CHAPTER 3

PCASA SYSTEM ARCHITECTURE

3.1 BLOCK DIAGRAM OF PCASA

New user → Capture video → Get signature → Collect different postures of Face

Update images

Features extraction

Existing user → Get DSC → Test signcryption Images

Generate signcryption

Generate DSC

Validate user

Figure 3.1 Block diagram of PCASA
Figure 3.1 illustrates the block diagram of PCASA. There are nine important stages. These processes are discussed in Chapter 5:

1. Capture of video
2. Face detection
3. Collection of different postures of face
4. Test signcrypted images when face is already exists
5. Comparison with existing image
6. Features extraction
7. Eigen faces
8. Nearest eigen classifiers
9. Reorganization

3.2 USE CASE DIAGRAM

![Use Case Diagram]

Figure 3.2 Use case diagram
Use case diagram explores the functions of the system and the variety of users of the system. The main components of the use case diagram:

1. Actor: It represents the variety of users of the system. Classifications of the actors are primary, secondary, off-stage.
   a. Primary actor – The person who demands the service of the system is considered as the primary actor.
   b. Secondary actor – is the provider of the service directly to the customer or user.
   c. Off-stage actor – is the person or system who acts as an intermediary in between the service request actor, the service provider actor to do the background process.
   d. Supportive actor – Any additional actor used to do the service, is considered as the supportive actor.

In the system, profounded in this research work, users are considered as the primary actors to get the service from the system. PCASA system is considered as the secondary actor. The signature recognizer is the supportive actor. The encryption system is the off-stage actor.

2. Use cases: Use cases are the text which depicts the common services of the system.

The role of each actor is:

User (primary actor) – request entry of the service of the system.
PCASA (secondary actor) – provides the service for recognition of the right user.
Signature recognizer (supportive actor) – receives the user’s signature.
Encryption system (off-stage actor) – encrypts the data.
Six use cases are derived from Figure 3.2. They are:

1. Login – is the process that is used by the user to enter into the system by giving their data.
2. Capture video – is the process for getting the face posture of the user; system automatically will take the video clip.
3. Get sign – is the process of the system that requests the signature from the user.
4. Match data – is the process of the system for checking the new input with the existing one.
5. Provide DSC – is the process which when given data match, generates Dynamic Secret Code (DSC).
6. Update trained set – This is the process wherein, whenever a user enters into the system, the system will update the different face postures.

3.3 CLASS DIAGRAM

Class diagrams forms the domain models of the system. It explores the entire structure of the system. More than that, class diagrams derive the relationship or association in between their interfaces. This denotes the logical view of the system.

Components of the class diagrams are

i) Classes – Real time group of objects.
ii) Aggregations – Collection of classes under one group.
iii) Generalization – General group name of combined sub groups.
iv) Specialization-Special group name of specific sub groups.
v) Inheritance-Relaying the attributes and behavior to its sub classes.
vi) Cardinality – Number of connectivity in the cardinal way.
vii) Modality - Availability of connectivity.
viii) Multiplicity - Number of connectivities belongs to its classes.
ix) Role name - Name of the behavior.
x) Attributes - Qualities in nature.
xi) Functions – Methods to do its works.
xii) Accessibility – Mode of visibility and retrieval authenticity.

a) Classes and their structure and behavior – Class names are indicated at the top of the class diagram as users, video, and sign. Each rectangle box indicates the attributes and behavior of the class. It has three compartments. The top compartment indicates the name of the class or object. The middle compartment indicates characteristics, qualities, attributes of the class. The bottom compartment indicates the functionalities of the class.

b) Association, aggregation, dependency and inheritance relationships – Relationships between the classes are indicated by arrows. If more than one class is needed for connection with another class, the middle agent is considered as association class.

Aggregation allows a combination of several classes.

Figure 3.3 shows the dependency between the classes by messages over an arrow. Multiple and multi level inheritance can also be denoted by hollow arrows.

c) Multiplicity and navigation indicators – Multiplicity is one of the characteristics of the class diagram used to denote the number of relationships. Navigation indicators are used to indicate the directions of the message passing by arrow.

d) Role names – Most of the class names are created by the role names.
Figure 3.3 Class diagram1

Figure 3.3 illustrates relationship between user, video and sign.

Figure 3.4 Class diagram2

Figures 3.3 and 3.4 illustrate the attributes and functions of the objects.
3.4 ACTIVITY DIAGRAM

The activity diagram determines the internal activities of the system and its behavior. Components of the activity diagrams are:

Swimlanes: They assign the proper path of space for the objects by dividing such objects over lines.

States of the action: Represent the actions of the entities as in the system flow steps.

Action flows: The flow of actions is represented by arrows with its respective responsibilities.

Object flows: It controls the action states.

Figure 3.5 Activity diagram
Figure 3.5 illustrates the entire range of activities of the PCASA system. The filled circle represents the start of the process. The rounded rectangle symbol represents activities. The diagonal symbol represents decision making conditional statements. The solid arrows represent the flow of the process. Rounded filled circle represents that the activities are halted.

3.5 SEQUENCE DIAGRAM

The sequence diagram is used to pass the messages in between two classes or objects. It mainly concentrates on classes and the intercommunication of its works by message passing.

- These diagrams focus on classes and the messages they exchange to accomplish some desired behavior.

- Sequence diagrams are a type of interaction diagrams.

They have the following components:

Class roles: Represents play roles of an object.

Lifelines: Represent the availability existence of an object over a period of time.

Activation: Represent the time during which an object is performing an operation.

Messages: Represent communication between objects.
Scenario 1: If a new user comes to register

Figure 3.6 Sequence diagram - Scenario 1

Scenario 2: If existing user comes

Figure 3.7 Sequence diagram - Scenario 2
Figure 3.6 explains the scenario of a new user entering the system. Figure 3.7 explains the scenario of an existing system user’s log in. Figure 3.8 illustrates the activity when the same user comes in but in a different style. Figure 3.9 illustrates the activity when an entirely different user looks like the existing one. The sequence diagram demonstrates the interactions between the user and the system. Rectangle boxes are represented by the objects. The solid arrow which carries message from left to right request the service to the system. Dotted arrow represents responses from the system. Messages are explained by the functions of each object.

Scenario 3: Same user with different clothing

Figure 3.8 Sequence diagram - Scenario 3
Scenario 4: If different user, looking as existing one

Figure 3.9 Sequence diagram - Scenario 4

3.6 PACKAGE DIAGRAM

Figure 3.10 Package diagram
Figure 3.10 illustrates the PCASA system packages. The user package has video and signature packages. The video package consists of images. Java.io is the built-in package of Java that is used by the user package to get and display the data.
CHAPTER 4

DEMONSTRATION OF EXISTING SYSTEMS

4.1 PCA ALGORITHM

The Principal Component Analysis Algorithm is used for the recognition of a human face using eigen vectors. Image M can be represented by a matrix using its pixel values. The vector which is used to represent these values is said to be a eigen values of an image. Sub vectors from these basis vectors to compare stored and acquired new images. Based on these vectors the principal components of the given image can be found. The acquired new image is then compared with the existing stored images to achieve the perfect match.

Figure 4.1 illustrates the block diagram of PCA.

![Block diagram of PCA](image-url)

Figure 4.1 Block diagram of PCA
4.2 SWAPPING ALGORITHM

Swapping algorithm is the method that is used to encrypt the message easily.

![Block diagram of swapping algorithm](image)

Figure 4.2 Block diagram of swapping algorithm
Figure 4.2 shows the block diagram of swapping algorithm.

Features of swapping algorithm are:

- It uses 32 bit key
- Stream buffer encoding
- ANSI coding
- Uses single primitive: XOR
- Data is divided into two halves

Swapping algorithm steps:

Step 1: Start.

Step 2: Get message.

Step 3: Swap it,

\[
\text{L} = \text{len}(\text{msg});
\]
\[
\text{Swap(int)}
\]
\[
\text{While}(i <= 0)
\]
\[
\{
\]
\[
L_0 \& R_0
\]
\[
\text{for } i = 1 \text{ to } L \text{ do}
\]
\[
R_i = L_{i-1} \text{ XOR } P_i;
\]
\[
L_i = F[R_i] \text{ XOR } R_{i-1};
\]
\[
\text{L} = \text{len}(\text{msg})/2;
\]
\[
\text{Swap(L)};
\]
\[
\}
\]
where

\[
F[a,b] = ((S_1,a \text{ XOR} S_2,b) \text{ XOR} S_3,c) \text{ XOR} S_4,d
\]
\[
S[a,b] = F[a,b]/2;
\]

Step 4: Get encrypted message.

Step 5: Stop
The algorithm provides reliable security. Since the key is generated by the system and the user, it gives higher security than dictionary password. It cannot be easily extracted by the hackers. After that the carrier signal is swapped with the generated key. This function has many combinations of AND, OR gates to collapse the text. This technique provides a strong encryption to a given text.

### 4.3 BLOWFISH ALGORITHM

Blowfish is a keyed, symmetric block cipher, designed in 1993 by Bruce Schneier and includes in a large number of cipher suites and encryption products. Blowfish provides a good encryption rate in software and no effective cryptanalysis of it has been found to date. However, the Advanced Encryption Standard now receives greater attention.

Schneier (1993) has designed Blowfish as a general-purpose algorithm, intended as an alternative to the aging DES and free of the problems and constraints associated with other encryption algorithms. When Blowfish was released, many other designs were proprietary, encumbered by patents or were commercial/government secrets. Schneier (1993) has stated that Blowfish is unpatented and will continue to remain so in all countries. It can be freely used by anyone.

The notable features of the design include key-dependent S-boxes and a highly complex key schedule. Figure 4.3 shows S-boxes manipulation of Blowfish algorithm.
This part is the inbuilt process for doing the encryption. It

• uses two primitives: addition & XOR

• divided data into two 32-bit halves $L_0$ & $R_0$

for $i = 1$ to $16$ do

\[ R_i = L_{i-1} \text{ XOR } P_i; \]
\[ L_i = F[R_i] \text{ XOR } R_{i-1}; \]
\[ L_{17} = R_{16} \text{ XOR } P_{18}; \]
\[ R_{17} = L_{16} \text{ XOR } i_{17}; \]

where $F[a,b,c,d] = ((S_{1,a} \text{ XOR } S_{2,b}) \text{ XOR } S_{3,c}) \text{ XOR } S_{4,d}$
4.4 SHA-256 ALGORITHM

The speciality of the blowfish algorithm is its expectation of its key and the complexity of the S-block routines.

SHA- is a group of hash functions denoted by its bits. It was developed by the U.S National Security Agency (NSA). It is considered as a U.S Federal Information Processing standard by NIST.

SHA-hashes each text character into a specific value. This mapping process can be done by the computerised program. Computation of the hash value is required for each character of given text data. 224,256,384,512 bits SHA have been developed yet.

The security provided by a hashing algorithm is entirely dependent upon its ability to produce a unique value for any specific set of data. When a hash function produces the same hash value for two different sets of data, a collision is said to occur. Collision raises the possibility for an attacker to computationally craft sets of data which provide access to information secured by the hashed values of pass codes or to alter computer data files in a fashion that would not change the resulting hash value and would thereby escape detection. A strong hash function is one that is resistant to such computational attacks, while a weak hash function is one where a computational approach to producing collisions is believed to be possible. A broken hash function is one where a computational method for producing collisions is known to exist.
Hashing algorithm steps are:

Step 1: Get original message.
Step 2: Get password to encrypt.
Step 3: Create message digest using original message.
Step 4: Generate hash value to each character of original message using key.
Step 5: Get output.

4.5 MD5 ALGORITHM

MD5 means Message Digest version 5. It has been developed by Ronald Rivest (1992). It accepts variable length message and converts it into a fixed length output of 128 bits. A given Input can be broken into small parts of 512 bit messages that are sixteen 32 bit blocks. Then the message is padded to get fixed length message which is divisible by 512.

The padding mechanism follows two steps detailed below:

i) Single bit value 1 is added to the end of the message.
ii) Later several zeros are added until it reaches its fixed length word up to 64 bit but lesser than the multiple of 512. The rest of the bits are filled with the original message.

MD5 message digest works with 128 bits which can be divided into 4 X 32 bit words. So that four different words can be formed. That is represented by A,B,C,D as given below.

\[ F(B,C,D) = (B \land C) \lor (\neg B \land D) \]

\[ G(B,C,D) = (B \land D) \lor (C \land \neg D) \]

\[ H(B,C,D) = B \oplus C \oplus D \]
\[ I(B, C, D) = C \oplus (B \lor \neg D) \]

\( \oplus, \land, \lor, \neg \) denote the XOR, AND, OR and NOT operations respectively.

Figure 4.4 Block diagram of MD5

The message is fragmented into 512 bit blocks. Each block maintains its own status whenever message is switched over from one block to another. \( F \) is a non-linear function which allows the performance of different operations on the message. Probably addition and XOR functions are used to scramble the message.