5.1 INTRODUCTION

This chapter proposes how Real Time Constraint notations are utilized for the Formal Verification of the current models such as ATM Transaction using biometrics [77, 78, 79, 80] (among which finger print recognition is very common and famous). To identify the users, the epidermis of the fingers is used, as every individual’s epidermis is unique. When user places the fingerprints on sensory pads, epidermis ridges are read and compared with already developed knowledge bank.

Transaction Oriented systems can be well defined using Finite State Machine (FSM), Sequence Diagrams (SDs) and can also be used for developing state charts. But, fingerprint ATM system specially needs support for branching, state information and composition of sequence diagrams.

Conventionally, access control for confidential information is managed by presence of specific artifact like key or card or possession of some specified information like a password or a
Personal Identification Number (PIN). Now-a-days, persons use passwords and PINs for many devices such as web-based services, their bank information computers, mobiles and many more. This results into problems regarding the use, memorization and security of these passwords. Several methods like frequently changing passwords and PINs, having lengthy passwords, making difficult combinations, using different passwords for different domains are used for increasing security but they cause more problem in memorizing. Therefore, other option for conventional PIN is looked using fingerprints.

The conventional method depends on the assumptions that these artifacts will be possessed only by the legitimate user and thus remains confidential. But, one cannot totally rely on these assumptions. People generally generate passwords relating their daily life and can be remembered easily, they keep same PINs for all domains thus making them vulnerable to be cloned. Users don’t understand the importance of confidentiality of these passwords and chances are that they may share them with friends and family members, or may write down the passwords at easily accessible places. Biometric techniques tend to solve the above mentioned problems. They ensure the actual presence of the person
rather than relying on the passwords. Thus people don’t need to remember the PINs or passwords.

Herein, we describe how UML sequence diagrams [81, 82, 83] can be used in the ATM verification system using fingerprints. For this, we first need to know the methods for creating sequence diagrams for fingerprint ATM modeling, along with the method by which the required content can be expressed as a FSM, to ensure precise, cohesive diagrams. We have presented an approach in which fingerprint based ATM Verification system is imbibed with the data, state and timing information. And the last transaction processing model usually used is reevaluated to show the system differences.

5.2 APPLICATION OF FINGER PRINT RECOGNITION IN ATM TRANSACTION – A BIOMETRIC APPROACH

A biometric system uses certain behavioral and / or physiological characteristic to authenticate a user. Thus it is considered as system which recognizes patterns. It can work as an identification system or as a verification system.

1. Verification system compares the collected biometric features of a user with his own template/templates already saved in the database which ensures one to one
type matching to authenticate a person depending on the result a claim is accepted or rejected.

2. Identification system compares the collected biometric features with all the templates of the system database. Thus it carries out one-to-many comparisons. Herein user does not claim for the identity, rather system establishes the identity.

Since authentication [84, 85] is commonly used term, hence a system acknowledges the user identity irrespective of verification or identification mode.

Block diagrams of both identification and a verification system are shown in Fig 5.1. The user enrollment module that carries out registration of individuals in biometric system database is common to both. In this phase, the biometric features of a user are scanned using a biometric reader to generate a raw digital expression of these features. A quality control is usually carried out for checking whether collected feature could be truly used by upcoming stages. For matching, this raw expression serves as input to feature extractor for developing a compressed; well-defined ‘template’.

As per the application, these templates can be stored either in system’s central database or on some secondary storage devices such as magnetic cards, smartcards etc.
The verification task is used at access point to verify the user by asking username and PIN. When the user enters this information using a keyboard, the biometric reader captures details of user and converts it into digital data, which is further used for identification. Feature extractor collects this digital data and generates a compact digital expression. Feature matcher then compares this expression with the template of one user (templates stored in system DB).

PIN is not used in the identification instead a biometric input expression is compared with the templates which is stored in the database and represents individually all the users. In case it does not match, an alarm message is generated indicating ‘unidentified accesses.’
Since database is very large, identification task becomes very expensive, therefore in order to limit the number of template comparisons, certain classification and indexing methodologies are used. Depending on type of application, a biometric system may operate on-line (i.e. generates immediate response) or off-line (where a delayed response is allowed).

On-line system uses live scanners to capture the individual’s features. It operates automatically where no supervision of enrollment process is required; no check over quality control is required.

Off-line system uses off line scanners like fingerprint cards. They are semi-automatic i.e. enrollment process needs to be attended. A manual quality control check is required and feature matcher generates a list of possible templates, which needs to be checked manually to get the final result.

There are two types of recognition modes, in which an application can operate:

1. **Positive recognition application**: It keeps a check by preventing same identity being used by several individuals. Thus ensuring the right user.
2. **Negative recognition application**: It keeps a check by preventing a user from using many different identities. Thereby verifying whether the person is the original one.

5.3. **TERMINOLOGY USED**

5.3.1 **Scenario, Message Sequence Charts, Sequence Diagrams, State Chart:**

The list of events, occurring during a specific execution of a system is defined as Scenario. It defines how a system must be used to carry out a function. It can be represented in different ways like text and graphics, formal and informal.

Sequence diagrams, as the name suggests depict the order of events with time and collaboration diagrams depicts the structure of communications among the objects.

State Chart Diagram represents the entities’ behavior. They specify the response of events to a particular instance. Basically, they define the behavior of classes and are also used for other modeling entities like actors, operations, methods, e-cases and subsystems.

Message Sequence Charts are used to visually formalize the system requirements during the design phase like ATM-Transaction using fingerprint recognition.
5.3.2 Composition and Integration of Scenarios:

Composition and integration of scenarios is very challenging task for the formal verification of bio-metric (fingerprint) ATM recognition. Sequence diagrams must represent the programs’ structures. Herein we present an approach which is used for modeling of execution structures and transfer of control.

Secondly we developed an approach for Fingerprint Verification System.

The aim here is to process and improve a model having sequential, concurrent, conditional and iterative executions. Many possible ways are available out of which the best for Fingerprint Verification System is to be determined.

Hsia [86] proposes a method enabling analysis of scenarios having conditional branching. Iteration behavior is added by Glinz [87]. Koskimies [88] and Systa [89] proposed a tool to handle “algorithmic way scenario diagrams” i.e. Sequence Diagrams (SD) with sequential, concurrent, closure (iterative) and conditional property. Thus a model, combining the components of each is developed which has sequential, concurrent, conditional, and iterative behavior.

One more target is to use sequence diagram composition for modeling transfer of control. The important thing is where control information must be illustrated. A method that can be
used is composition information can be added to diagrams individually.

5.3.3 Finite State Machines (FSM):

It can be defined as a conceptual model (software, physical, biological, mechanical, or electronic).

It is a mathematical model of a system which reduces the complexity of model by specifying certain assumptions like:

1. A system can be present in some possible ways called states which are finite in number.
2. System’s behaviour for a given state must remain same.
3. System remains in a state for some specific time period.
4. System changes from one state to another by some well defined transitions which are finite in number.
5. Events cause these transitions.
6. Transitions are instant (i.e. occur in zero time).

5.3.4 Object Constraint Language (OCL):

It allows defining constraints over object – oriented models, hence is called an expression language.

A constraint can be defined as restrictions on some values of an object oriented model. OMG (Object Management Group) goals at providing a common framework for development of applications (based on object oriented programming
methodologies) and standard for object oriented analysis and design.

Object Constraint Language - OCL has much wider scope. With OCL many under – specified domains using OCL and UML have been identified.

OCL is used for the following:

1. specification of constraint on operations
2. to define guard
3. as a navigation language
4. Pre and Post conditions imposed on operations and methods
5. Type specification for Stereotypes invariants
6. Invariants specification in classes and their types

UML operation semantics are expressed by Pre / Post conditions. Pre condition defines the condition that must be true for successful execution of operations. The post condition defines what will be true after completion of execution i.e. it defines the return value and state change.
5.4 PROPOSED APPROACH FOR FORMAL VERIFICATION

OF FINGER PRINT ATM - TRANSACTION THROUGH

REAL TIME CONSTRAINT NOTATION (RTCN) : -

This approach shows how Fingerprint Verification Model can be used in banking sector in the working of ATM with the help of fingerprint recognition system. It highlights the use of Sequence Diagram (SD’s) and their corresponding Finite State Machine (FSM) and Real Time Constraint Notation (with the help of OCL). We have four objects which exchange messages: the user, ATM, Consortium, and Bank. Here, only ATM object are represented by state charts. The scenarios relates to the same initial (start) condition.

5.4.1. Using Sequence Diagrams - SD’s:

Case-1: Transaction Fails at server site database due to discrepancy with Finger Print Impression (FPI):

ATM-transaction gets failed in this case because of mismatching of finger print in finger print collection database (DB) file server location as shown in Fig 5.2
Fig 5.2: Sequence Diagram for Fingerprint (Biometric) Verification
ATM, Case 1: Unmatched Fingerprint (FPI)

Case 2: Transaction done at server database due to mapping of Finger Print Impression (FPI):
In this case, finger print matches accurately at finger print database (DB) file server site and thus ATM transaction is successfully processed shown in Fig 5.3:
Fig 5.3: Sequence Diagram (SD’s) for Fingerprint Verification ATM, Case2: Equivalent FPI, Successful Transaction

5.4.2 Finite State Machine (FSM) corresponding to Sequence Diagrams (SD’s):

Finite State machine as can be created to represent the above two cases:
Fig 5.4: Finite State Machine (FSM) for ATM Transaction by FPI

Once Finger Print Impression Verification is done for ATM Transaction, corresponding Finite State Machine (FSM) is generated which is represented in Fig 5.4. Finite State Machine has ten (10) states which are denoted as $q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9$, where initial state is represented by $q_0$ and the final state by $q_9$. Transition function ($\delta$) of FSM can be defined as in Fig 5.4 as follows:

1. $\delta (q_0, \text{insert card}) = q_1$
2. $\delta (q_1, \text{request for FPI}) = q_2$
3. $\delta (q_2, \text{FPI}) = q_3$
4. $\delta (q_3, \text{FPI matching}) = q_4$
5. 5.1 $\delta (q_4, \text{FPI matched}) = q_5$
   5.2 $\delta (q_4, \text{FPI mismatched, re FPI}) = q_1$
6. $\delta (q_5, \text{request for nature of account}) = q_6$
7. $\delta (q_6, \text{request for net amount to be withdrawal}) = q_7$
8. $\delta (q_7, \text{amount entered as per condition of ATM}) = q_8$
9. **9.1** \( \delta (q_8, \text{transaction aborted}) = q_0 \)

9. **9.2** \( \delta (q_8, \text{request to amount received}) = q_9 \) (Final State)

10. \( \delta (q_9, \text{request to receive card}) = q_0 \) - Initial State

**5.4.3. Real Time Constraint Notation (RTCN)**

**corresponding to Sequence Diagrams - SD’s (Case 1 and 2) using Object Constraint Language - OCL:**

Here user’s ATM Transaction after verification of finger print is represented using OCL as proposed below:

\[
\text{User::Finger Print Impression (FPI)}
\]

**pre:** Finger Print Identification

**post:** Request Transaction.Saving Account \( \rightarrow \text{reject}(\text{not}(FPI=\text{User})) \)

**OR**

Request Transaction.Saving Account \( \rightarrow \text{reject}(FPI=\text{User})) \)

**and**

(Request Transaction and Saving Account@pre+Account Balance\(->\text{Minimum Balance})

**and**

(Request Transaction and Saving Account@post+Account Balance\(->\text{Account Balance–Withdrawal Amount})

The above proposed approach ensures sequence diagram data to develop precise State charts. Secondly conversion of
sequence diagrams into corresponding Finite State Machine - FSM along with Object Constraint Language - OCL, pre and post conditions for transaction process in the process of ATM. Using state, message pre-conditions, and timing information (data) in the notations of OCL and FSM, sufficiently ensure Fingerprint ATM Verification side by side checks diagram composition and information content to assess adequacy for Fingerprint ATM Verification.

5.5 SIMULATION BY UMLtoCSP

For ensuring the formal verification of UML and OCL models the tool UML to CSP [104] tool has been developed using constraint programming. This tool is used for validation of various properties of different models. The formal development of the tools has been carried out for user using the constraint programming logic in environment.

This tool also supports Model Driver Development (MDD) and major focus is on the change paradigm from existing software code to models. These tools play an important role is improving the quality of software products. This tool focuses on analysis of UML class diagrams incorporated with OCL constraint has been carried out, It supports all the programming aspects of UML and OCL languages.
The major thrust of this verification tool is bounded verification which makes it capable for automatic and decision oriented verification. The initial development of the model is for the problem of Constraint Satisfaction Problem (CSP).

Various correctness properties of UML/OCL models have been proposed which are as follows:

1. **Strong: Satisfiability**: The instances of the model should be finite regarding class and association.
2. **Work Satisfiability**: The instances of the model must be finite where nonempty population is one class.
3. **Liveliness of a class**: The instance of the model must be finite where \( X \) is not empty.
4. **Lack of constraint sub assumption**: There exist a constraint in between two \( C_1 \) and \( C_2 \) where one is satisfied and another is not satisfied.

UML to CSP is a standalone application from the command line or graphical user interface. Finally, this tool can verify the quality characteristics of UML class diagrams with OCL constraints. This approach is fully automatic and gives excellent feedback to users/designer of the model.
Fig. 5.5: Classes & Attribute representation in umltocsp tool

The classes and associated attributes along with their values are represented in the tool as given in Fig. 5.5. Normally the functionality of this tool can be described in 4 steps. In first step, the model has two types of files, XMI files with UML class diagram and a text file with OCL constraints.

Fig. 5.6: Constraint property checking method in umltocsp tool
In step 2, properties are selected for verification as given in Fig. 5.6. The property selected for verification must be strongly satisfiable. Step 3 is optional and limits of the search are carried out in terms of their limits. The designer may choose the default values for the attributes.

![Fig. 5.7: Constraint condition execution method in umltoCSP tool](image)

Using all the above setup of the experiments, the whole process is executed as given in Fig. 5.7. The verification of the model is done automatically. The results are shown graphically in the form of object diagram.
We have also executed our proposed model in the UMLtoCSP tool the object diagram for the whole execution is given in Fig. 5.8. The model is found to be strongly satisfying with its OCL constraints and UML diagrams.