_width > 1.7$
	stretched vertically	$m\_openness \geq 0.8$



**Figure 6.9: Geometry QPM design**

### 6.4.2.1 Encoding the query

The first step in the processing of a Geometry Query is the encoding of the entered query. This means that each Basic Geometry Query present in the entered query is replaced by a unique integer. This integer is used to represent the Basic Geometry Query throughout the code of the Geometry QPM. This is done to reduce the complexity of the code and make the parsing of the query easier. A dictionary is used to store the mapping between Basic Geometry Queries and their corresponding integers. Table 6.4 shows a small part of the dictionary. As becomes clear from table 6.4, the dictionary is able to map a Basic Geometry Query to its corresponding integer in a number of ways. This increases the robustness of the geometry query. As an example of how an entered query is encoded, consider the query from the previous subsection: ‘(mouth open & eyes slit) + eyebrows raised’. The query encoder extracts each basic geometry query present in the query, and tries to map it to its corresponding integer using the dictionary. In this case, the encoding will be successful: each basic geometry query is present in the dictionary and there are no brackets present at invalid positions. The result of the encoding process will be a new string: ‘(1&9)+13’. All redundant white spaces are removed as well.

**Table 6.4**  
**Subsection of the dictionary used to encode a geometry query**

Basic Geometry Query	Integer
mouth wide open	0
mouth open wide	0
mouth opened wide	0
mouth open	1
open mouth	1
opened mouth	1
mouth opened	1
...	...

### 6.4.2.2 Spelling Correction

As with the Label query, a *spelling correction module* is used to correct small spelling errors during the encoding step. If a basic geometry query is not present in the dictionary, the spelling correction module is used to check if the basic geometry query is present in the dictionary when allowing for one insertion, deletion or substitution.

If for example the query that was entered looked like ‘(mout open & eyes slit) + eyebrows raised’, the spelling correction module will find that when allowing for one deletion in the basic geometry query ‘mout open’, it would be present in the dictionary as ‘mouth open’. A string ‘(mouth open & eyes slit) + eyebrows raised’ will be generated, send to the GEOMQApplet and displayed to the user as a spelling suggestion.

### 6.4.2.3 Parsing the query

If the encoding of the entered query was successful, the encoded query is send to the *Recursive Descent Parser Module* of the geometry QPM. This parser module is able to process strings that confirm to a grammar compliant with all possible formats of an encoded geometry query:

EXPRESSION := TERM | TERM ( '+' | '&' ) TERM

TERM := NUMBER | '(' EXPRESSION ')'

NUMBER := 0 | 1 | 2 | ..... | N

The parser will scan the string character for character and execute the basic geometry queries represented by an integer one at a time. The result of executing a basic geometry query is a set of FED entries. The AND ('&') and OR ('+') operators are applied on two sets of FED entries and result in a new FED entry set. While scanning the query, the parser checks for query format errors. This means that if for example somewhere in the encoded query the string '++' occurs, an error is generated, sent to the GEOMQApplet and displayed to the user.

#### 6.4.2.4 Executing basic geometry queries

The *Basic Geometry Query Processing Module* takes an integer as argument and returns a set of FED entries that exhibit the geometrical feature as indicated by the integer as result. Determining if a geometrical feature is present is accomplished as described in section 4.5.3.1.

#### 6.4.2.5 Displaying the results

If encoding, parsing and execution of the entered geometry query was successful, the results are sent back to the GEOMQApplet and displayed to the user as shown in figure 6.10. Analogous to the Action Unit query, it is possible to browse through the results using the 'previous' and 'next' buttons.



Figure 6.10: Geometry Query results displayed to the user in the GEOMQApplet

## 6.5 Incremental Query

If a user has no information on the label, active action unit's or geometrical features of the facial expression he is looking for, he can try to find it through issuing an Incremental Query. With this query, the user incrementally selects a facial expression that resembles the facial expression he is looking for the closest.

When the user starts the incremental query, the INCQApplet is loaded from the server. Figure 6.11 shows the initial state of the INCQApplet. It displays the four pivot facial expressions to the user. The user can start to search for a certain facial expression by clicking on the picture or on the corresponding 'more' button of the facial expression that resembles the facial expression he is looking for the closest. The number on the 'more' button indicates the number of facial expressions that can be found in that search direction. If the user immediately sees the facial expression he was looking for, he can click on the corresponding 'details' button to view the complete FED entry.

Figure 6.12 shows the INCQApplet after the user has selected the facial expression at the top left. The 'more' button indicated that 4 facial expressions could be found at this search level. This can be verified as follows: there are 3 visible plus  $0 + 0 + 0 + 1$  (numbers on the 'more' buttons of the pivot expressions) = 4 facial expressions at this level in total. The user can go back to the previous search level by clicking the 'previous' button. Figure 6.13 shows the INCQApplet after the user subsequently selects the facial expression at the bottom right. Again, the number of facial expressions indicated by the 'more' button complies with the total number of facial expressions that can be reached from this level. Finally, figure 6.14 shows the INCQApplet when the user subsequently selects to view the details of the expression at the bottom right.

The next subsections describe how the extent to which two facial expressions resemble each other is determined, and how an Incremental Query subspace (the four pivot facial expressions and corresponding clusters) are determined.

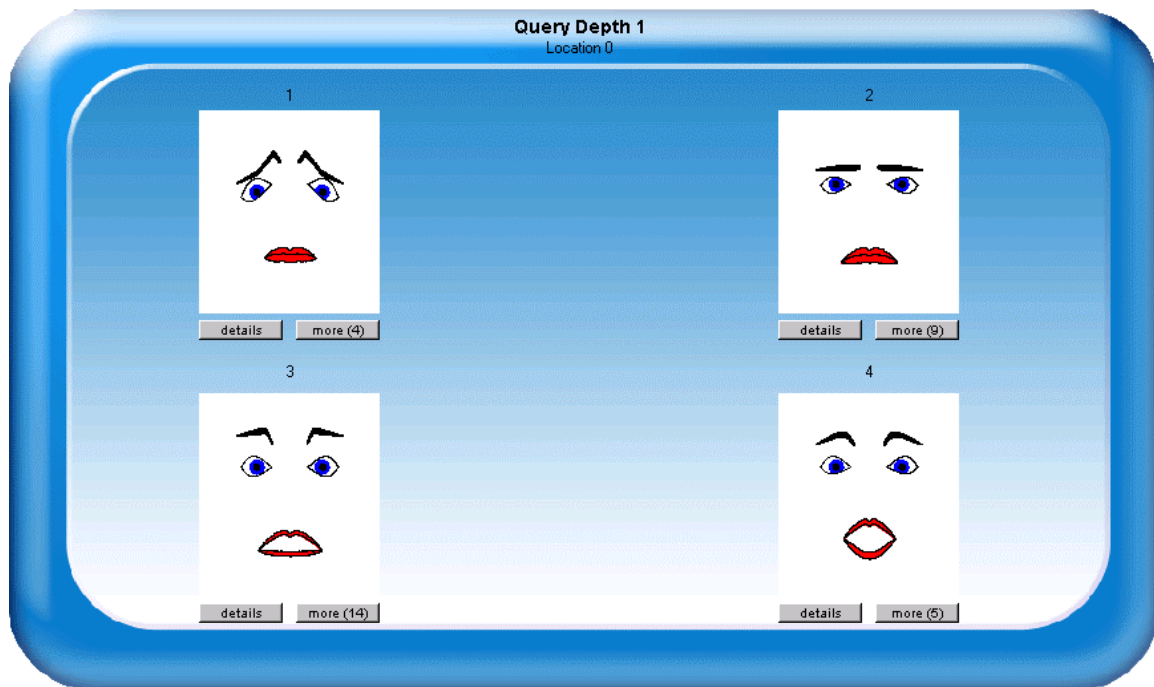
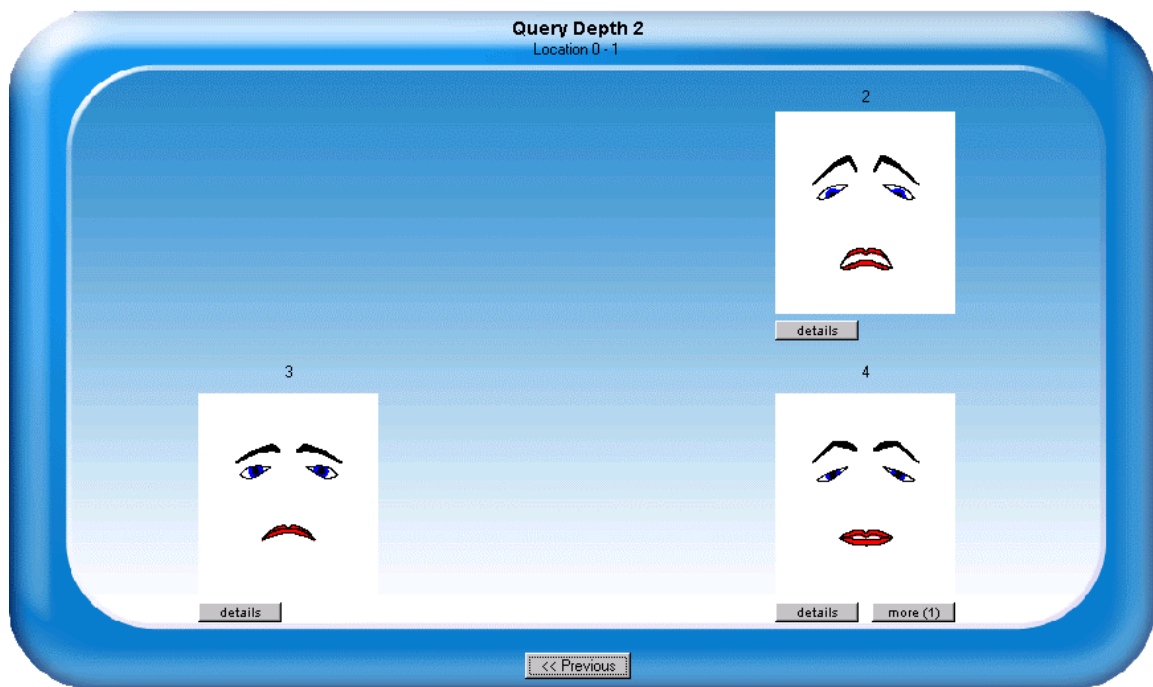
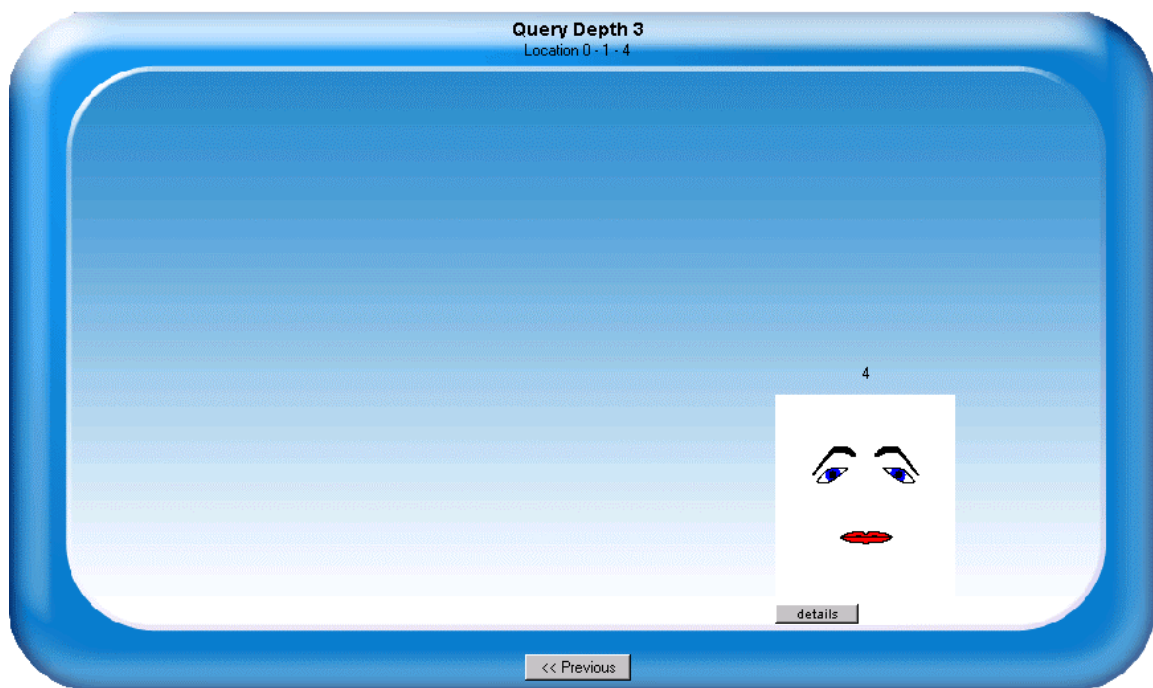


Figure 6.11: initial state of the INCQApplet





**Figure 6.12:** state of the INCQApplet after selection of the top left facial expression of figure 6.11



**Figure 6.13:** state of the INCQApplet after selection of the bottom right facial expression of figure 6.12



**Figure 6.14: the complete FED entry belonging to the bottom right facial expression of figure 6.13**

### 6.5.1 Determining the likeness of two facial expressions

The coordinates of the FCP's of facial expressions entries are used to determine the extent to which two facial expressions resemble each other. The FCP coordinates are already normalized in the face model of Kobayashi and Hara (see section 2.4.2), so as a first approach, a Euclidian distance metric could be used to determine a measure of likeness of two facial expressions A and B:

$$similarity(A, B) = \frac{\| \vec{A} - \vec{B} \|}{\| \vec{A} \| \| \vec{B} \|}$$

Certain facial features have a different impact on the interpretation of the facial expression. For example, a small difference in the openness of the eyes generally has a smaller effect on the interpretation of the displayed facial expression than a small difference in the extent to which the mouth corners are curled up- or downwards. For this reason, a second approach could include weight factors for each of the 30 FCP's:

$$similarity(A, B) = \frac{\sum_{i=1}^{30} w_i (x_{Ai} - x_{Bi})^2}{\| \vec{A} \| \| \vec{B} \|}$$

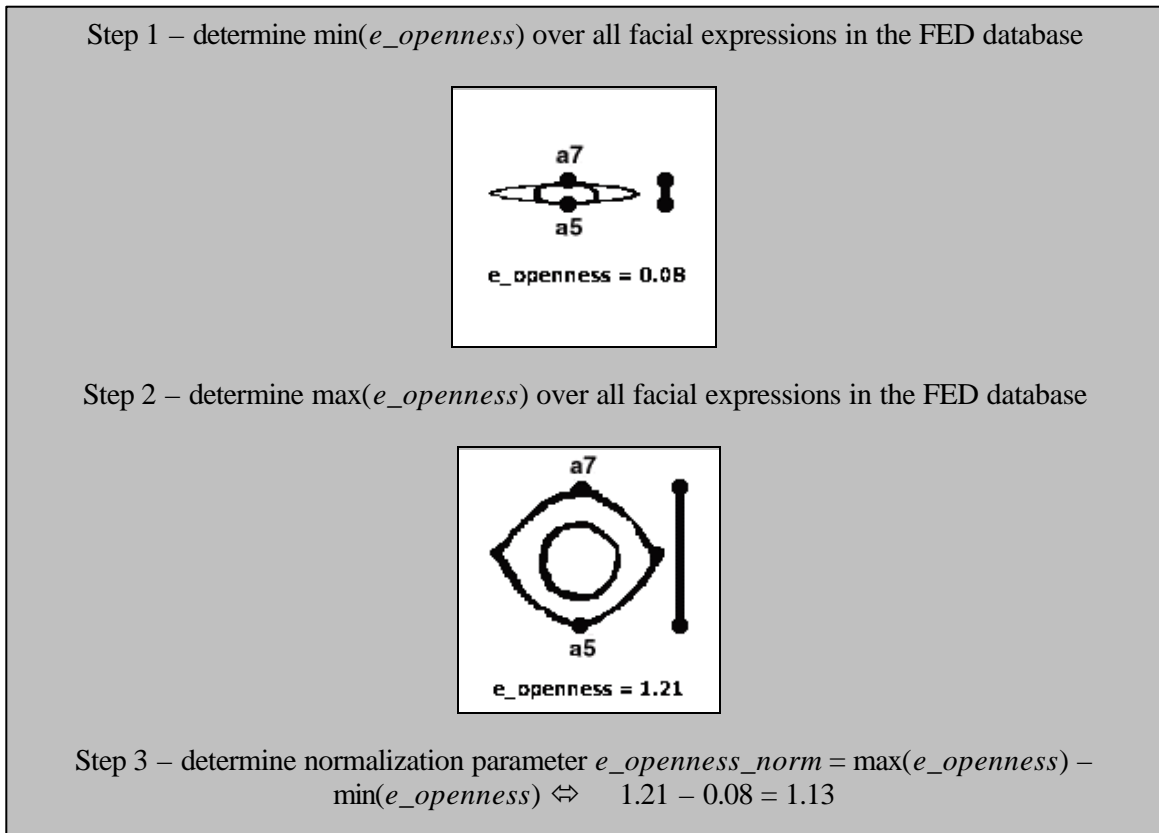


For the Incremental Query however, the choice was made to reduce the feature space from 30 to 6 dimensions using the 6 facial features that appeared to discriminate best between all active facial expressions in FED (see section 6.1, table 6.1):

Before these features can be used to determine the likeness between facial expressions, they have to be normalized. If absolute values are used, certain differences in features may have an inappropriately large influence: the value of the feature *eb\_height* is of a different order than the value of the feature *e\_slanting*.

In order to normalize a feature value, it is necessary to determine the minimum and maximum value this feature can have. This is accomplished by determining the facial expression with the smallest feature value and the facial expression with the largest feature value. The absolute distance between these two feature values is used to normalize the values of the feature of all other facial expressions.

Certain facial expressions sketched with FaceShop were based on an available picture of a person displaying the facial expression. People can have different anthropomorphic facial characteristics; it is possible that another person displaying the same facial expression has the physical ability to open his mouth much farther. It would therefore be better to determine the minimum and maximum facial feature value for each individual, but with the Incremental Query, the group norm is used as an individual norm. As an example, consider how the normalization parameter for the feature *e\_openness* is determined:



#### Algorithm 6.1: determining the feature normalization parameters

For all facial expressions that are being compared, the absolute value of the feature *e\_openness* is divided by *e\_openness\_norm*. This ensures that the value of *e\_openness* always lies between 0 and 1. This normalization is applied for each feature of the facial expression. This makes it

possible to determine the extent to which two facial expressions  $exp1$  and  $exp2$  resemble each other. This is accomplished by calculating the total normalized distance as follows (using the normalized feature values of the two facial expressions  $exp1$  and  $exp2$ ):

$$D = \text{total normalized distance between } exp1 \text{ and } exp2 = \\ (eb\_height1 - eb\_height2) + (eb\_frowned1 - eb\_frowned2) + (e\_openness1 - e\_openness2) \\ + (e\_slanting1 - e\_slanting2) + (m\_openness1 - m\_openness2) + (m\_mos1 - m\_mos2)$$

If  $D$  is equal or close to zero, it is likely that the two expressions  $exp1$  and  $exp2$  are very similar. If  $D$  is negative,  $exp1$  is said to be ‘smaller’ than  $exp2$ . Alternatively,  $exp1$  is said to be ‘bigger’ than  $exp2$  if  $D$  is positive. In general, a facial expression is said to be smaller than another facial expression on all likeness features, if the eyebrows are located lower and slanting more upwards, the eyes are less open and slanting more downwards and the mouth is less open and displaying a smaller measure of smile. Alternatively, a facial expression is said to be bigger on all likeness features if the opposite is true.

### 6.5.2 Determining the minimum and maximum facial expression of a set

Using the normalized distance  $D$ , it is possible to determine the *minimum* and *maximum* facial expression of a set of facial expressions. The minimum expression is determined by applying the following algorithm:

**Initialization:**

- Let  $Q$  be the set of facial expressions from which the minimum facial expression has to be determined.
- Let variable  $min$  contain the first expression of the set  $Q$  (the currently found minimum expression)
- Let variable  $cur$  contain the second expression of the set  $Q$  (the facial expression currently being reviewed)

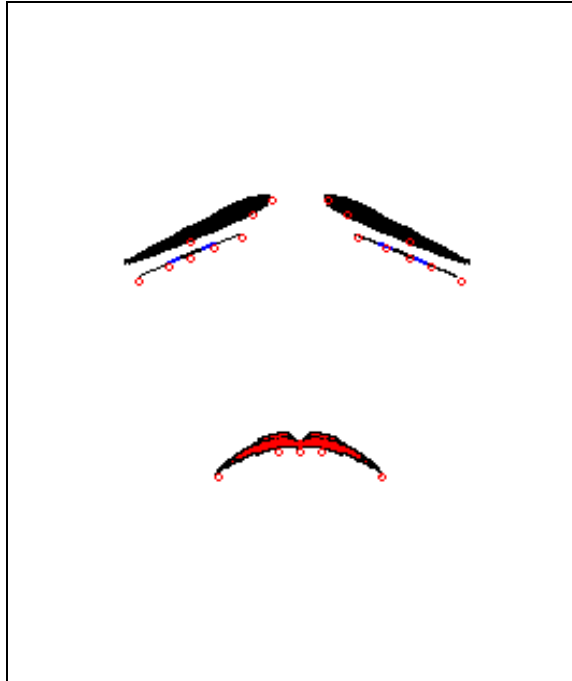
**Execution:**

*Until the end of  $Q$  is reached do:*

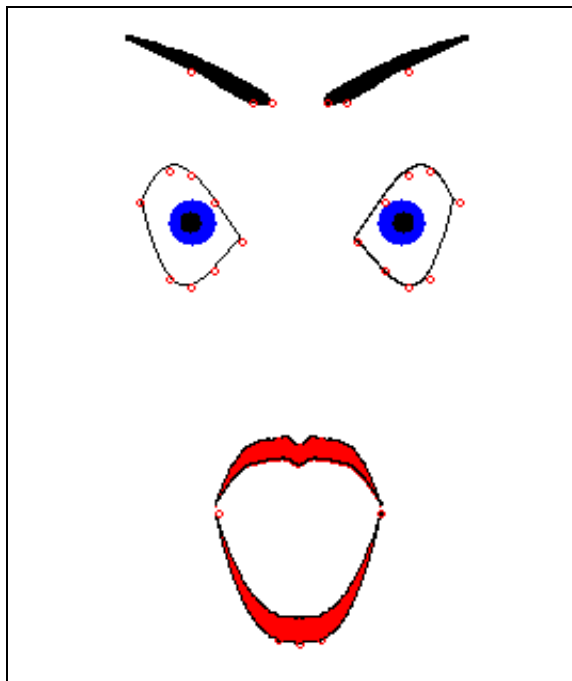
1. Determine the normalized distance  $D$  between facial expressions  $cur$  and  $min$ .
2. If  $D < 0$ , set  $min = cur$ .
3. Get the next expression from  $Q$  and store it in  $cur$
4. Go to step 1.

#### Algorithm 6.2: determining the minimum facial expression in a set

The maximum expression in the set can be determined in the same manner with step 2 of the execution equaling: if  $D > 0$  set  $max = cur$ . Figure 6.15a and 6.15b show the theoretical minimum and maximum facial expressions that can be constructed with FaceShop.



**Figure 6.15a: theoretical minimum facial expression**



**Figure 6.15b: theoretical maximum facial expression**

### 6.5.3 Incremental QPM implementation

Figure 6.16 shows the design of the Incremental QPM (see also figure 4.9). When a user selects a pivot expression at the INCQApplet or selects to view the previous pivot expressions, the subspace location string is updated and send to the QPIncrementalServlet of the communication layer. Here the subspace location string is received and passed along to the Incremental QPM. This subsection describes how the Incremental QPM creates an Incremental Query Subspace (4 clusters and corresponding pivot expressions) from the subspace location string by describing each step in detail.

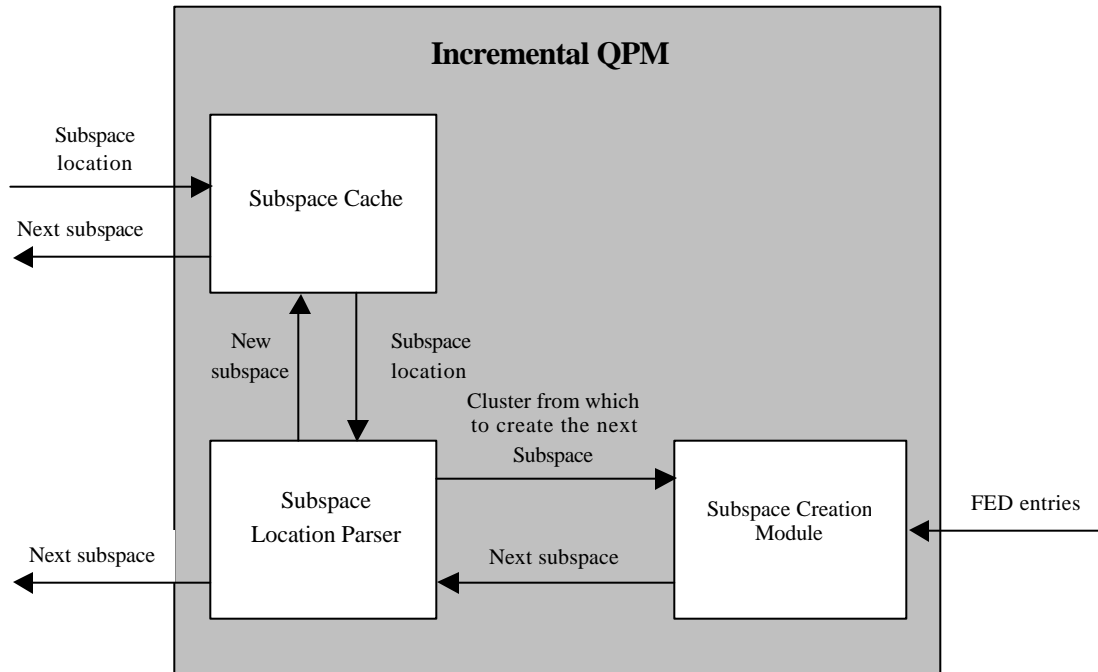


Figure 6.16: Incremental QPM design

#### 6.5.3.1 Subspace Location String

As mentioned in section 4.5.4, a subspace in the context of the Incremental Query is defined as the four clusters and corresponding pivot expressions that equally divide a certain set of facial expressions from the FED database. The state of the Incremental Query is determined by the subspace that is shown to the user. The state is encoded in the form of the *subspace location string*, which consists of integers ranging from 0-4. An integer represents the next cluster that is to be divided into four new clusters and pivot expressions. The first integer of the subspace location string is always equal to 0, indicating the initial subspace created from all active FED entries in the database.

#### 6.5.3.2 Subspace Cache

Because of the fact that relatively heavy calculations have to be performed in order to obtain an Incremental Query subspace from the subspace location string, a cache was incorporated in the Incremental QPM. The first action taken by the Incremental QPM is to check if the subspace

corresponding to the subspace location string received from the QPIncrementalServlet is present in the Subspace Cache. If this is the case, the next four clusters and pivots can be returned immediately. Otherwise, the next subspace has to be determined by the Subspace Location Parser and the Subspace Creation Module.

Each subspace that is created is stored in the Subspace Cache, if it is not already present. The Subspace Cache is cleared whenever a FED entry is added, edited or deleted from the FED administrative website, to avoid outdated subspaces containing data that is no longer accurate.

### 6.5.3.3 Subspace Location Parser

If the subspace corresponding to the received subspace location string is not present in the cache, the subspace has to be determined. The *Subspace Location Parser* proceeds to extract the first integer from the subspace location string. The first integer is always a zero, which tells the parser that the next subspace has to be created from all active entries in the FED database. The Subspace Creation Module determines the 4 clusters and corresponding pivot expressions.

Subsequently, the next integer is extracted. This integer indicates from which cluster of the initial subspace the next subspace has to be created by the Subspace Creation Module. This process is repeated until the end of the subspace location string is reached. The subspace determined at that point corresponds to the subspace indicated by the subspace location string and is returned to the applet, where it is presented to the user through the four pivot expressions.

### 6.5.3.4 Subspace Creation Module

The normalized features and algorithms for determining the minimum and maximum facial expression in a set, as described in the previous sections, are used to create an Incremental Query Subspace from a collection of facial expressions. Algorithm 6.3 shows the algorithm implemented in the Subspace Creation Module of the Incremental QPM that determines a subspace from a set of facial expressions. Figure 6.17 shows a graphical representation of the algorithm. The set of facial expressions for which 4 clusters and corresponding pivot expressions have to be determined is ordered ascending by normalized distances to the minimum facial expression in the set. Subsequently, the set is divided into 4 clusters of equal size. The facial expressions that lie the closest to the middle of each cluster become the pivot expressions.

Algorithm 6.3 is based on geometrical characteristics of the data, and can be influenced by outliers and a non-uniform distribution of the data, but although a huge database for testing the algorithm's performance to the full extent is not available yet, the results so far look promising.

1. determine feature normalization parameters
2. determine *min* : the minimum facial expression in the set
3. determine *max*: the maximum facial expression in the set
4. determine *middle*: the facial expression between *min* and *max*
5. determine *middle2* : the facial expression between *min* and *middle*
6. determine *middle3* : the facial expression between *middle* and *max*
7. determine *pivot1* : the facial expression between *min* and *middle2* (*cluster Q1*)
8. determine *pivot2*: the facial expression between *middle2* and *middle* (*cluster Q2*)
9. determine *pivot3*: the facial expression between *middle* and *middle3* (*cluster Q3*)
10. determine *pivot4*: the facial expression between *middle3* and *max* (*cluster Q4*)

**Algorithm 6.3: creating an Incremental Query Subspace from a set of facial expressions**

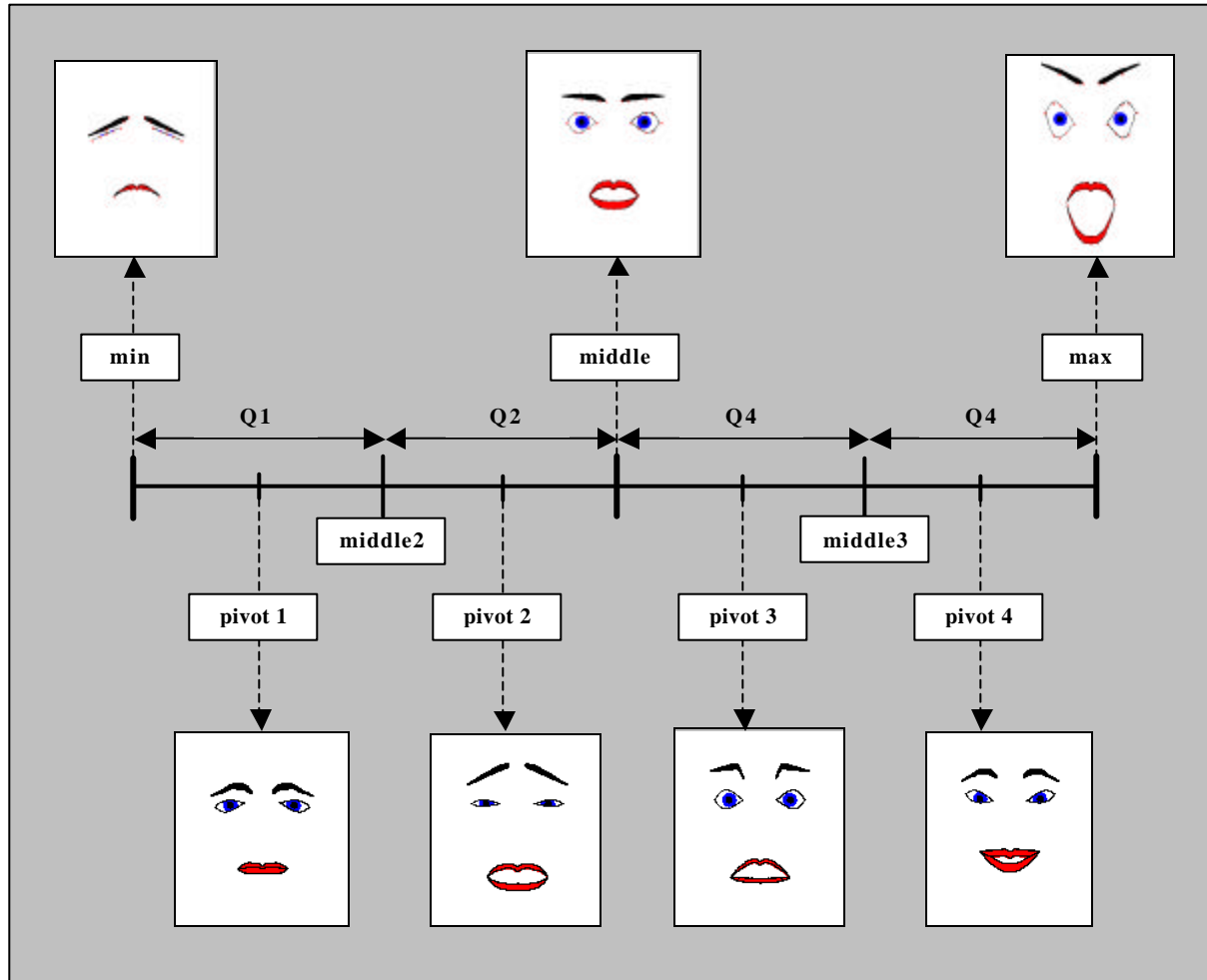
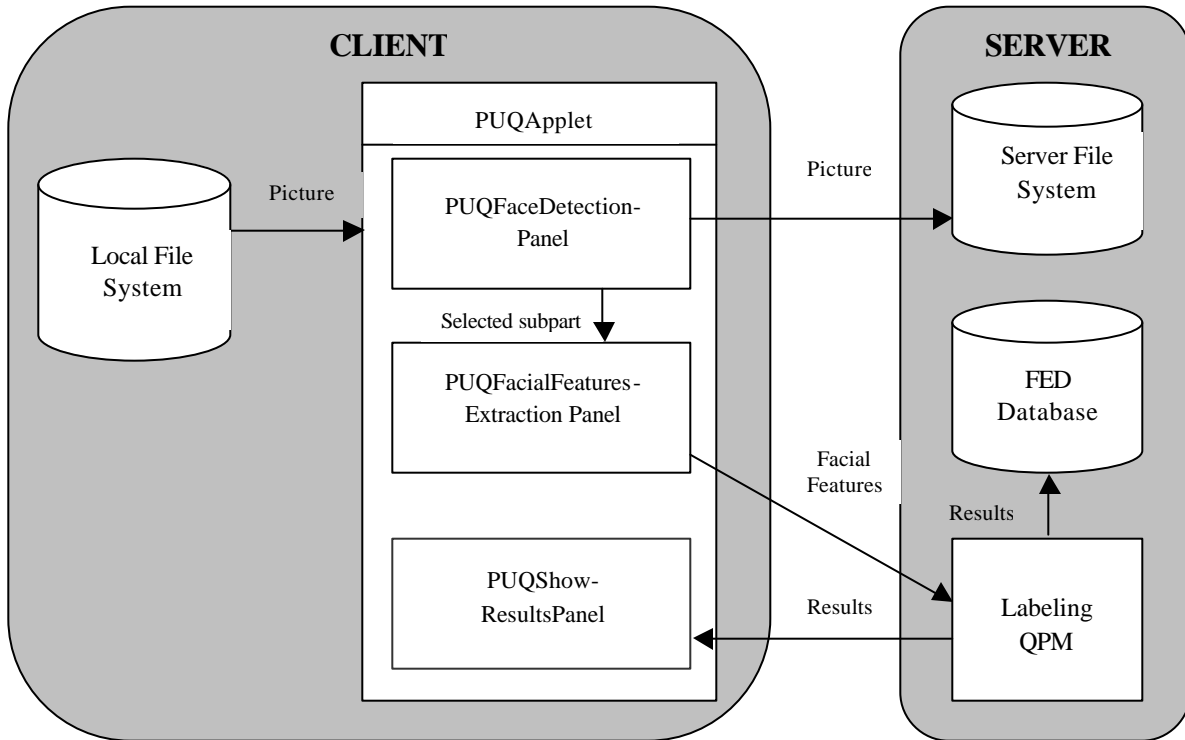


Figure 6.17: graphical representation of algorithm 6.3

## 6.6 Picture Query

When a user decides to issue a Picture Query, the label of an unknown facial expression shown in a picture has to be determined by FED. The Picture Query is implemented through the PUQApplet, which serves as a container for three panels that represent the three steps of the automatic facial expression recognition process: the PUQFaceDetectionPanel, PUQFacialFeaturesExtractionPanel and the PUQShowClassificationResultsPanel. Figure 6.18 shows how the Picture Query is implemented. The picture is selected by the user from the collection of files present on his local filesystem. Labeling of the facial expression takes place at the server and is performed by the Labeling QPM. If the labeling process was successful, the results are stored in the FED database.

How the three steps of the automatic facial expression recognition process are implemented partly depends on how facial expressions are represented in the system. As mentioned in section 3.1, FED uses the face model developed by Kobayashi and Hara to model facial expressions. The following three sections detail how the three steps of the automatic facial expression recognition are implemented.



**Figure 6.18: Picture Query design**

### 6.6.1 Face Detection

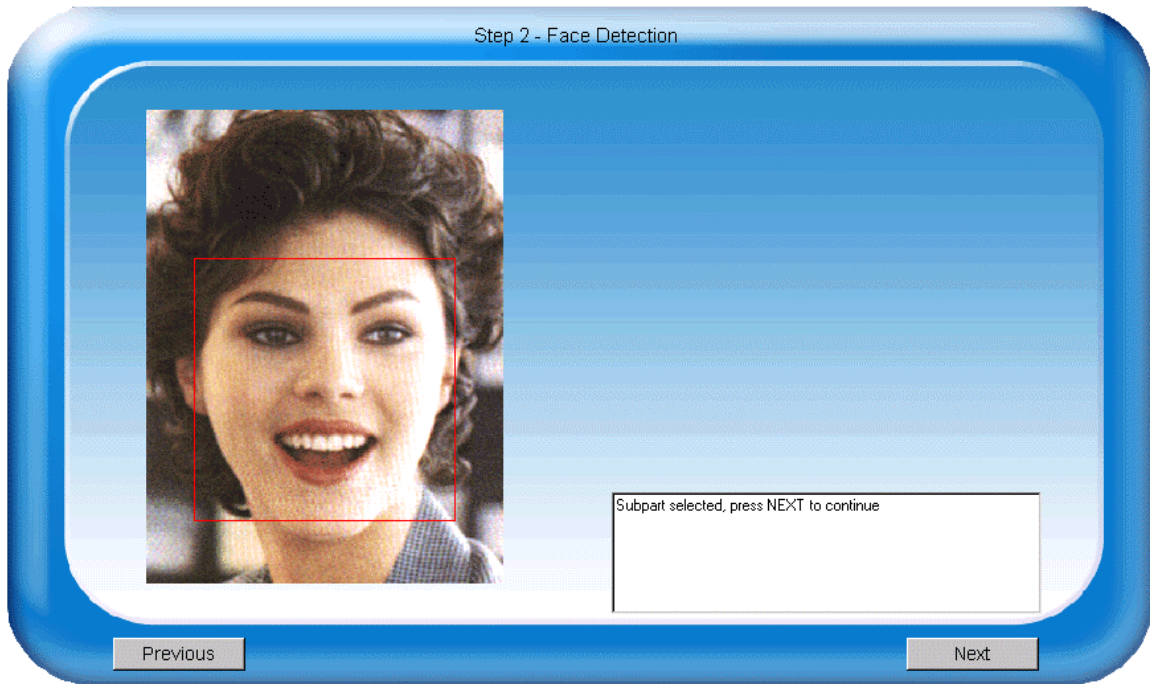
The face detection step of the automatic facial expression recognition process is performed manually by the user. There are ways to automatically detect the location of the face in the picture. This may be implemented in the future. The user could then ask the system for a suggestion for the location of the selection rectangle, by clicking a *system suggestion* button.

In the current prototype, the user is asked to use the mouse to select a subpart of the picture that contains the face. This is in fact optional, because the uploaded picture may already only contain a face. When the user clicks on the picture and drags the mouse, a selection rectangle is continuously drawn onto the picture to indicate the selection being made by the user. The user can undo this selection by clicking anywhere on the picture.

When the user is satisfied with the selection he has made, he can continue by clicking the *next* button. The user is then redirected to the next screen which implements the next step of the facial recognition proces, the feature extraction. Figure 6.19 shows a screenshot of the PUQFaceDetectionPanel that implements the manual face detection.

### 6.6.2 Feature Extraction

The feature extraction step of the facial expression recognition is performed semi-automatically: the user has to determine the position of the 30 facial characteristic points (FCP's) of the face model of Kobayashi and Hara, but the system provides feedback, help and restrictions on the FCP positions.



**Figure 6.19: screenshot of the PUQFaceDetectionPanel**

#### *Determining the FCP's*

Figure 6.20 shows a screenshot of the PUQFacialFeaturesExtractionPanel, which is used to implement the feature extraction. It shows the selected subpart of the picture selected by the user on the left, and an example picture showing the position of the 30 FCP's on the right.

The user is asked to subsequently position all 30 FCP's by clicking on the picture. A textfield informs the user which FCP is to be placed. The example picture shows where the FCP is supposed to be positioned. In figure 6.20, the first three FCP's have been placed. The blue rectangle indicates the area in which the next FCP a4 has to be placed. The concept of these so-called *Valid Areas* is discussed in more detail later in this section.

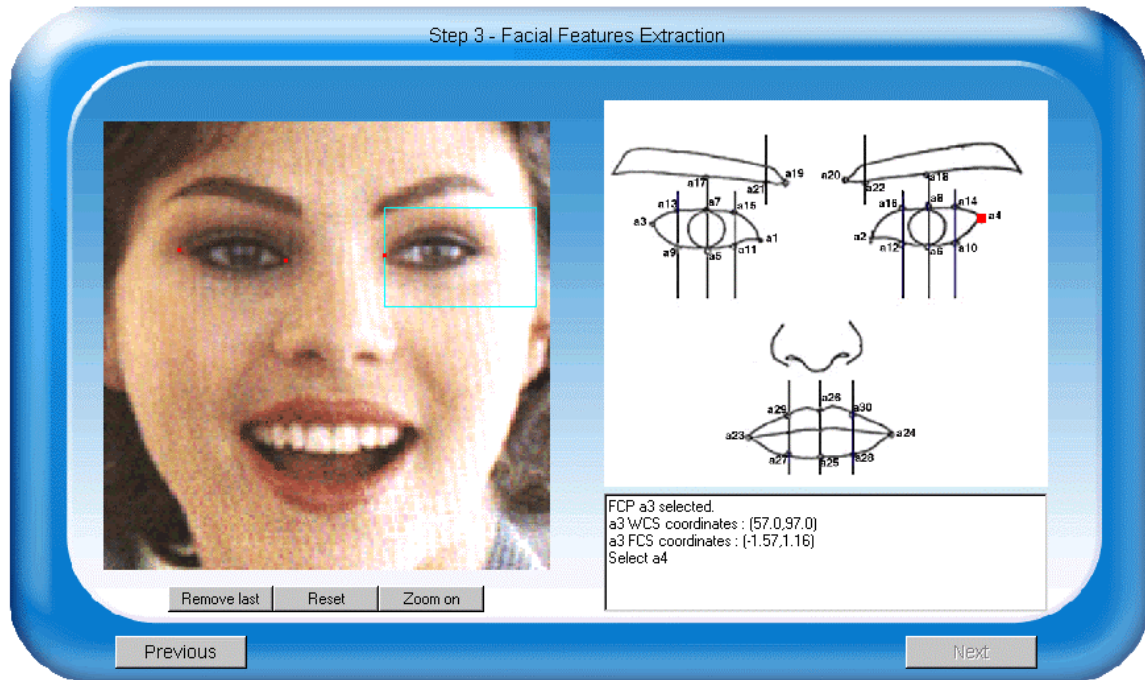
The user can remove the last FCP placed or reset the whole picture by clicking the *remove last* or *reset* button below the picture. When the user clicks on an existing FCP, this FCP is selected and a selection marker is drawn around it. This FCP can now be moved by dragging the mouse. When all FCP's have been placed, the *next* button is enabled and the user can proceed to the final step of the facial recognition process, the classification.

#### *Zooming*

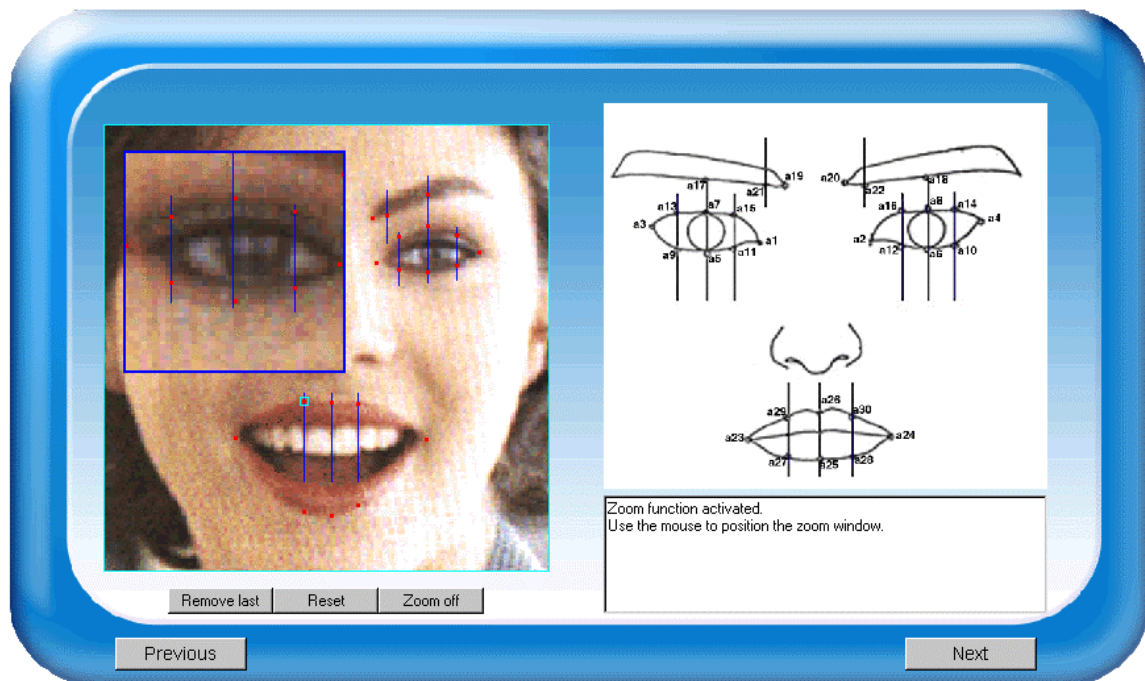
The user can use the zooming feature to position FCP's more accurately. The zooming feature is activated when the zoom button is clicked. A zoom window will then appear on top of the picture (see figure 6.21).

The zoom window can be moved over the picture by moving the mouse over the edges of the zoom window or by pressing the arrow keys, and displays the center of the section of the picture that it is covering, magnified by a factor two. In principle, the system is implemented to handle any magnifying factor, but in practice this has no added value, since the magnified section of the picture becomes unclear fairly quickly.





**Figure 6.20:** screenshot of the PUQFacialFeaturesExtractionPanel used to determine the FCP positions



**Figure 6.21:** screenshot of the PUQFacialFeaturesExtractionPanel with active zoom window

FCP's can be placed and moved in the zoom window in exactly the same way as they can be placed and moved in the picture. The difference is that FCP's placed or moved in the zoom window can be positioned with an accuracy of half a pixel. It must be noted that the coordinates

of the haralines are always calculated with a precision of half a pixel, whether the zooming feature was used or not.

If a FCP is positioned, the coordinates of the FCP have to be transformed from *zooming coordinate system* (ZCS) to *world coordinate system* (WCS) coordinates. This transformation can be described as follows:

$$x_{wcs} = z\_origin\_x + \left( \left( \frac{1}{zoom\_factor} \right) * x_{zcs} \right)$$

$$y_{wcs} = z\_origin\_y + \left( \left( \frac{1}{zoom\_factor} \right) * y_{zcs} \right)$$

Where  $z\_origin\_x$  and  $z\_origin\_y$  are the WCS coordinates of the top left corner of the zoom window. When the zoom window is created, the coordinates of the FCP's and haralines that are visible in the zooming window are transformed from WCS to ZCS coordinates for drawing purposes:

$$x_{zcs} = (x_{wcs} - z\_origin\_x) * zoom\_factor$$

$$y_{zcs} = (y_{wcs} - z\_origin\_y) * zoom\_factor$$

### Normalization

As mentioned in section 2.4.2, Kobayashi and Hara normalize input images by applying three transformations to the coordinates of the FCP's: a translation, a rotation and a scaling. The coordinates of the FCP's are in effect transformed from one coordinate system, the *world coordinate system* (WCS), to another, the *facial coordinate system* (FCS). Fig 6.22 shows the relation between the FCP coordinates in WCS and the FCP coordinates in FCS.

The three transformations are defined as follows:

- Translation

First of all, the origin of the coordinate system is shifted to the tip of the nose. This is achieved by first calculating the distance (in pixels) between the eyes (i.e. the distance between the first two FCP's  $a1$  and  $a2$ ). This distance becomes the base quantity of the FCS.

$$base = \sqrt{((a2.x - a1.x)^2 + (a2.y - a1.y)^2)}$$

The midpoint between FCP's  $a1$  and  $a2$ ,  $(x0,y0)$ , is also determined. The origin  $(origin\_x, origin\_y)$  is then positioned base perpendicular to  $(x0,y0)$ . All FCP's are translated over  $(origin\_x, origin\_y)$ .

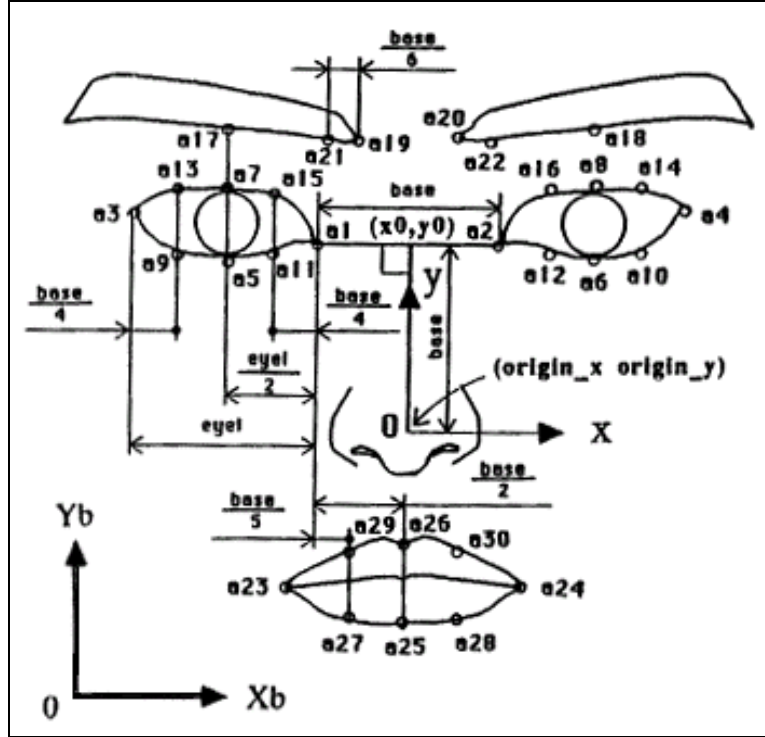


Figure 6.22: WCS to FCS transformation of FCP coordinates

$$origin\_x = \frac{1}{2}(a1.x + a2.x) - (a2.y - a1.y)$$

$$origin\_y = \frac{1}{2}(a1.y + a2.y) + (a2.x - a1.x)$$

- Rotation

All FCP's are rotated so that all FCP's are expressed with respect to the z-axis of the FCS to compensate for variations in in-plane orientation. The rotation angle is defined by:

$$a = \tan^{-1} \frac{(a2.y - a1.y)}{(a2.x - a1.x)}$$

- Scaling

All FCP's are divided by *base*, in order to compensate for the distance between the camera and the face and anthropomorphic differences between people.

### Continuous transformation between WCS and FCS

The FCS is created as soon as the user has positioned the first two FCP's  $a_1$  and  $a_2$ . The coordinates of all FCP's are known in WCS and FCS coordinates throughout the feature selection process.

In principal, the normalization of the pictures could take place after all FCP's have been positioned. However, because the coordinates of the haralines depend on the FCS coordinates of certain FCP's, it is necessary to continuously transform the coordinates of the FCP's from WCS to FCS coordinates. The transformation from WCS to FCS coordinates can be expressed as ( $origin\_x=tx$ ,  $origin\_y=ty$ ):

$$(x_{fcs} \quad y_{fcs} \quad 1) = (x_{wcs} \quad y_{wcs} \quad 1) \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -tx & -ty & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{base} & 0 & 0 \\ 0 & \frac{1}{base} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos a & \sin a & 0 \\ -\sin a & \cos a & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The last three matrices represent the translation, scaling and rotation of the FCP's. These matrices do not depend on the coordinates of any of the FCP's, and can be reduced to a single 3x3 matrix as soon as the FCS is created. This reduces the transformation to eight floating point operations. This resulting transformation matrix has to be evaluated again if  $a_1$  or  $a_2$  is moved (i.e if the FCS has changed).

If a coordinate set is loaded, or the FCS coordinates of a FCP change (which can happen if the position of the haraline it depends on change), the WCS coordinates have to be recalculated from the FCS coordinates. This is done by applying the inverse of the transformation described in the preceding subsection:

$$(x_{wcs} \quad y_{wcs} \quad 1) = (x_{fcs} \quad y_{fcs} \quad 1) \begin{pmatrix} base & 0 & 0 \\ 0 & base & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos a & \sin a & 0 \\ -\sin(-a) & -\cos(-a) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ tx & ty & 1 \end{pmatrix}$$

### Semi-automatic feature extraction

The feature extraction process is thus a manual process. However, by imposing x-coordinate restrictions on 22 FCP's through the concept of *haralines*, described later in this section, the process of feature extraction can be made semi-automatic. Also, FCP's can only be placed or moved freely in certain area's defined by the position of other FCP's. By implementing an intelligent checking mechanism the user is restricted to placing or moving a FCP in the for that FCP valid area. This intelligent checking mechanism is also described in more detail later in the next section.

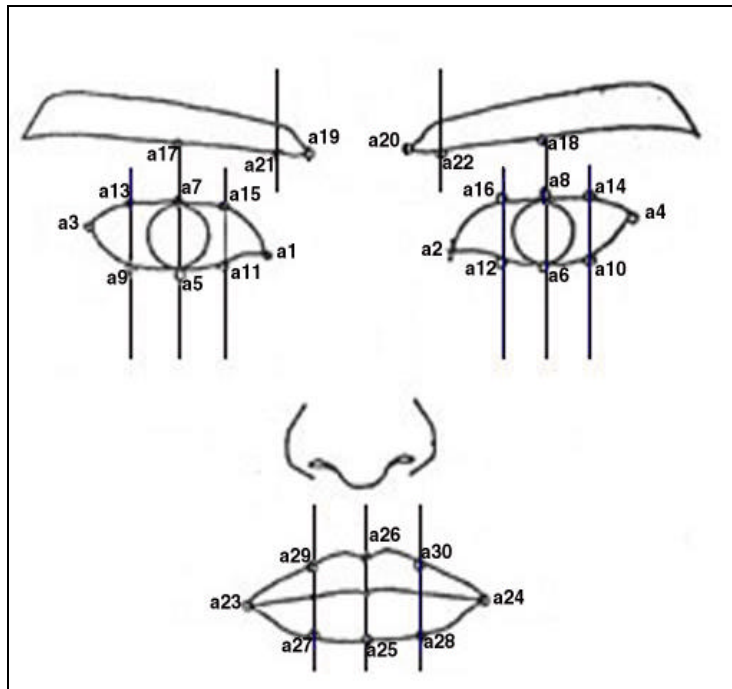
### *Haralines: dependencies between FCP coordinates*

In the face model defined by Kobayashi and Hara, the 30 FCP's are located at the contours of the eyebrows, eyes or mouth. To locate these points, vertical lines called *haralines* are defined. The x-coordinates of the haralines depend on the distance between the inner corners of the eyes. The x-coordinates of 22 of the 30 FCP's are defined by the x-coordinate of a corresponding haraline. figure 6.23 shows where the haralines are located and the FCP's that depend on them.

During the positioning of the 30 FCP's, the haralines are drawn onto the picture. If the user places a FCP whose x-coordinate depends on the position of a haraline, the FCP is snapped to this haraline. Also, if a FCP whose x-coordinate depends on a haraline is being moved, it can only be moved along that haraline. If a FCP that determines the position of one or more haralines is moved, these haraline(s) and all currently placed FCP's whose x-coordinate(s) depend on them move in the picture accordingly. Because of these dependencies, the FCS x-coordinates of certain FCP's are fixed (see table 6.5).

The order in which the position of the haralines is determined differs from the order in which they are displayed in the picture. This different order of appearance of the haralines is introduced because it may seem more intuitive to the user.

After positioning the first four FCP's, the two haralines through the centers of the eyes are drawn first. These are located exactly in the middle of the eyes between FCP's a3 and a1 and FCP's a2 and a4 respectively. When the user has positioned FCP a5 through a8, the remaining four haralines through the eyes are displayed too. These haralines are located at a distance of  $(1/4 * base)$  from FCP a1, a2, a3, a4 respectively. After the positioning of FCP's a19 and a20, the two haralines through the eyebrows are displayed. These haralines are positioned  $(1/6 * base)$  from FCP a19 and a20 respectively. The last three haralines through the mouth are displayed only after FCP's a23 and a24 have been placed. The haraline in the middle is located exactly in the middle of FCP's a23 and a24. The left haraline is located  $(1/5 * base)$  to the right of a1, the right one  $(1/5 * base)$  to the left of a2.



**Figure 6.23: The Haralines in the face model of Kobayashi and Hara**

**Table 6.5**  
**Dependencies between FCP's**

FCP	FCS X-coordinate
a1	-0.5
a2	0.5
a5, a7, a17	$0.5 * (a3 + a1)$
a9, a13	$a3 + 0.25$
a11, a15	-0.75
a6, a8, a18	$0.5 * (a2 + a4)$
a10, a14	$a4 - 0.25$
a12, a16	0.75
a25, a26	0
a27, a29	-0.3
a28, a30	0.3

### *Intelligent Checks*

It makes no sense to allow the user to place or move FCP's to any position in the picture. The position of some FCP's logically depends on the positions of other FCP's. For example, FCP a5, at the bottom center of the left eye, should always be located between FCP's a3 and a1. These restrictions are implemented as valid area's for each FCP. When the user places or moves a FCP, the valid area for that FCP is calculated and drawn onto the picture as a rectangle. The user can only place or move the FCP within this rectangle. Table 6.5 shows the dimensions of the valid area's for all FCP's. Some valid area's have several possible dimensions. This comes from the fact that if a FCP is moved, it is possible that some FCP's have been placed already that make a more accurate definition of the valid area possible. For example, if the user moves a1, and a3 has not been placed, the left border of the valid area is the left border of the picture. If a3 has been placed, the left border of the valid area can be determined more specific as a3.

### 6.6.3 Classification – determining the label of the unknown facial expression

Once the coordinates of the FCP's have been determined, the user can let FED determine the label of the unknown facial expression. This is accomplished by comparing the FCP coordinates of the unknown facial expression to the FCP coordinates of the (active) facial expressions present in the FED database. Again, as with the Incremental Query, the most discriminative facial features of all active FED entries were used to implement the classification algorithm (see table 6.1). Algorithm 6.5 is used by the Labeling QPM to determine the label of an unknown facial expression.

The first step performed by the Labeling QPM is the calculation of the feature normalization parameters. The algorithm used to perform these the normalization parameters is identical to the one used in the Incremental QPM implementation (algorithm 6.1).

When the normalization parameters have been determined, the normalized feature values of the expression that is to be labeled are determined. Because of the fact that the FCP's of the unknown facial expression have been determined manually, the facial features will most likely not be symmetrical, as they are with facial expressions generated with FaceShop. The facial features of table 6.1 are therefore averaged over both sides of the facial expression.

**Table 6.6**

**Area's where FCP's can be placed and / or moved freely (R, L, T, B = right, left, top and bottom of picture border)**

FCP	Left	Right	Top	Bottom
a1	a3; L	a2; R	2.0; T	0.0; B
a2	a1	a4; R	2.0; T	0.0; B
a3	L	a1; a5	2.0; a7	0.0; a5
a4	R	a2; a6	2.0; a8	0.0; a6
a5	a3; a9	a1; a11	2.0; a7	0.0
a6	a2; a12	a4; a10	2.0; a8	0.0
a7	a3; a13	a1; a15	2.0; a17	a7
a8	a2; a16	a4; a14	2.0; a18	a6
a9	a3	a5	a7	0.0
a10	a6	a4	a8	0.0
a11	a5	a1	a7	0.0
a12	a2	a6	a8	0.0
a13	a3	a5	2.0	a9
a14	a8	a4	2.0	a10
a15	a7	a1	2.0	a11
a16	a2	a8	2.0	a12
a17	a13	a15	T	a7
a18	a16	a14	T	a8
a19	a17	0.0	T	a1
a20	0.0	a18	T	a2
a21	a17	a19	T	a7
a22	a20	a18	T	a8
a23	L	0.0	0.0	B
a24	0.0	R	0.0	B
a25	a23; a27	a24; a28	0.0; a26	B
a26	a23; a29	a24; a30	0.0	a25
a27	a23	a25	0.0; a28	B
a28	a25	a24	0.0; a29	B
a29	a23	a26	0.0	a27
a30	a26	a24	0.0	a28

The feature values are compared to the normalized feature values of all facial expressions in the FED database. The distance per individual feature, facial feature (eyebrows, eyes and mouth) and the average feature distance are calculated.

Then a number of facial expressions is selected as initial candidate matches. The facial expression with the shortest average feature distance and all facial expressions with a average feature distance not larger than twice the shortest distance are selected. Also, all facial expressions with a average distance less than 0.03 longer than the shortest distance are included.

Not all of these initial candidates will be valid matches of the unknown facial expression. These invalid facial expressions are filtered out. All facial expressions with an individual feature distance larger than 0.4 are excluded, as are all facial expressions with an average facial feature distance  $> 0.2$  or a total average distance  $> 0.2$ .

In principle, all resulting facial expressions are considered valid matches of the unknown facial expression. However, if the number of valid matches is larger than three, the Labeling QPM is

apparently not able to distinguish which facial expression is shown, and zero matches are returned. In principle, the distance criteria of the algorithm could then be relaxed, so as to return at least one match. Figure 6.24 shows the classification results after the feature extraction step was completed for the picture in figure 6.21.

### Initialization:

- Let *unknown\_exp* be the facial expression that is to be labeled
- Let *Q* be the set of all facial expressions from the FED database
- Let *cur\_exp* be the currently examined facial expression from the FED database
- Let *matching\_entries* be the set of resulting matching facial expression entries

### Execution:

*Determine the feature values of the facial expression that is to be labeled:*

1. Determine the feature normalization parameters *eb\_height\_norm*, *eb\_frowned\_norm*, *e\_openness\_norm*, *e\_slanting\_norm*, *m\_openness\_norm*, *mos\_norm* (algorithm 6.1)
2. Determine the normalized feature values of *unknown\_exp*

$$\begin{aligned}
 eb\_height\_u &= eb\_height\_abs / eb\_height\_norm \\
 eb\_frowned\_u &= eb\_frowned\_abs / eb\_frowned\_norm \\
 e\_openness\_u &= e\_openness\_abs / e\_openness\_norm \\
 e\_slanting\_u &= e\_slanting\_abs / e\_slanting\_norm \\
 m\_openness\_u &= m\_openness\_abs / m\_openness\_norm \\
 mos\_u &= mos\_abs / mos\_norm
 \end{aligned}$$

*Determine feature distances to all facial expressions in the FED database:*

*Until the end of Q is reached do:*

- 1) Get the next facial expression from *Q* and store it in *cur\_exp*
- 2) Determine the normalized feature values of *cur\_exp* (as step 2)
- 3) Determine the feature distances between *unknown\_exp* and *cur\_exp*

$$\begin{aligned}
 eb\_height\_distance &= (eb\_height\_u - eb\_height\_c) \\
 eb\_frowned\_distance &= (eb\_frowned\_u - eb\_frowned\_c) \\
 e\_openness\_distance &= (e\_openness\_u - e\_openness\_c) \\
 e\_slanting\_distance &= (e\_slanting\_u - e\_slanting\_c) \\
 m\_openness\_distance &= (m\_openness\_u - m\_openness\_c) \\
 mos\_distance &= (mos\_u - mos\_c)
 \end{aligned}$$

- 4) Determine the average feature distance of the eyebrows, eyes and mouth:

$$eb\_distance = (eb\_height\_distance + eb\_frowned\_distance) / 2.0$$



$$e\_distance = (e\_openness\_distance + e\_slanting\_distance) / 2.0$$

$$m\_distance = (m\_openness\_distance + mos\_distance) / 2.0$$

- 5) Determine the total normalized distance  $D$

$$D = \text{total normalized distance between } unknown\_exp \text{ and } cur\_exp = \\ (eb\_height\_distance + eb\_frowned\_distance + e\_openness\_distance + \\ e\_slanting\_distance + m\_openness\_distance + mos\_distance) / 6.0$$

- 6) Goto step 1)

*Determine the initial matches:*

4. Determine  $shortest\_distance = \min(D)$ ;
5. Store the facial expression entry with  $D = shortest\_distance$  in *matching\_entries*
6. Store all facial expressions with the following property in *matching\_entries*:

$$(D < 2.0 * (shortest\_distance)) \text{ OR } (D < (shortest\_distance + 0.03))$$

*Filter out unlikely matches:*

7. Remove all facial expression entries with total distance  $D > 0.2$  from *matching\_entries*
8. Remove all facial expression entries with an individual feature distance ( $eb\_height\_distance, eb\_frowned\_distance, e\_openness\_distance, e\_slanting\_distance, m\_openness\_distance, mos\_distance$ )  $> 0.4$  from *matching\_entries*
9. Remove all facial expression entries with an average feature distance ( $eb\_distance, e\_distance, m\_distance$ )  $> 0.2$  from *matching\_entries*

*Return results:*

10. Determine the matching probability =  $(1.0 - D)$  for each facial expression entry stored in *matching\_entries*
11. If (number of facial expressions stored in *matching\_entries*  $> 3$ )  
     classification fails  
     else  
     return *matching\_entries*

**Algorithm 6.5: Determining the closest matching entries of an unknown facial expression**

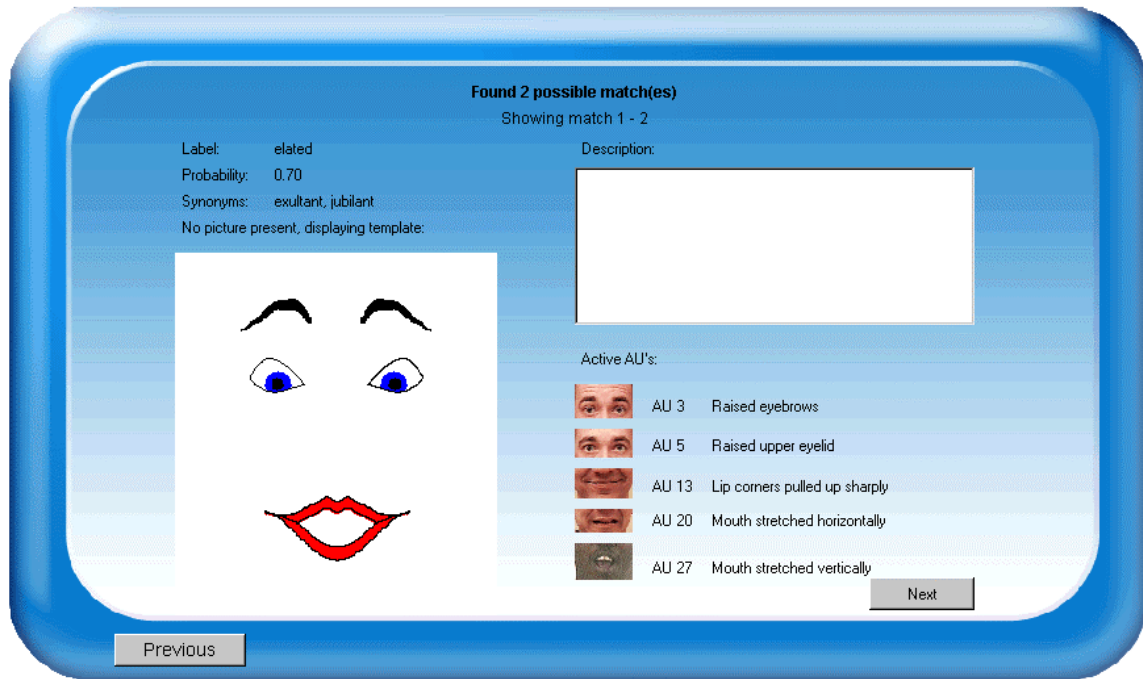


Figure 6.24: Picture Query results

## 6.7 FaceShop Query

Like the Picture Query, the FaceShop Query allows a user to determine the label of an unknown facial expression. In this case however, no face detection is needed and the facial feature extraction step is performed automatically. The user only has to sketch a facial expression and submit the query. The algorithm used to classify the sketched facial expression is identical to the algorithm used with the Picture Query (see section 6.6.3). Figure 6.25 shows the FSQApplet, which implements the GUI for the FaceShop Query. Figure 6.26 shows the result when the process query button is pressed.

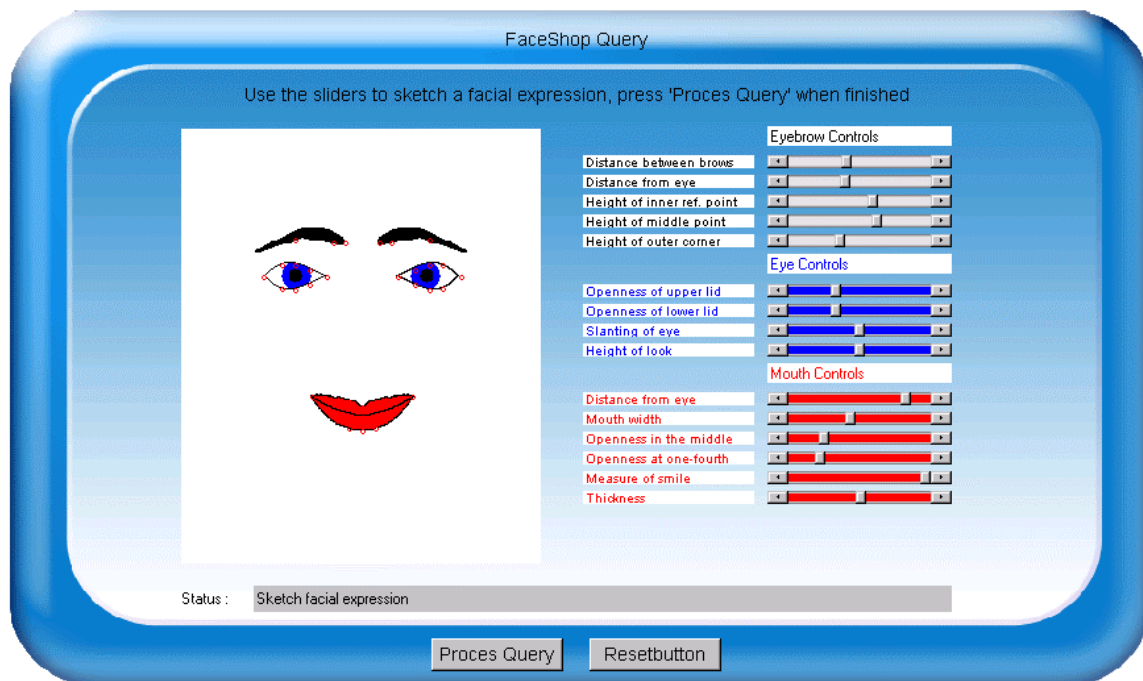


Figure 6.25: screenshot of FSQApplet with a sketched facial expression

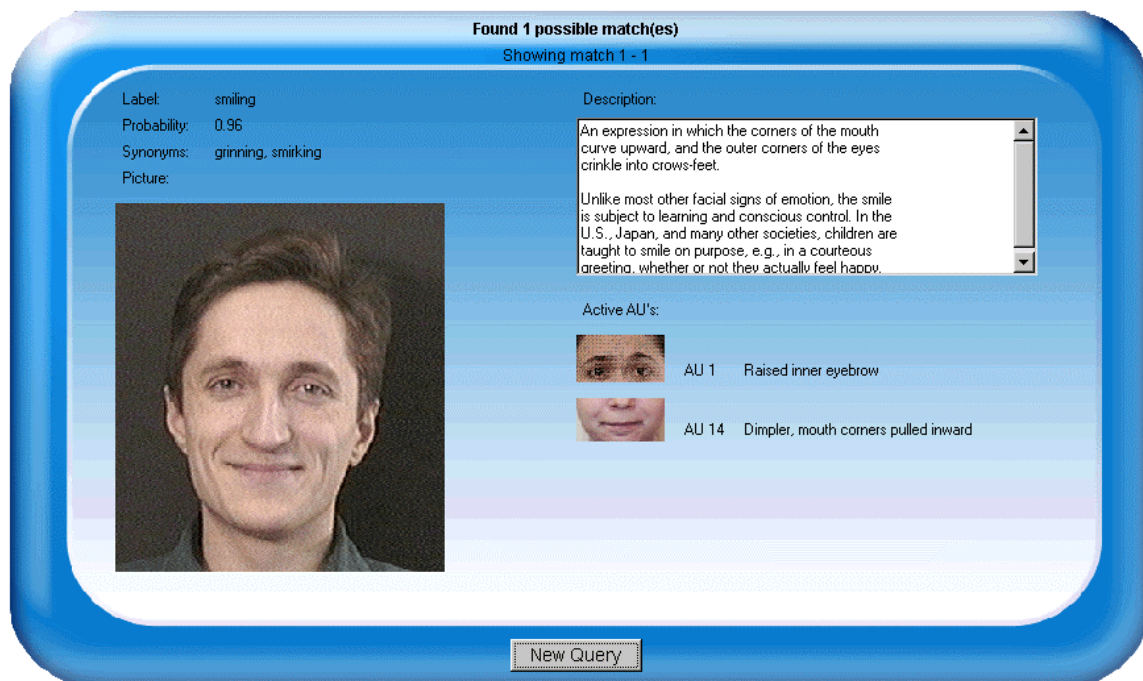


Figure 6.26: screenshot of the FSQApplet displaying the label result of the sketched facial expression of figure 6.25

## Chapter 7: Results

*This chapter details the results of the project. Section 7.1 analyzes the performance of the FED website. Section 7.2 gives an evaluation of the FED corpus, followed by an analysis of the performance of the different queries into FED in section 7.3.*

### 7.1 Website performance

This subsection describes the performance of the FED website. Important aspects of the performance are the usability, speed, scalability, extendibility and adaptability and security of the website, which are discussed in the following subsections.

#### 7.1.1 Usability

The FED website was tested by a group of 30 students. The GUI of the various query possibilities proved to be a valid approach; the design was approved of and in general, the explanation of the query was found to be satisfactory. It should be noted that these students all are skilled with computers, and the query explanation may have to be simplified or extended when FED become available to the general public. Small changes to improve the usability have been made based on suggestions of users.

The FED administrative backend website has been online from March 2002, and has proved extremely useful and time-saving when managing the FED corpus and verifying query performances.

#### 7.1.2 Speed

When discussing the speed of the FED website, the most important issue is the time that is needed to process a query request and display the results to the user. A number of factors determine the time needed.

First of all, the speed and memory of the server determine how long it will take to receive and process a certain query request. The processing time may vary for each different type of query. In the case of FED, the server has a 833MhZ processor. This server is capable of processing query requests fairly fast. The second column of table 7.1 shows the average time needed to process a certain query at the server (FaceShop and Picture Query processing and GUI creation are identical). Of course, complicated queries will take a bit longer to process, but the differences are negligible. It is difficult to estimate the time needed to process a query at the server, when the number of entries in FED is much larger than the current 56. Given the fact that 7000 entries could eventually be stored in the FED database, the performance of the query processing modules may have to be evaluated again at that time.

**Table 7.1**  
**Query processing times**

Query	Average process time at server	Average total response time
Label	0.224	1.600
Action Unit	0.250	1.913
Geometry	0.362	2.124
FaceShop / Picture	0.288	1.183

( Client - Intel Pentium 933 MhZ; Server - Intel Pentium 833 MhZ; Bandwidth - 100 Mbit/s )

A more important factor determining the total time needed to process a query request and display the results to the user than the capacity of the server is the speed and bandwidth of the available internet connection. With certain queries, it is possible that the number of results is quite large. For example, it is possible to issue a Geometry Query for ‘mouth open + mouth closed’, which will return most of the entries in FED. In that case, the amount of data that has to be sent from the server to the client is also quite large. Additionally, for each resulting entry that has a picture associated with it, this picture has to be sent to the client as well (at the moment the user selects to view the entry).

Finally, the speed and available memory of the user’s computer determines the speed with which the Java applets are executed. Java is not known for its lightning speed execution, and when testing the various queries with a Pentium 200 MhZ computer, it took up to 20 seconds to create the GUI needed to display the results of the query. The third column of table 7.1 shows the average total response time per query type. The average total response time of the FaceShop and Picture Query is relatively small, because of the fact that the average number of results that is to be displayed is generally small (3 at most).

### 7.1.3 Scalability

If in the future the FED website is visited by many hundreds or even thousands of people a day, an important issue will be the number of users that can issue a certain query at the same time. This depends on the capacity of the Java servlets, which handle all the data traffic between the client and the server, and on the capacity of the server itself.

The system administrator can alter the amount of resources available to each servlet, and if the server is unable to handle all query requests, it is possible to replace it with a new server with a larger capacity, or add one or more additional servers to process all query requests. In this way, the FED website is scalable to handle large numbers of query requests simultaneously.

### 7.1.4 Extendibility / Adaptability

The FED website is set up modularly in two respects. This modular design ensures that adaptation and extension of the FED website can be accomplished with relative ease.

Firstly, all FED query possibilities and FED management functions are implemented separately. This means that a developer adapting or extending the functionality of a certain query possibility or management function doesn’t need to concern himself with the implementations details of existing query possibilities or management functions. For the same reason, adding a new query possibility or management function is relatively easy.

Secondly, the implementations of all FED query possibilities and FED management functions are set up modularly themselves. The GUI is always implemented in a specific Java applet, communication with the server is performed through one or more Java servlets and all processing and database access takes places through Java code residing on the server. This means that adapting a certain FED query possibility or FED management function usually involves adapting only one of the components (as long as the interface with the connected components does not change).

### 7.1.5 Security

An important issue concerning the security of the FED website is the security of the FED database. It would be disastrous if malice users were able to gain access to the database and corrupt the data. Database access from the Java applets on the client computer would be to insecure. Special tools for decompiling Java .class files exists, with which the database security

information could be determined from the applet's source code with relative ease. To increase the security of the database, all database access takes place at the server.

Also important is the security of the FED administrative website, where authenticated users have the possibility to add, edit or delete FED entries and FED admin users. To increase security, users can only gain access to the administrative website by supplying a valid username and password. Furthermore, all the HTML code that makes up the administrative website is generated by calling a dedicated Java servlet with specific parameters. No actual HTML-files exist. This ensures that malice users cannot bypass the user authentication procedure by directly loading an HTML-page belonging to the administrative website in their browser.

### 7.2 FED corpus

At the moment, the FED database contains 56 active facial expression entries. For this graduation project this number of entries was sufficient. The main goal of the project was to develop a prototype and test the concept of an online Facial Expression Dictionary. Completing the database to contain all 7000 possible facial expressions and ensuring the scientific validity is something that can be considered when FED is completely finished.

Statistical analysis was performed on the feature vectors (FCP's) to get an idea of the distribution of the facial expressions. Sammon's mapping is an iterative method based on a gradient search (Sammon Jr, 1969). The aim is to map points in  $n$ -dimensional space usually into 2 dimensions. The algorithm finds the locations in the target space so that as much as possible of the original structure of the measurement vectors in the  $n$ -dimensional space is conserved. Figure 7.1 illustrates the principle of Sammon's mapping.

Figure 7.2 shows the distribution of the feature vectors of all active facial expressions present in the FED database. Figure 7.2 was obtained using Sammon's mapping. As becomes clear from figure 7.2, certain facial expressions that are similar intuitively are also positioned relatively close together in the feature space. For example, *'disgust'* and *'appalled'* are located close together, as are *'tired'* and *'exhausted'*.

This is however not always the case. The facial expressions *'peaceful'* and *'worried'* are examples of the latter. Although the interpretations of these facial expressions differ greatly, they are located closely together in the feature space. This does not necessarily mean that the facial expressions have been defined incorrectly. It is very well possible that there are only subtle differences between two facial expressions, while the interpretations of the facial expressions differ greatly.

Another observation that can be made is that a relatively large number of facial expressions are located in the center of figure 7.2. This is the reason that no absolute distance metric was used when trying to find the closest matching entries (algorithm 6.5). If the unknown facial expression is located somewhere at the top right of the feature space, the algorithm should return *'triumphant'*. If the unknown facial expression is located somewhere in the center of the feature space, certain entries are not even considered to be valid matches, although their distance to the unknown facial expression is smaller than the distance to *'triumphant'* in the previous case.

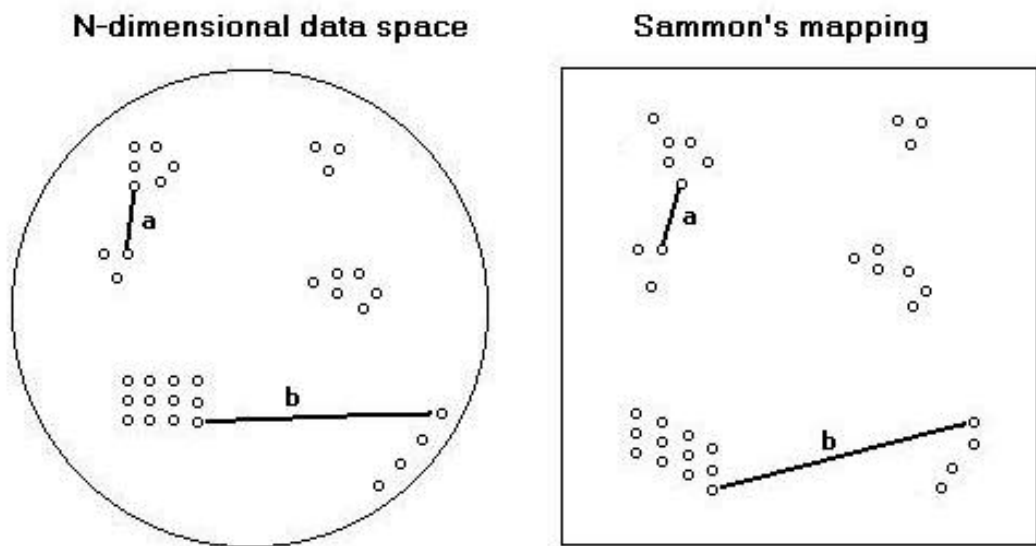


Figure 7.1: Sammon's mapping

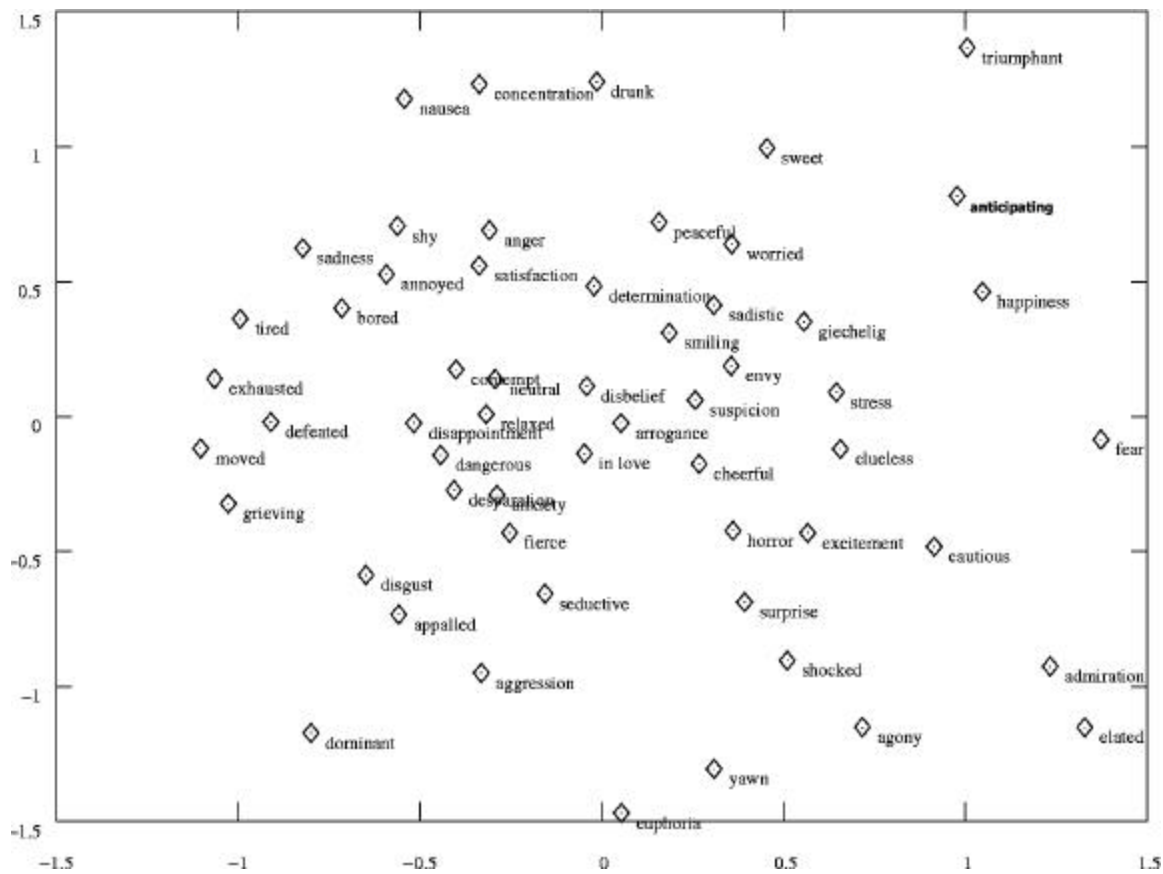


Figure 7.2: Distribution of FED entries

### 7.3 Search modalities

As mentioned earlier a group of approximately 30 students have tested the FED website. Several useful remarks led to minor improvements, but essentially the consensus was that issuing a query into FED worked well. Table 7.2 gives an overview of the queries issued in the period FED was tested. The Incremental Query is not included in the overview (it cannot be measured in terms of success or failure, as the other queries). The remainder of this section will describe the specifics of all six search modalities in more detail.

**Table 7.2**  
**Query test results**

Query	#Queries	#Successful queries	Hitrate	Average # results
Label	149	89	0.59	0.7
Action unit	36	34	0.94	11.4
Geometry	29	19	0.65	4.9
FaceShop	144	89	0.61	0.9
Picture	46	25	0.54	1.3

#### *Label Query*

When a Label Query was issued, one or more resulting matches were found in almost 60% of the cases. Given the fact that not every facial expression label entered would have been present in FED, this indicates that the Label Query functions adequately. Additional functionality that could be implemented are the automatic completion of entered keywords to facial expression labels present in the FED database, and the possibility to browse through all entries alphabetically on facial expression label.

#### *Action Unit Query*

Given the fact that almost every AU is present in at least one facial expression of FED, it is not surprising to see that an Action Unit Query returned one or more results in almost all cases. At the moment, the AU QPM returns all facial expressions with at least one matching AU. As the number of entries in FED increases, a more logical approach would be to return only the facial expressions with *all* AU's of the query present. Also, it is desirable to extend the number of possible AU's associated with a facial expression in FED from 30 to 44. This would involve adapting the FaceShop implementation, because of the fact that at the moment certain AU's cannot be modeled with FaceShop.

#### *Geometry Query*

Issuing a Geometry Query involves a little getting used to. Users have to grasp the concept of issuing a textual query as a regular expression. In principle, it is possible to issue queries that return no results by specifying features that exclude each other: the query 'mouth open & mouth closed' will not return any facial expressions. This explains the fact that 10 of 29 queries didn't return any matches.



### *FaceShop Query*

In 61% of the cases, the Labeling QPM is able to determine one or more probable matches of an unknown facial expression sketched with FaceShop. This is about as good as can be expected, given the fact that for each facial expression there is only one template is present in FED, while certain facial expressions can be sketched in several different ways. For example, if ten people were asked to sketch the facial expression *'happiness'*, it is very unlikely that the resulting sketches are exactly alike, while all ten expressions could be valid representations of *'happiness'*.

### *Picture Query*

The Labeling QPM is also used to determine the label of a facial expression shown in a picture. In comparison with the FaceShop Query, the percentage of successful queries is slightly smaller (54%). This can be explained by the fact that in this case, the positions of the FCP's are determined manually. As with the FaceShop Query, it is not certain what the actual performance of both queries is: finding one or more closest matches doesn't necessarily mean that the *correct* matches have been found. A more structured testing cycle could be performed to determine the exact performance of the ability of FED to label an unknown facial expression.

### *Incremental Query*

As mentioned earlier, it is not possible to determine the same statistics for the Incremental Query as with the other queries. In principle, it would be possible to log the number of times users have chosen to issue an Incremental Query.

One major problem with the Incremental Query is the fact that the measure of likeness, used to determine the clusters and corresponding pivot facial expressions from a set of facial expressions, is based on six facial features that can cancel each other out. The effects are that for the extreme situations (clusters 1 and 4), the algorithm works really well. If the user selects the first or the fourth cluster, all facial expressions are alike and clearly belong to the same cluster of facial expressions. For clusters 2 and 3 however, the resemblance is not so striking. However, it can be expected that the performance of the Incremental Query will increase as the dictionary becomes filled with more entries. The reason for this is that if there are more facial expressions, the bigger the chance that the pivot facial expressions of clusters 2 and 3 do resemble each other.

## Chapter 8: Conclusions and future work

*In the introduction, a number of implementation and research goals were set. This chapter will evaluate the extent to which these goals have been reached. Section 8.1 discusses the viability of the FED website implementation. In section 8.2, the usefulness of the FED corpus is discussed. Section 8.3 discusses the extent to which the individual queries into FED function correctly. Whether or not FED is a viable implementation of an online Facial Expression Dictionary in particular and of an online Nonverbal Dictionary in general is determined in section 8.4. Finally, section 8.5 describes possible future work on FED.*

### 8.1 Website architecture

The previous chapter discussed the performance of the FED website. It is clear that the website architecture and system design of FED form a valid approach for implementation of an online Facial Expression Dictionary: the entries in the dictionary are easily manageable and it is relatively easy to scale, adapt and extend FED. Also, the FED website is fairly secure: it will not be easy for malice users to corrupt the database.

There are however also several disadvantages to the FED website as it stands today. Issuing a query can take a relatively long time, depending on the performance of the user's computer and the speed and bandwidth of the available internet connection. With a high-speed internet connection and a relatively fast computer, the website speed is adequate: even queries that require a large number of calculations and return relatively large numbers of results have an acceptable response time of a few seconds (see table 7.1).

Also, when several users issue the same type of query simultaneously, an error can occur where all users get the same query results. The reason for this is that the query results are temporarily stored in a status object at the servlet and retrieved through a separate connection. In this way, a user can retrieve the query results of a query issued by another user. It proved impossible to send a query and retrieve the results directly through just one connection. This is caused by a bug in the Java Virtual Machine of Microsoft Internet Explorer 5.0 (IE5). This bug will be corrected in future releases of Microsoft Internet Explorer. Fortunately, it would take little effort to change the code that handles the data traffic between applets and servlets, so it is possible to correct this situation in the future. For the moment however, the problem exists and cannot be solved by upgrading the capacity of the Java servlets to handle more users simultaneously.

### 8.2 FED Corpus

All facial expressions of the FED corpus were labeled manually. This means that FED is based on *subjective* data: if other people were asked to sketch the same 56 facial expressions with FaceShop, the distribution of entries over the feature space could be completely different.

When trying to determine the viability of the implementation of certain query possibilities into FED, it generally doesn't matter if the facial expression labeled 'happiness' actually really represents 'happiness'. The viability of the Label, Action Unit, Geometry and Incremental Query can be tested independent of the extent to which the FED data is subjective or scientifically correct and agreed upon.

This is not the case however when the label of an unknown facial expression is determined through a FaceShop or a Picture Query. Because of the fact that certain facial expressions can be realized in a number of ways, it is possible that an unknown facial expression represents a certain facial expression, but is located relatively far away from the template of that facial expression in the feature space.

As mentioned in chapter 3, a possible solution to this problem could be to implement each entry in FED as a cluster of different realizations of a certain facial expression. This could be accomplished by letting several groups of users create a subjective database. The union or average over all these individual databases will provide a general database for FED. In that case, it would be possible to define a probability density function of a facial expression in the feature space. An important question will then be if the clusters overlap or if they are separated nicely.

### **8.3 Search modalities**

The Label, Action Unit and Geometry Query have proven to work adequately for the current version of FED. When FED is extended and adapted in the future, or the FED corpus is extended, certain modifications and improvements will have to be made to the implementations of these queries, but this will generally be easy and not take a long time.

A problem with the Incremental Query implementation is the fact that certain facial features can cancel each other out. Although this may not pose a problem when the FED corpus is sufficiently large, it may be prudent to consider an alternative method of determining a measure of likeness between two facial expressions.

Also, although the labeling of an unknown facial expression through a FaceShop or a Picture Query has proven to work quite well and although there is room for improvement when a FED entry is implemented as a cluster, it can be concluded that it is possible to successfully issue a nonverbal query into a multimodal database with FED.

### **8.4 FED : a viable approach?**

The FED website as it stands today can be considered a successful prototype of an online Facial Expression Dictionary. A proof of concept has been demonstrated. There are many advantages to the design chosen for FED. The system is scalable, extendible, adaptable and secure. A possible disadvantage is the speed of the website when a user doesn't have access to a high-speed internet connection.

As a first-order solution, the choice was made to represent each facial expression entry in FED through a template. This approach has proven to work: it allows users to issue a nonverbal query and let FED determine the label of an unknown facial expression. It is clear however that a more natural approach would be to extent FED to a second-order solution with each facial expression represented by a cluster. This would allow for a more accurate labeling of an unknown facial expression.

It seems possible to use the approach taken with FED to create a complete Nonverbal Dictionary, which would contain information about all the other ways people communicate with each other nonverbally besides facial expressions. Naturally, the database structure needed to store for example sound clips representing a certain tone of voice would be different, and issuing a nonverbal query would require completely different techniques than automatic facial expression recognition, but the same design principles could be applied.

### **8.5 Future work**

It is impossible to realize all theoretically possible functionality of an online Facial Expression Dictionary in the time that was available. This section gives a number possibilities for future work on FED. This includes additional functionality that could be added to FED, augmentation of the information associated with an entry in FED and possible adaptations of the conceptual design.

### *Clusters instead of templates*

As mentioned earlier, certain facial expressions can be realized in a number of ways / to a certain degree. One template representing a facial expression will not cover all possible realizations of that facial expression. A next logical step would thus be to associate a cluster of different realizations with each facial expression entry in FED. This would make more sophisticated classification of unknown facial expressions possible.

### *Facial Action Encoder*

Behavioral science researchers often use active Action Units to represent facial expressions. For this reason, each facial expression entry in FED has a number of active AU's associated with it. At the moment, a FED administrator has to determine the active AU's of a facial expression manually. This can quite easily lead to errors and inconsistencies.

In principle, it is possible to determine the active AU's of a facial expression automatically, using the coordinates of the FCP's. Furthermore, it would then be possible to determine the extent to which an AU is active, instead of just determining if it is present or not. If such a so-called *facial action encoder* is implemented, the scientific validity and thus the usability for behavioral science researchers would increase dramatically.

Also, it would be prudent to extend the number of possible AU's that can be associated with a facial expression from 30 to 44. This would mean that the implementation of the FaceShop tool would have to be adapted, or a different facial expression generation tool would have to be used. In that case, as with FaceShop, the coordinates of the FCP's of the face model of Kobayashi and Hara would have to be determined automatically.

### *3D faces instead of 2D faces*

Naturally, a 3D image of a facial expression would contain more information than a 2D image such as created with FaceShop. It would thus be nice to introduce this concept in FED as well.

### *Fill the FED database with interpretation data from the field of psychology*

As mentioned in the introduction, it is difficult to obtain a complete and scientifically valid description of a facial expression. At the moment, only a small number of facial expressions contain a complete and scientifically valid interpretation. It would be desirable to augment each FED entry with a complete and scientifically valid interpretation.

### *Extend the concept of a FED entry*

The current definition of an entry in FED is incomplete. Section 3.1 gave a description of all the possible information that can be associated with an entry in a Facial Expression Dictionary. The usefulness of FED can be improved if an entry in FED is extended to this type of entry.

### *Fully automatic facial expression labeling*

The FED system allows users to determine the label of a facial expression shown in a picture. This is accomplished semi-automatically: the user has to select a subpart of the picture containing the face and determine the positions of the 30 FCP's manually.

If the face detection and feature extraction step could be performed fully automatically, the user friendliness and performance of this type of FED query could be improved. Image processing

techniques can be used to automatically determine the location of the face. The selected subpart could be displayed as a suggestion to the user. Also, the FCP's could be positioned automatically and displayed to the user, who could then manually adjust the FCP's that he thinks are not positioned correctly.

This could save a user time, and there is no error introduced in the position of the FCP's due to incorrect positioning by the user. Of course, the algorithm that determines the positions of the FCP's would introduce an error too, but this method is still expected to be more robust.

#### *Complete the FED database*

As mentioned earlier, there exist approximately 7000 facial expressions. It would of course be nice if all of these were present in the FED database.

#### *Individual rights for each Admin user*

At the moment, all users of the FED admin website have identical rights. This means that each user is allowed to view, add, edit and delete FED entries and FED admin users, and have the right to view both the Admin and Query log. It could be desirable to give each user specific rights as to what actions they are allowed to perform. For example, if someone is interested in the FED system and wants to know more about it's implementation, a new FED user with limited rights could be created. This user could for example only be allowed to view all FED entries and the Query Log.

#### *Allow FED admin users to browse through successfully classified FaceShop sketches and uploaded pictures*

All FaceShop sketches and uploaded pictures that are successfully labeled are stored in the FED database. It would be useful if an authenticated user was able to view these entries. This could give additional insight into the behavior and performance of the Labeling QPM.

#### *Netscape compatible*

The FED website can be viewed by users using a web browser. At the moment, the vast majority of Internet users (approximately 90%) use Microsoft Internet Explorer (IE) as their web browser. There are however a number of other browser programs available, of which Netscape is the most widely used.

When implementing a website, it is always a challenge to ensure that it is both IE and Netscape compatible. There are small differences in implementation that can cause relatively big differences in the appearance and/or workings of a website. In the case of FED, their proved to be a problem with making the website Netscape compatible. There is a bug in Netscape, which makes it impossible to use Java Servlet technology: the HTTP headers are not written to the servlet. There is no workaround, and the bug will not be fixed in the future.

Ideally, FED would be made Netscape compatible. However, this could mean that the complete website design and implementation would have to be revised.

## Bibliography

Baron, R.A., Byrne, D. and Johnson, B. T. (1998, fourth edition) *Exploring Social Psychology*. Allyn and Bacon, Boston, Massachusetts, USA.

Birnam, S. (2001) *Distributed Java 2 platform database development*. Sun Microsystems Press, Palo Alto, California, USA.

Ekman, P., Friesen, W. (1978) *Facial Action Coding System*. Consulting Psychologists Press, Inc., Palo Alto California, USA.

Ekman, P., Friesen, W. (1975) *Unmasking the Face*. Englewood Cliffs, New Jersey, USA, Prentice-Hall.

Facial Expression Homepage (2002) [http://www.mis.atr.co.jp/~mlyons/facial\\_expression.html](http://www.mis.atr.co.jp/~mlyons/facial_expression.html)

Filip, J., Holoubek, L. (2000) Automatic recognition of facial expressions in linedrawings. Project report, Delft University of Technology, the Netherlands.

Hara, F., Kobayashi, H. (1993) *A Basic Study of Dynamic Recognition of Human Facial Expressions*. Proceedings of IEEE International Workshop on Robot and Human Communication, pp.271-275 (1993-11).

Hara, F., Kobayashi, H., Tange A. (1995) *Real-Time recognition of six basic facial expressions*. Proceedings of IEEE International Workshop on Robot and Human Communication, pp.179-186 (1995-07).

Huang, T. S., Zhou, X.S. (November 2000) *A generalized relevance feedback scheme for image retrieval*. Proceedings of SPIE Vol. 4210: Internet Multimedia Management Systems, Boston, Massachusetts, USA.

Huang, T. S., Zhou, X.S. (2002) *Relevance feedback for image retrieval: a comprehensive review*. ACM Multimedia Systems Journal, special issue on CBIR.

Iyengar, P.A., Samal A. (1992) *Automatic recognition and analysis of human faces and facial expressions: a survey*. Pattern Recognition, vol. 25, no. 1, pp 65-77, 1992.

Jain, L.C., Halici, U., Hayashi, I., Lee, S.B., Tsutsui, S. (1999) *Intelligent biometric techniques in fingerprint and face recognition*. CRC Press, Boca Raton, Florida, USA.

Java Developer Connection (2002) <http://www.developer.java.sun.com>

Java Servlet API (1998)

<http://developer.java.sun.com/developer/onlineTraining/Servlets/Fundamentals/servlets.html#CGIReplacements>

JDK 1.2 API Specification (1998) <http://java.sun.com/products/jdk/1.2/docs/api/index.html>

Leippe, M.R., Zimbardo, P. G. (1991) *The psychology of attitude change and social influence*. McGraw-Hill, New York, USA.

Leung, M.Y.Y., Hui, H.Y., King, I. (1996) *Facial expression synthesis by radial basis function network and image warping*. IEEE International Conference on Neural Networks, volume III, pages 1400-1405, Washington D.C., IEEE Computer Society.

Matsumoto, D. (2000, second edition) *Culture and Psychology*. Oxford University Press, New York, USA.

Ostermann, J., Murat Tekalp, (2000) *A. Face and 2-D Mesh Animation in MPEG-4*. Image Communication Journal, Tutorial Issue on MPEG-4 Standard, Elsevier.

Pantic, M. (2001) *ISFER, Facial Expression Recognition by Computational Intelligence Techniques*. Ph.d thesis, Delft University of Technology, the Netherlands.

PostgreSQL (1995) [http://www.ca.postgresql.org/docs/aw\\_pgsql\\_book/index.html](http://www.ca.postgresql.org/docs/aw_pgsql_book/index.html)

Rodrigues, L. H. (2001, first edition) *Building Imaging Applications with Java Technology*. Addison-Wesley, Boston, Massachusetts, USA.

Rothkrantz, L.J.M., Wojdel, A. (2000) *A Text Based Talking Face*. Proc. of the third Int. Workshop on Text, Speech and Dialogue, Brno, Czech Republic.

Sridharan, P. (1997) *Advanced Java Networking*. Prentice Hall, Upper Saddle River, New Jersey, USA.

Tanenbaum, A.S. (1997 fourth edition) *Computer Networks*. Prentice Hall, Upper Saddle River, New Jersey, USA.

Terzopoulos, D. Waters, K. (1991) *Techniques for realistic facial modeling and animation*. Computer Animation '91, pages 45-58, Springer-Verlag, New York, USA.

Typarse (1998) <http://www.netexpress.net/~tyarker/TYParse.html>

World Wide Web Consortium (W3C) <http://www.w3.org/>

Wojdel, A. (1999) *Overview of literature about facial animation*.

## Appendix A: Overview of Java classes

### Package onvd.data

Contains various data structures that are used to send data between the client and server and provide additional functionality / operations on the data

**Table A1**  
Classes belonging to package onvd.data

Class	Function
ONVDEntry	Superclass for all three kinds of entries in FED: templates, the results of successful FaceShop queries and the results of successful Picture Queries
TEEntry	Subclass of ONVDEntry. Contains all information associated with a template facial expression.
FSEntry	Subclass of ONVDEntry. Contains all information associated with a successfully classified FaceShop sketch.
PUEntry	Subclass of ONVDEntry. Contains all information associated with a successfully classified facial expression shown in an uploaded picture.
Interpretation	Represents the interpretation associated with a template facial expression, FaceShop query result or Picture Query result.
AUEntry	Contains all information of an Action Unit.
UserEntry	Contains all information of a FED admin user.
AdminLogEntry	Represents an Admin Log entry.
QueryLogEntry	Represents a Query Log entry.
GeometryQueryResults	Represents the results of a Geometry Query
INCQuerySubspace	Represents an Incremental Query Subspace
KeywordQueryResults	Represents the results of a Label Query

### Package onvd.postgresql

Contains classes needed to access the PostGreSQL database and perform operations on the data.

**Table A2**  
Classes belonging to package onvd.postgresql

Class	Function
DBConnection	Provides a connection to the PostGreSQL database
PostgresToJava	Converts data from a PostGreSQL representation to a representation through Java classes from package onvd.data
JavaToPostgres	Converts data from a representation through Java classes from package onvd.data to a PostGreSQL representation



## Package onvd.servlets

Contains a class ServletFunctions that is used by the Java servlets of the FED website for logging purposes and contains a function that returns a connection to a specific Java servlet.

**Table A3**  
Classes belonging to package onvd.postgresql

Class	Function
ServletFunctions	Provides miscellaneous functions used by Java servlets of the FED website; provides connection to specified servlet

## Package onvd.kobayashihara

Contains classes needed implement the face model of Kobayashi and Hara.

**Table A4**  
Classes belonging to package onvd.kobayashara

Class	Function
FCS	Represents an instance of the coordinate system of the face model of Kobayashi and Hara
FCP	Represents a Facial Characteristic Point
FCPStack	Represents a stack of FCP's
HaraLine	Represents a Haraline
HLStack	Represents a stack of Haralines
ValidArea	Represents an area in which an FCP can be placed and/or moved freely
IntelligentChecks	Provides the functionality to check if a certain FCP is contained in a for that FCP valid area
FacialDistances	Compresses the 30 FCP's as 21 facial distances
Point2Ddouble	Represents a 2D point with double precision

## Package onvd.vdi

Contains classes needed to implement the Visual Data Inspection tool

**Table A5**  
Classes belonging to package onvd.vdi

Class	Function
VDIScatterPlot	Extension of the Java awt.Canvas class; given two datasets, the origin of the scatterplot is calculated and the points are displayed on the canvas
VDIPoint	Represents a point to be displayed on a VDIScatterPlot canvas
VDIFeatureCalculation	Calculates the value of a certain facial feature
TYParse	Arithmetic expression parser, used by class VDIFeatureCalculation
VDIFeatureFormatException	Exception thrown by class VDIFeatureCalculation if a feature is not a valid arithmetic expression

## Package onvd.graphics

Contains several Java classes that implement a certain part of the GUI of FED, or provide functions for image manipulation

**Table A6**  
Classes belonging to package onvd.graphics

Class	Function
ImageCanvas	Modified version of the Java awt.Canvas class with additional functionality for manipulating the image displayed
PictureData	Contains all information of a picture that is to be displayed on an ImageCanvas
FaceShopImage	Stripped version of the FaceShop tool; contains only the image of a facial expression
TransparantLabel	Modified version of the Java awt.Label class; appears transparent
ColorConvert	Converts HTML color codes to a Java awt.Color representation and vice versa
DateField	Implements a date GUI component that allows a user to select a date

## Package onvd.queryprocessing

Contains the classes that process query requests

**Table A7**  
Classes belonging to package onvd.queryprocessing

Class	Function
KeywordQuery	Processes a Label Query
AUQuery	Processes an Action Unit Query
GeometryQuery	Processes a Geometry Query
GeometryQueryEncoder	Encodes a Geometry Query
GeometryQueryParser	Parses a Geometry Query
GeometryQueryScanner	Reads a Geometry Query character for character
GeometryQueryFunctions	Retrieves all FED entries with a certain geometrical feature present
GeometryQueryToken	All characters that can occur within a Geometry Query string
GeometryQueryEncoderErrorException	Exception thrown by the GeometryQueryEncoder class if encoding of the query fails
GeometryQueryEncoderErrorCorrectedException	Exception thrown by the GeometryQueryEncoder class if encoding of the query fails, but a spelling suggestion was generated
GeometryQueryScannerException	Exception thrown by the GeometryQueryScanner class if the entered query is in an illegal format
SpellingChecker	Checks to see if a string matches another string when allowing for one insertion, deletion or substitution
IncrementalQuery	Given a subspace location string, returns the corresponding Incremental Query Subspace
FacialExpressionClassification	Determines the label of an unknown facial expression

## Java Servlets

Servlet	Function
AdminAddTEServlet	Stores a new template expression in the database
AdminEditTEServlet	Updates a template expression in the database
AdminDeleteTEServlet	Deletes a template expression from the database
AdminGetAllTEServlet	Retrieves all template expressions from the database
AdminGetAllActiveTEServlet	Retrieves all active template expressions from the database
AdminGetAllAUServlet	Retrieves all Action Units from the database
AdminGetAllTEPicturesServlet	Retrieves the name and location on the server of all template expression pictures
AdminGetAllUsersServlet	Retrieves all FED admin users from the database
AdminManageUsersServlet	Adds, edits or deletes a FED admin user
AdminGetQueryLogServlet	Retrieves the Query Log
AdminGetAdminLogServlet	Retrieves the Admin Log
QPKeywordServlet	Receives a Label query and returns the results
QPActiveAUServlet	Receives a Action Unit query and returns the results
QPGeometryServlet	Receives a Geometry query and returns the results
QPIncrementalServlet	Receives a subspace location string and returns the next Incremental Query Subspace
QPFaceShopServlet	Receives the coordinates of the FCP's of a facial expression sketched with FaceShop and returns the classification results
QPPictureUploadServlet	Receives the coordinates of the FCP's of a facial expression shown in an unloaded picture and returns the classification results
QIPictureUploadServlet	Uploads a picture selected by the user to the server and initializes the Picture Query
AdminLoginServlet	Verifies the username and password needed to access the FED administrative backend website
AdminHTMLGeneratorServlet	Generates the HTML pages that comprise the FED administrative backend website
AdminWriteTETToFileServlet	Writes the coordinates of the FCP's of all active template facial expressions to a text file

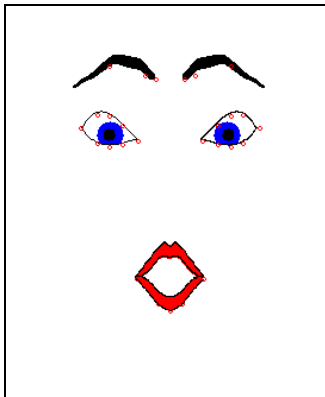
## Java classes used to implement FED management functions

Management function	Clientside Java classes used to implement GUI	Servlet(s)	Serverside Java classes
FED entry management	AdminTEApplet AdminTEMainPanel AdminTEAddPanel AdminTEEditPanel AdminTEDeletePanel AdminTEAddAUDialog	AdminAddTEServlet AdminEditTEServlet AdminDeleteTEServlet AdminGetAllTEServlet AdminGetAllActiveTEServlet AdminGetAllAUServlet AdminGetAllTEPicturesServlet	PostgresToJava JavaToPostgres
FED user management	AdminUserApplet AdminUserAddPanel AdminUserEditPanel	AdminManageUsersServlet	PostgresToJava JavaToPostgres
Query log	AdminQueryLogApplet	AdminGetQueryLogServlet	PostgresToJava JavaToPostgres
Admin log	AdminLogApplet	AdminGetAdminLogServlet	PostgresToJava JavaToPostgres
Visual Data Inspection	AdminVDIApplet AdminVDIScatterPlotPanel VDIScatterPlot VDIPoint VDIFeatureCalculation TyParse VDIFeatureFormatException	-	-

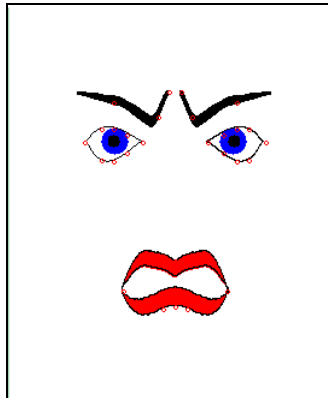
## Java classes used to implement queries into FED

Query	Clientside Java classes used to implement the GUI	Servlet	Serverside Java classes used for processing
Label	KWQApplet ShowQueryResultsPanel	QPKeywordServlet	KeywordQuery SpellingChecker
Action Unit	AUQApplet AUQSelectAUPanel ShowQueryResultsPanel	QPActiveAUServlet	AUQuery
Geometry	GEOMQApplet ShowQueryResultsPanel	QPGeometryServlet	GeometryQuery GeometryQueryEncoder GeometryQueryParser GeometryQueryScanner GeometryQueryFunctions GeometryQueryToken GeometryQueryEncoder- Exception GeometryQueryEncoder- ErrorCorrectedException GeometryQueryScanner- Exception SpellingChecker
Incremental	INCQApplet ShowQueryResultsPanel	QPIncrementalServlet	IncrementalQuery
FaceShop	FSQApplet FaceShopPanel ShowClassification- ResultsPanel	QPFaceShopServlet	FacialExpression- Classification
Picture	PUQApplet PUQFaceDetectionPanel PUQFacialFeatures- ExtractionPanel PUQShowClassification- ResultsPanel ShowClassification- ResultsPanel	QPPictureUploadServlet	FacialExpression- Classification

## Appendix B: FED corpus



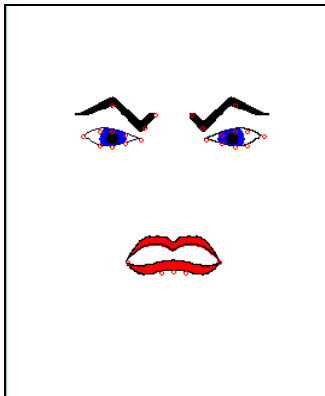
*Admiration*



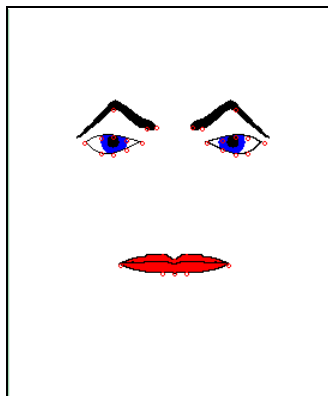
*Aggression*



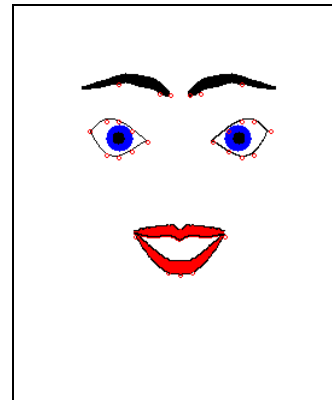
*Agony*



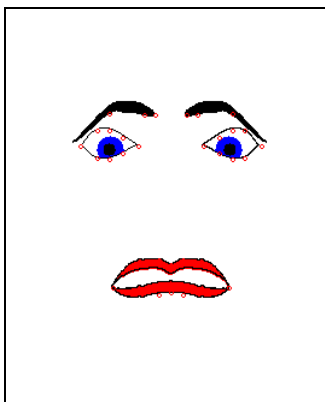
*Anger*



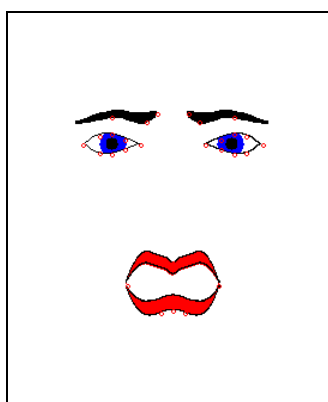
*Annoyed*



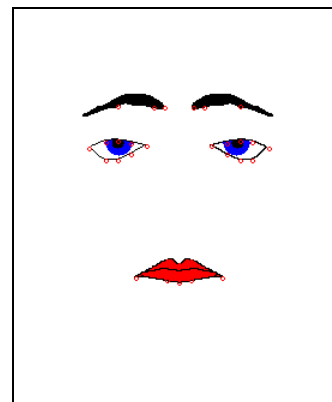
*Anticipating*



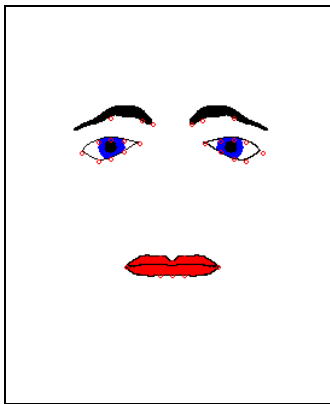
*Anxiety*



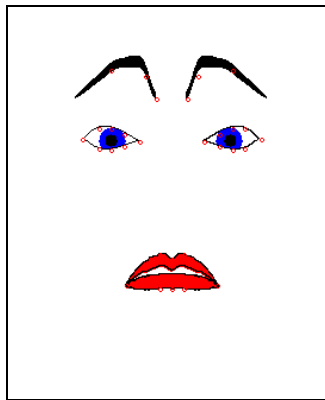
*Appalled*



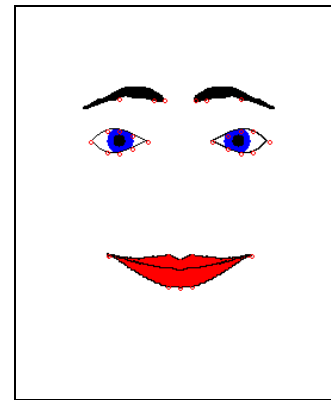
*Arrogance*



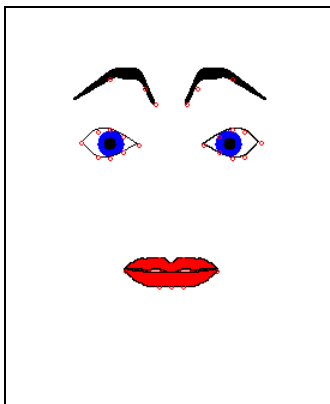
*Bored*



*Cautious*



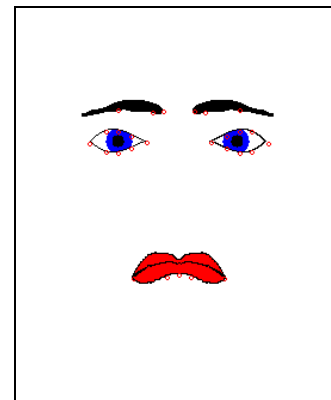
*Cheerful*



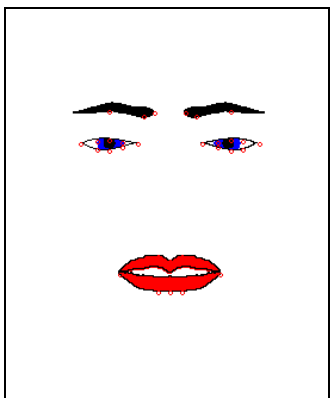
*Clueless*



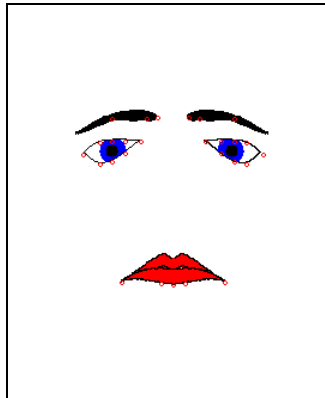
*Concentration*



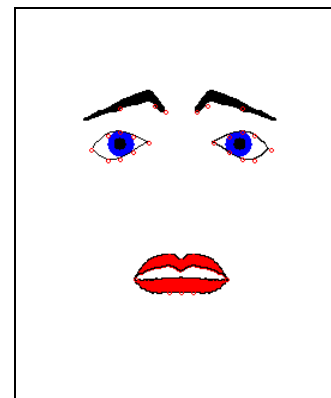
*Contempt*



*Dangerous*



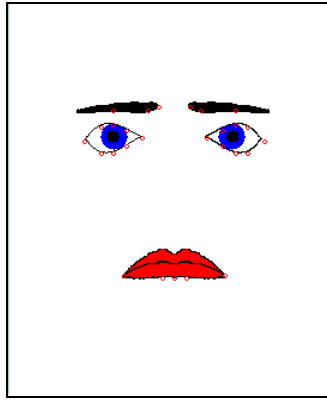
*Defeated*



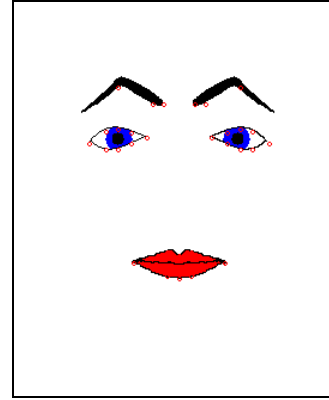
*Desperation*



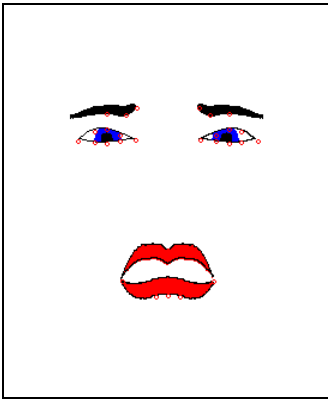
*Determination*



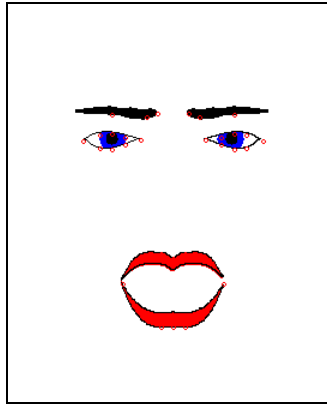
*Disappointment*



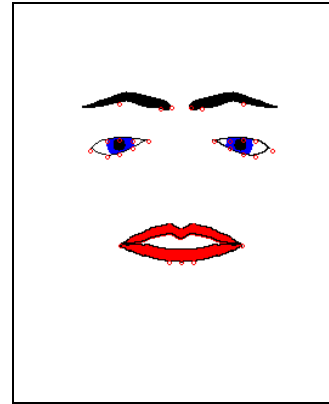
*Disbelief*



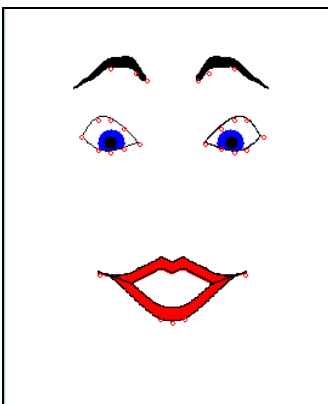
*Disgust*



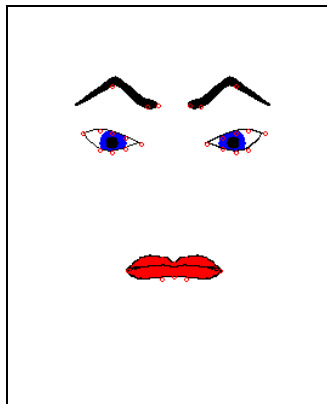
*Domination*



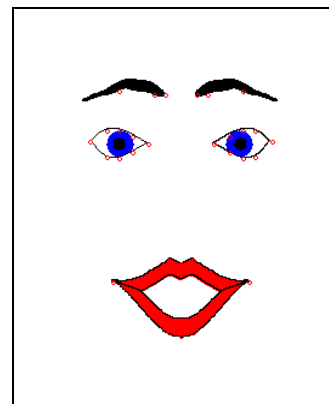
*Drunk*



*Elated*

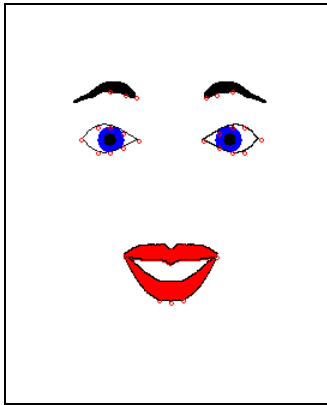


*Envy*



*Euphoria*

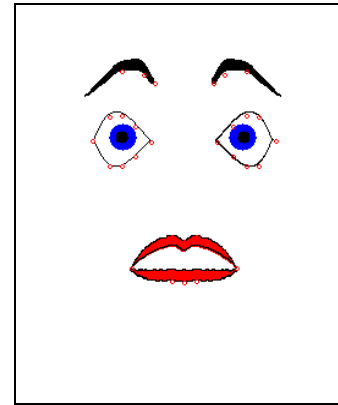




*Excitement*



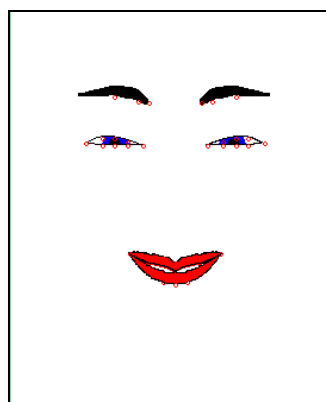
*Exhausted*



*Fear*



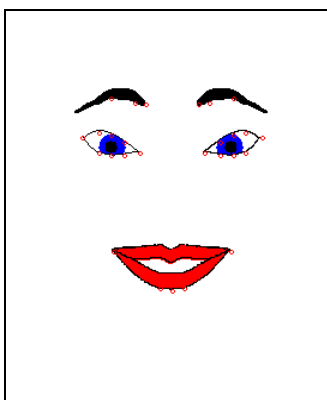
*Fierce*



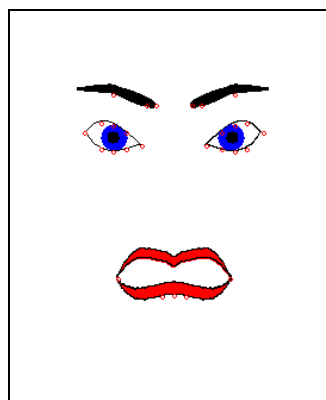
*Giggle*



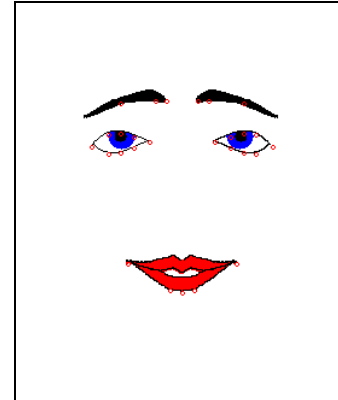
*Grieving*



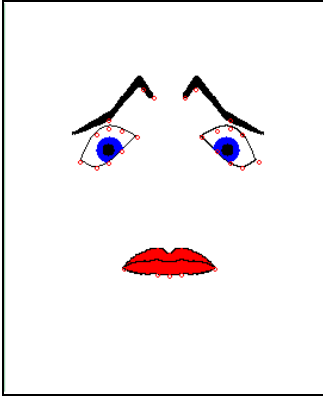
*Happiness*



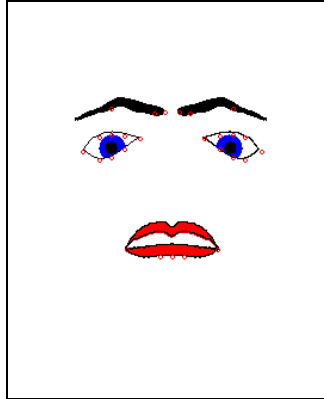
*Horror*



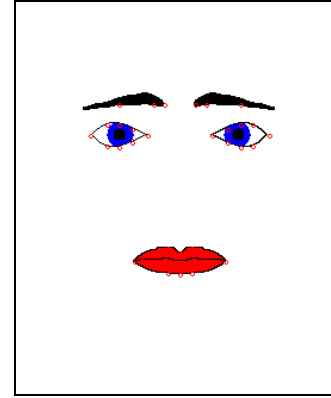
*In love*



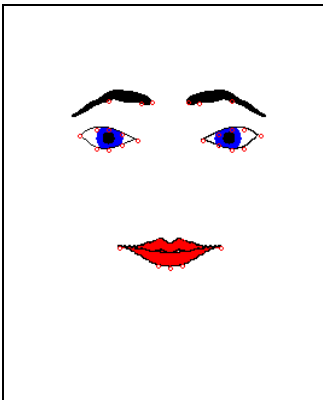
*Moved*



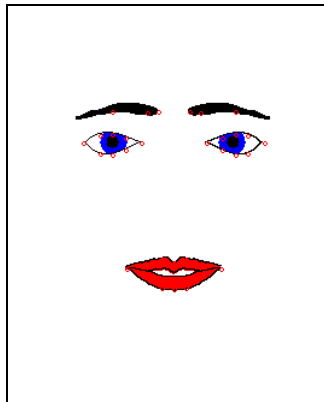
*Nausea*



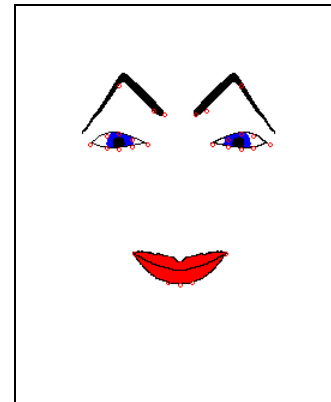
*Neutral*



*Peaceful*



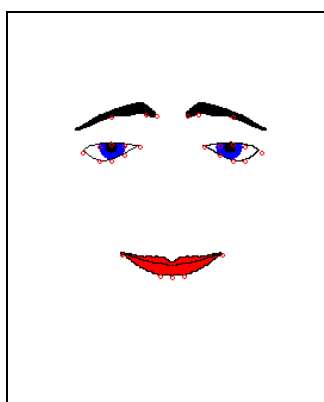
*Relaxed*



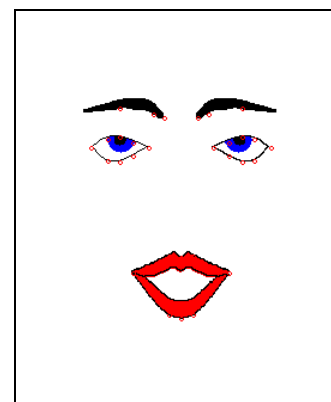
*Sadistic*



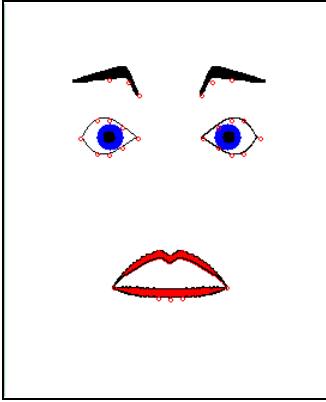
*Sadness*



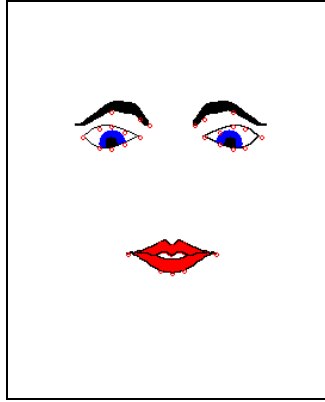
*Satisfaction*



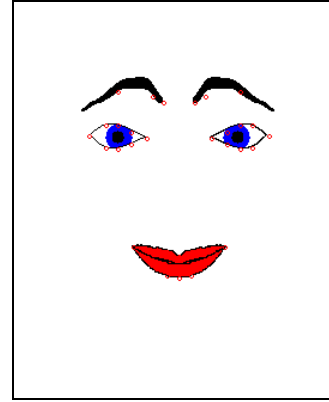
*Seductive*



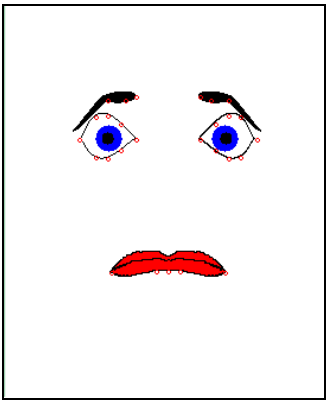
*Shocked*



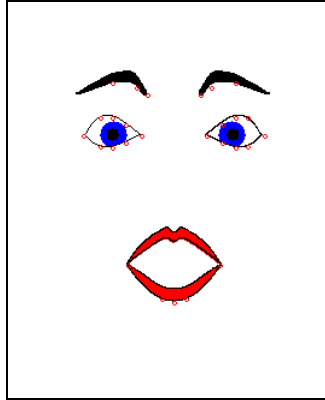
*Shy*



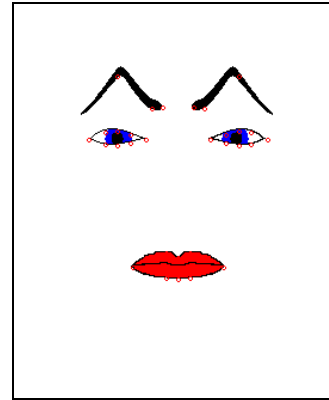
*Smiling*



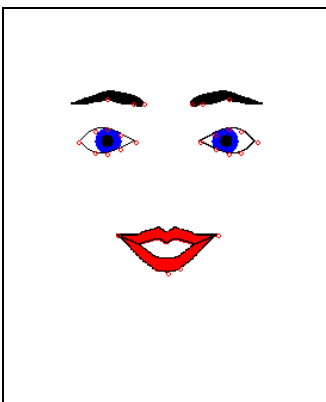
*Stress*



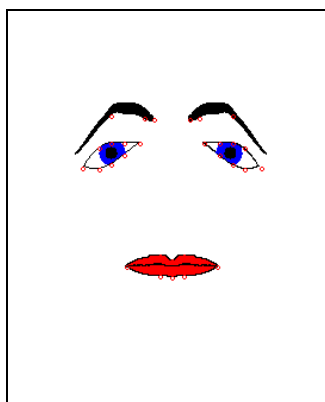
*Surprise*



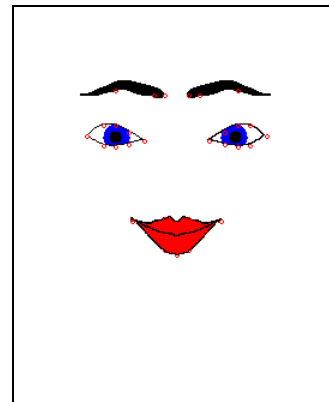
*Suspicion*



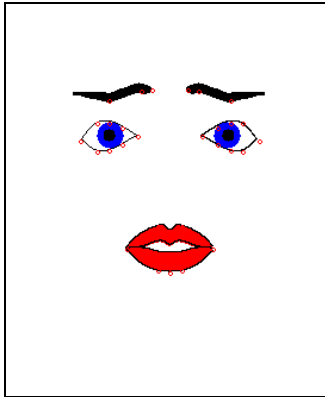
*Sweet*



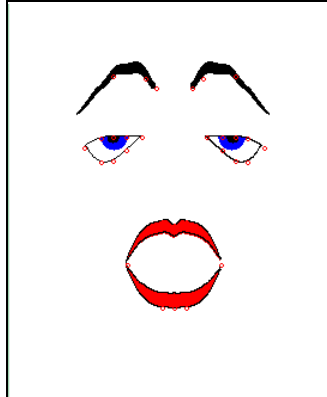
*Tired*



*Triumphant*



*Worried*



*Yawn*

## Appendix C: Paper

### FED – an online Facial Expression Dictionary

E.J. de Jongh    L.J.M. Rothkrantz

Delft University of Technology, Mekelweg 4, 2628 CD Delft

#### Abstract

People communicate with each other through spoken words and nonverbal behavior. Verbal communication is used to convey objective information, whereas nonverbal communication is used to convey subjective and affective information. Due to a number of reasons, confusion and misunderstandings can arise when people communicate with each other. A verbal dictionary can be used to look up the meaning and spelling of a certain word and can help people communicate verbally. The goal of this research was to develop an online Facial Expression Dictionary as a first step in the creation of an online Nonverbal Dictionary. A Nonverbal Dictionary would have the same functionality as a verbal dictionary and could help people communicate nonverbally.

## 1. Introduction

Human communication is based on verbal and nonverbal behaviour. It is commonly assumed that natural language is used to communicate objective information to other people and nonverbal behaviour is used to convey subjective and affective information. Speech has a verbal and a nonverbal aspect. It is more appropriate to speak about the denotative and connotative aspect of multi-modal communication. The denotative aspect is based on the grammar and the connotative aspect is based on the rules of communication.

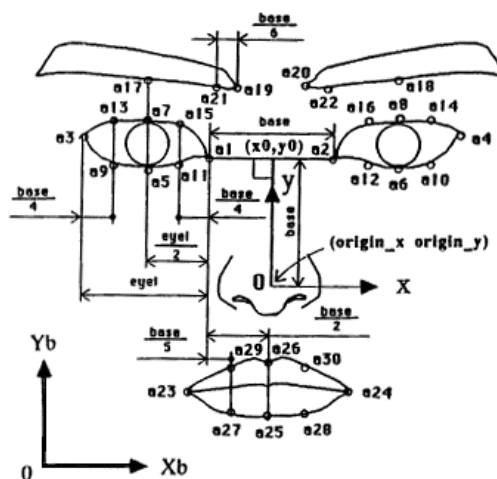
In [1] P.Ekman introduced the concepts emblems and emotional emblems. The last ones are expressed by employing parts of the corresponding affects they refer to, while the first ones are used to replace and repeat verbal elements. Most of the time both are intentional, deliberate actions used to communicate. In general, they are produced consciously and are driven by the semantics of the utterance. They are conventionalised. Since they are discourse driven, the user enters their appearance. What is needed is a library of possible emblems. Efron gave a large list of them [2] and Ekman proposes a set of words, which have a corresponding emblem. Nevertheless the user can build his/her own emblem and add them to the library [3]. This paper describes the development of such a library: an online facial expression dictionary or FED for short. FED could be extended to a complete Nonverbal Dictionary [5], which would contain information about all the ways people communicate with each other nonverbally.

## 2. Related Work

In 1970, P.Ekman and F.W.Friesen [4] developed an universal Facial Action Coding System (FACS) based on Action Units (AU's). Each facial expression can be described in terms of active AU's. An AU is said to be active if the corresponding face muscle is flexed. FACS is used in FED to define a facial expression analogous to how a word in a verbal dictionary is defined by using characters.

Also, an essential feature of a nonverbal dictionary is that it is possible to issue a nonverbal query. For FED, this means that it has to be possible to issue a query using a facial expression itself. This could be accomplished by either letting users supply a picture containing a facial expression, or by letting users sketch a facial expression online by using a facial expression generation tool. Subsequently, FED would then determine the label of the facial expression. Implementing this type of query requires techniques from the fields of automatic facial expression recognition and multi-modal information retrieval.

Each facial expression stored in FED will have to be represented in a way that enables the implementation of automatic facial expression recognition. The face model developed by Kobayashi and Hara [6] was used to represent a facial expression in FED. This model is based on 30 so-called facial characteristic points (FCP's), which describe the shape of the eyebrows, eyes and mouth (see figure 1).



**Figure 1: The Face Model developed by Kobayashi and Hara**

## 3. FED – an Online Facial Expression Dictionary

A corpus-based approach was chosen to create FED. Ideally, FED would contain a 24-hour view of the facial expressions of all the people on the planet. Since this clearly is unattainable, a simpler approach had to be taken. The basis for all entries in FED is a picture of a facial expression. This picture is either a picture of a facial expression taken from the real world or a picture generated by a facial expression generation tool. All FED entries will be stored in a database.

Analogous to an online verbal dictionary, users will have the possibility to issue various queries into FED online. Furthermore, a FED administrator will have the possibility to manage the entries in FED. These management facilities can be used to create the FED corpus. Finally, FED will be set up in such a way that enables easy adaptations and extensions of FED in the future.

There are several problems that need to be considered when trying to implement a facial expression dictionary:

As mentioned in the previous section, not all facial expressions are universal. The meaning of a facial expression can differ per culture and context.

Not every possible facial expression can be labeled / has a meaning.

People can display a mixture of several facial expressions.

Facial expressions are not always displayed to the same degree.

Certain facial expressions can be displayed in a number of different ways. For example, the facial expression indicating 'happiness' can be displayed in a number of different ways.

## 4. Design

FED has been implemented as a website that handles query requests via a client-server architecture. Figure 2 shows the global design of the FED system. The individual components of the FED system can be described as follows:

- The *FED main website* enables users to issue queries into FED. It consists of static HTML pages and Java applets. The applets implement the GUI for issuing a query into FED. For each query, there exists an applet that implements the GUI for that query.
- The *communication layer* of the FED system resides on the server and handles all data traffic between the client and the server. The communication layer consists of a collection of Java servlets.
- The *Query Processing Module* or *QPM* of the FED system also resides on the server and consists of several modules, each of which has the ability to process a specific type of query. Each module is implemented through one or more static Java classes.
- The *FED admin website* provides the GUI for the management part of the FED system. Like the main website, it consists of static HTML pages and Java applets. This website is only accessible through user authentication.
- The *Admin Processing Module* or *APM* implements the functionality needed to manage the FED system. Like the QPM, the APM consists of several modules that in turn consist of one or more static Java classes.
- The *FED database* contains all the entries in the dictionary, admin user information, and log info. The PostgreSQL database management system is used to implement the database.

There are several advantages to this setup. Firstly, it ensures that the time to process a query is independent of the capacity of the computer of the user, because all calculations involved in processing a query are performed at the server, which is a high performance computer. Secondly, this setup increases the security of the website because all database access takes place at the server. A third advantage is that by separating the GUI, communication and processing modules, the FED

system becomes more easily adaptable and extendible. If all query processing, communication and GUI code is located in one applet, the code can become very complex and thus difficult to adapt and/or extend. Finally, by using Java servlets in the communication layer, it becomes easy to upgrade the capacity of FED, i.e. the number of simultaneous users the website can handle. This is possible through increasing the amount of resources available to the servlets.

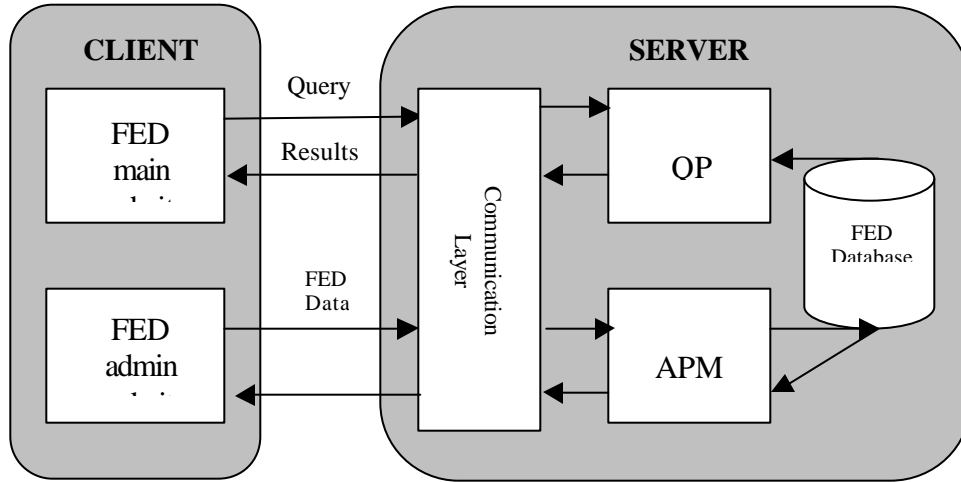


Figure 2: FED global design

## 5. FED entry implementation: Clusters vs. Templates

With FED, it is possible to let the system determine the label of an unknown facial expression shown in a picture or the label of a facial expression sketched online. This is accomplished by comparing the FCP coordinates of the unknown facial expression to the FCP coordinates of the FED entries. The closest match determines the label of the unknown facial expression. The performance of these queries depends on the way a facial expression is represented in FED.

One way of representing a facial expression entry is through a cluster of different realizations of that facial expression. Each realization has its own FCP coordinates and active AU's. Another possibility is to represent a facial expression entry by just one template expression. The latter approach can be described as a first-order solution, whereas the first approach can be seen as an extension of this method or a second-order solution.

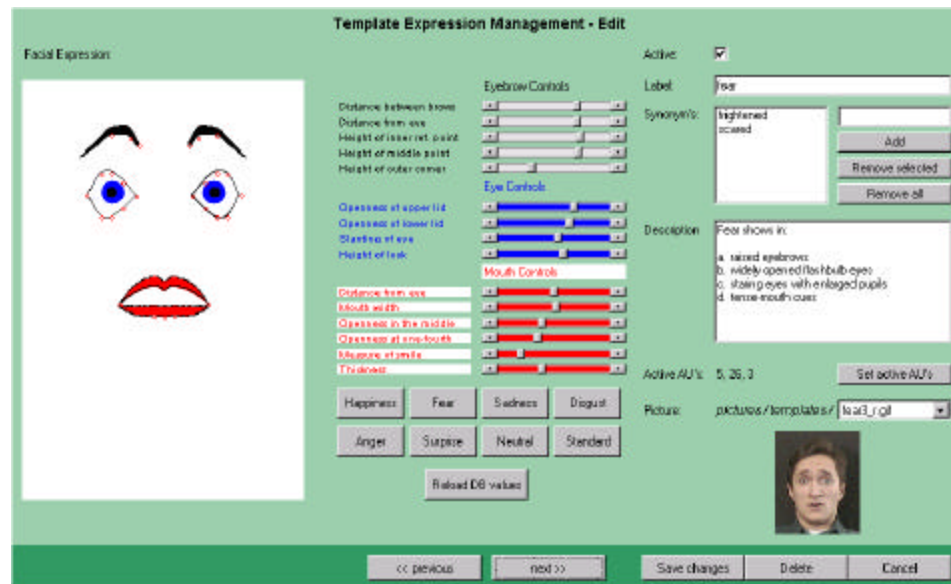
As a first approach, the choice was made to represent each facial expression by just one template. Given the fact that this will not cover all possible realizations of that facial expression, this is likely to affect the performance of FED queries to a certain extent. Implementing each entry through a cluster however would mean that there would be no time to implement all the basic functionality of FED.



## 6. Data acquisition

Research has shown that there exist approximately 7000 different facial expressions (Ekman). The number of possible combinations of 44 action units is of course much larger, but not every combination constitutes an expression. Some action units are mutually exclusive, and other combinations form expressions that do not have any meaning. Ideally, FED would contain all 7000 possible facial expressions, each with a scientifically valid interpretation. The primary goal of this project however is to develop a prototype and to show FED is a viable concept of a facial expression dictionary.

For each entry in FED, the coordinates of the FCP's have to be determined. This could be accomplished manually, but this can be a tedious and time-consuming process. Instead, FED makes use of a facial expression generation tool called FaceShop. This tool enables a user to sketch a facial expression by changing the position of a number of sliders. FaceShop calculates the coordinates of the FCP's automatically. Other information associated with an entry in FED is the facial expression label, label synonyms, description, and example picture. Also associated with each entry are the active Action Units of the facial expression.



**Figure 3: editing a FED entry**

FED entries can be added, edited and deleted by authenticated users at the administrative backend website of the FED system. Figure 3 shows a screenshot of the FED entry edit screen. The active AU's have to be set manually by the user, who can select them from a popup screen that appears when the 'Set active AU's' button is clicked. Not all 44 AU's can be selected, because at the moment FaceShop is unable to model all AU's.

## 7. Search Modalities

Analogous to a verbal dictionary, it is also possible to issue certain queries into FED. The representation of FED entries through FCP's enables the implementation of a number of additional queries, not present with a verbal dictionary.

### **Label Query**

As with a verbal dictionary, it is possible to issue a query on a keyword. In the case of FED, the entered keyword is matched against the facial expression label, label synonyms and words in the description of entries in FED.

### **Query on active AU's**

With this query, the user selects the action units that are active in the facial expression(s) he is looking for from a list of AU's.

### **Query on geometrical features**

The coordinates of the FCP's can be used to let users issue a query for facial expressions with certain geometrical features of the eyebrows, eyes and/or mouth present. Examples of valid queries are 'mouth open', 'eyebrows raised' and 'eyes slanting upwards'. Using the logical AND and OR operators, it is also possible to issue an aggregate query. Further restrictions can be imposed on a query by using the bracket operators '(' and ')'. An example of an aggregate query is 'eyes slanting upwards OR (mouth open AND eyebrows raised)'.

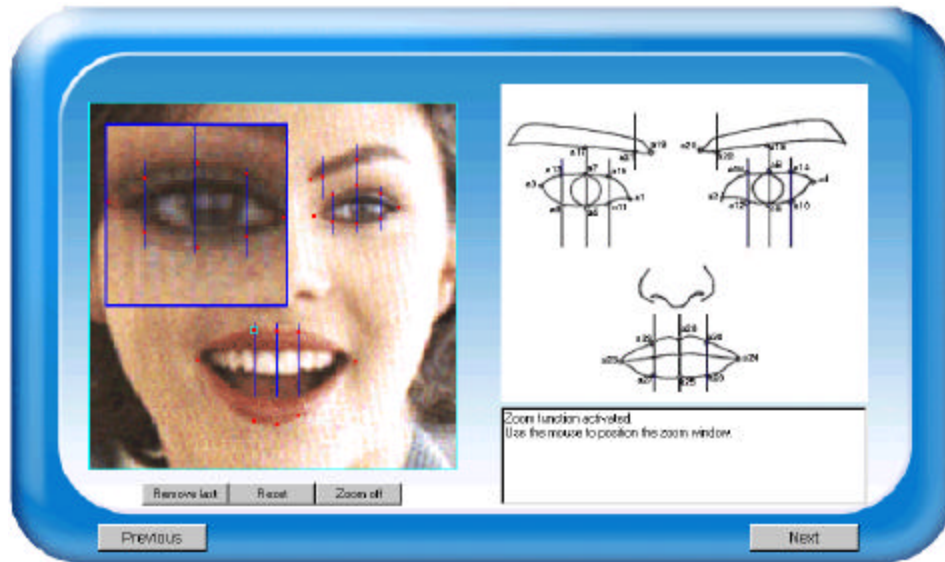
### **Incremental Query**

It is possible to look up a certain facial expression incrementally. With this query, the user is presented with four facial expressions as representatives of 4 clusters covering the whole dictionary. The user then selects the best fitting picture. The selected cluster is again divided into 4 clusters and corresponding representative facial expressions that are again presented to the user, who again has to select the best match.

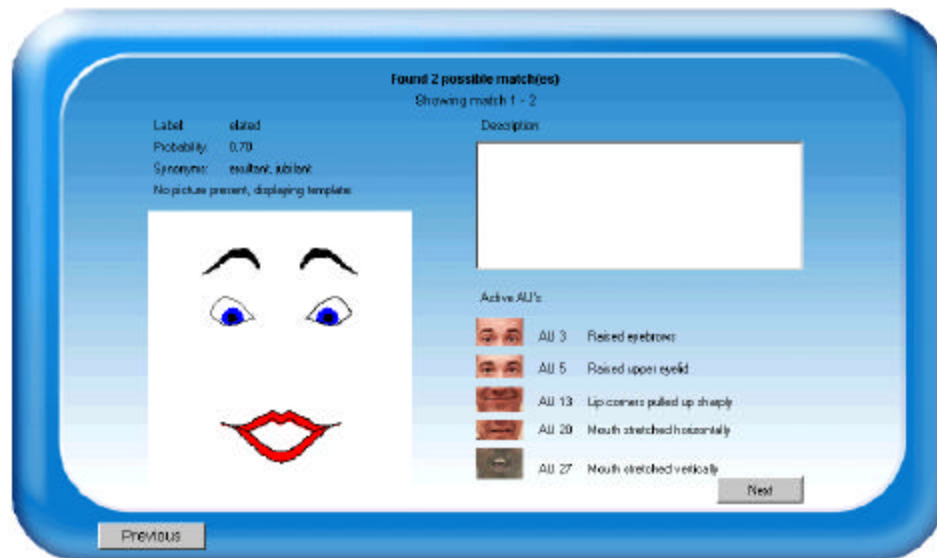
### **Labeling of an unknown facial expression**

Users can determine the label of an unknown facial expression in two ways. First of all, it is possible to determine the facial expression shown in a picture supplied by the user. With this type of query, the user determines the positions of the FCP's semi-automatically. Using known relations between the coordinates of the FCP's, the user is restricted to place each FCP in a for that FCP valid area. Furthermore, the x-coordinates of certain FCP's are fixed depending on the coordinates of other FCP's. Also, a zooming function is available. Figure 4 shows a screenshot of the process of semi-automatically positioning the FCP's. When all FCP's have been determined, FED will determine the closest matching entries.

The second method of determining the label of an unknown facial expression is by sketching it with FaceShop. As mentioned earlier, in that case the FCP coordinates are determined automatically. Figure 4 shows a screenshot of the semi-automatic positioning- of the FCP's.



**Figure 4: determining the FCP coordinates semi-automatically**

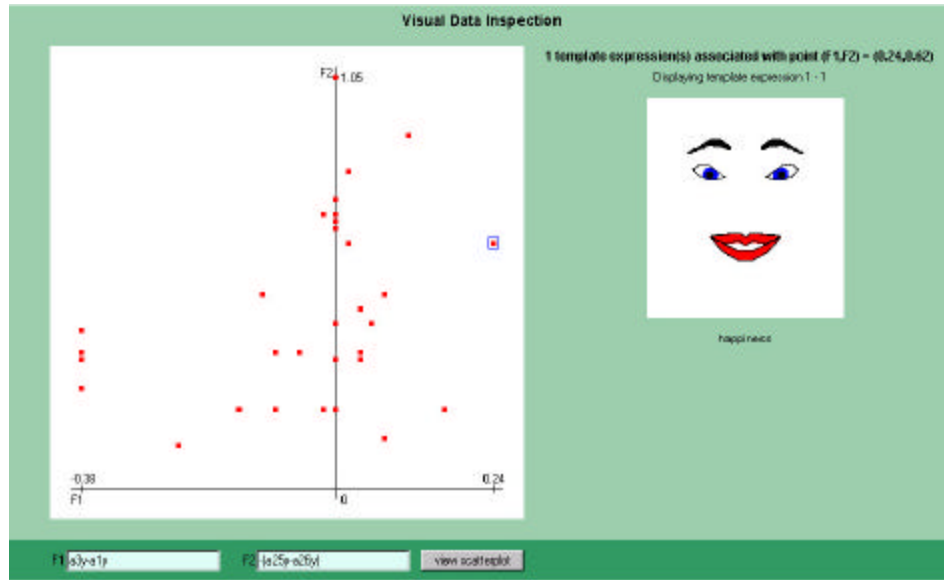


**Figure 5: results of the labeling of the picture of figure 4**

## 8. Visual Data Inspection

In order to determine to what extent certain geometrical features discriminate between facial expressions, a visual data inspection tool was developed. With this tool, it is

possible to create a scatter plot of two geometrical features. A user can define the two features by specifying an arithmetic expression on the FCP's. Figure 6 shows the openness of the mouth plotted against the slanting of the eyes.



**Figure 6: Visual Data Inspection of facial expression features in**

## References

- [1] P.Ekman. Movements with precise meanings. *Journal of communication*, 26:14-26, 1976
- [2] D.Efron. *Gesture, Race and Culture*. The Hague, Mouton & Company, 1972.
- [3] C.Pelachaud, N.I.Badler, and M.Steedman. Generating facial expressions for speech, *Cognitive Science*, vol 20, no.1:1-46, 1998.
- [4] P.Ekman and W.F.Friesen. *Unmasking the Face*. Englewood Cliffs, New Jersey, USA, Prentice-Hall, 1975.
- [5] A.Wojdel and L.J.M.Rothkrantz. *A Text Based Talking Face*. Proc. of the third Int. Workshop on Text, Speech and Dialogue, Brno, Czech Republic, 2000.
- [6] H.Kobayashi and F.Hara. *Recognition of Mixed Facial Expressions by Neural Network*. IEEE International workshop on Robot and Human Communication, 381-386, 1972.