2003

Estimating reliability impact of biometric devices in large scale applications

Karthikeyan Mahadevan
West Virginia University

ESTIMATING RELIABILITY IMPACT OF BIOMETRIC DEVICES IN LARGE SCALE APPLICATIONS

by

Karthikeyan Mahadevan

**Thesis submitted to the College of Engineering and Mineral resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of**

**Master of Science
in
Electrical Engineering**

Approved by

Department of Electrical Engineering and Computer Science

**Morgantown, West Virginia
2003**

Keywords: Biometrics, component-based software engineering

Abstract

ESTIMATING RELIABILITY IMPACT OF BIOMETRICS IN LARGE SCALE APPLICATIONS

by Karthikeyan Mahadevan

In the last two decades, there has been a tremendous growth of biometric applications especially in security. Reliability of the biometric devices is extremely important.

This thesis discusses an approach for estimating the reliability of systems, which contain biometric user authentication subsystem. The ECRA (Early Component Based Reliability Assessment) tool utilizes an easy to use interface and employs the Bayesian algorithm to predict the system reliability. This application of the ECRA technique to biometrics is new. Using the UML diagrams and the ECRA tool, the reliability of the system is predicted.

Dedication

Dedicated to my family who has always supported me.

Mahadevan

Kulakodi

Vijay

I would also like to dedicate my thesis to Dr.Cukic for his patience and for the knowledge he has imparted to me.

Acknowledgments

The author wishes to thank the people who made this work possible. First of all I would like to thank my advisor and mentor Dr. Bojan Cukic for his support and understanding, and for giving me the opportunity to work with him at West Virginia University. This thesis is primarily based on the previous work of Mr. William Smith and I like to say thank you for his support and the opportunity that I had working with him. I would like to thank the other two members of my Graduating Committee, Dr. Nutter and Dr. Choudhry for their support and providing me with precious knowledge that I have gathered during my studies at West Virginia University.

Finally I like to say big thank you to Ms. Veena Adityan who has cheered me during the difficult moments of working on this thesis.

TABLE OF CONTENTS

List of Figures

List of Symbols, Abbreviations, or Nomenclature

1. BIR – Biometric Information Record
2. COTS – Commercial off-the-shelf software
3. DB - Database
4. DD – Deployment Diagram
5. ECRA – Early Component – based Reliability Assessment
6. GSI – Grid Security Infrastructure
7. LAN – Local Area Network
8. PKI – Public Key Infrastructure
9. ROC – Receiver Operating Characteristics
10. SD – Sequence Diagram
11. UCD – Use Case Diagram
12. UML – Unified Modeling Language
13. WAN – Wide Area Network

Introduction

The term "Biometrics" is derived from the Greek words bio meaning life and metric meaning to measure. The first known example of biometrics in practice was a form of finger printing in China in the 14th century. Biometrics has found applications in the field of computer and network security. Today the science of biometrics is one of the well studied and documented fields.

In today's computer world, many see biometrics as a solution to various authentication and security problems. Password is one of the main weak links in the security chain, the reason being human error. Human error can be anything from choosing obvious passwords to leaving the passwords on one's desk. Biometrics can nullify the security breaches that are caused by human errors. Security depends on one of the following: what you have (tokens etc.), what you know (passwords, PIN's etc.) or who you are (biometrics). Why biometrics is necessary? Tokens can be lost or stolen, passwords could be forgotten. Neither tokens nor passwords can provide *positive identification* of the person. A biometric system in simpler terms is a pattern recognition system. It includes both software and hardware that are necessary for identifying an individual. The process of acquiring and storing a pattern into the database is called biometric enrollment. For authenticating a user, a live biometric is captured using a scanner and it is converted into a template which is matched with the stored template. Biometric devices can be employed for both verification ("Am I who I claim to be?") and identification ("Who am I? ") purposes.

This chapter will provide an introduction to biometrics and software reliability engineering. It will present a description of what software reliability analysis is, how it is performed, and the benefits of performing it. In addition, this chapter will provide an insight into how the work and research presented here will contribute to the field of biometrics. This chapter concludes with a short description into the content of the remaining chapters of this thesis.

1. Biometrics

Biometrics is a science of recognizing people based on physiological or behavioral characteristics. Biometrics is one of the fast growing fields in the industry. It has diverse applications ranging from user authentication to day-to-day applications. Even though the science of Biometrics has been well studied and documented, *the robustness and reliability of biometric devices has not yet been defined completely.*

The most commonly studied features are: fingerprints, face, hand geometry and iris. Combining a few of these to obtain better performance is called multi-modal biometrics [12].

2. Software Engineering

“Software Engineering is defined as:

(1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

(2) The study of approaches as in (1) “[20]

Software not only refers to the programs written, but also to the documentation that is necessary for developing, installing, using and maintaining a software system. The challenge for a software engineer is to produce high quality software using a finite number of resources and within a predictable amount of time. Because of the complexity and range of the tasks to be automated, a software engineer must be able to assess and apply existing computing knowledge, derived from more fundamental subjects, in a cost-effective and functional way.

Well-engineered software does what the user wants, and can be redesigned to suit the needs of the user changes [5]. The following are the attributes of well-engineered software:

1. Maintainability: It should be easy to maintain the software. It should be possible to upgrade the software depending on the needs of the customer without much cost.

2. Dependability: Dependability includes a range of characteristics namely: reliability, security and safety. Dependable software is one, which does not cause any damage (economical/physical) in case of system failure.

3. Efficiency: The software should not waste the resources like memory or processing power.

4. Usability: It should be easy to use and should provide documentation. An appropriate user interface taking into account the background of the intended users is necessary.

The process of implementing a complex software system with reusable components is called “Component Based Software Engineering”. In component based software engineering: A component represents a distributable piece of implementation of a system like a model or code of a software system.

3. Software Reliability

“Reliability is defined as the probability of failure-free operation for a specified time in a specified environment” [4]. From a user’s perspective, reliability is a measure of how well system users think it provides the services they require.

Reliability is an attribute of any computer-related component (software, or hardware, or a network, for example) that consistently performs according to its specifications. It has long been considered one of three related attributes, the other two being availability and serviceability that must be considered when making, buying, or using a computer product or component. *This is one aspect, which has not found the attention it needs.*

A failure corresponds to unexpected run-time behavior observed by a user of the software. A fault is a static software characteristic that causes a failure to occur. All faults need not necessarily cause failures. They only do so if the faulty part of the software is used.

Why is Reliability important? Software systems are increasingly being used in safety-critical applications such as nuclear power plants, aircraft, submarines or medical devices, where the assurance of software reliability has become an issue of great importance. For instance, consider a scenario where users are authenticated using their fingerprint in a Power Grid. In such a situation failure of the Biometric authentication system may prove problematic.

Assessing reliability of Biometric devices can be viewed from two different aspects. The device in itself is hardware; so rigorous testing of the device in various environmental conditions is one aspect of the problem. This has been studied though complete results are unavailable. Then the process of identifying or authenticating a person based on Biometrics is a software process. This can be viewed as a software reliability issue.

4. Statement of Problem

This thesis discusses an approach for estimating the reliability of systems, which contain biometric user authentication subsystem. The ECRA (Early Component Based Reliability Assessment) tool also utilizes an easy to use interface and employs the Bayesian algorithm to predict the system reliability. This application of the ECRA technique to biometrics is new. Using the UML diagrams and the ECRA tool, the reliability of the system is predicted.

5. Thesis Outline

The remainder of the thesis is separated into three chapters. Chapter two discusses some of the other research options that have been investigated to address biometric reliability assessment. Chapter three continues by discussing the ECRA methodology. It covers the assumptions and calculations of the reliability model. The next chapter discusses how ECRA has been applied to the study of biometric reliability. It also delves in details of the models, the UML diagrams (use case, sequence and deployment diagrams), the component reliability records, the results that were obtained and the inferences. Finally, Chapter four concludes with a summary of the work that has been done thus far and also discusses future work planned.

Chapter 1: RELATED WORK

A failed operation of software can lead to economic loss and may even cause loss of human lives, thus proving its importance in daily life. Therefore, unreliable software is not acceptable and should be identified in the early stage of software development. Software reliability is one of the key metrics for determining the quality of software. It is often defined as the probability of a failure-free operation of a computer program within a specified exposure time interval [5].

Quality can be defined as “conformance to requirements at the start of use". How long can a product stay in operation with conformance to the requirements? That is where reliability comes into play. The quality level might be described by a single fraction defective. To describe reliability, a probability model that describes the fraction fallout over time is needed. This model is called *life distribution model* [5].

There have been studies conducted for a good number of years for measuring reliability of given software resulting in many analytical models. The focus of these models has been the observation of the behavior of the software based on operational usage profiles. Data is collected for a period of time and measurements are made. There has been little focus on the structure of the software. Also the later stages of the software developmental cycle have been targeted for the assessment. These models can be classified based on the means they use. They are as follows:

1. Statistical Sampling Methodologies: Problems arise when there is not enough test data or if there are changes in the software.
2. Markov Modeling: The approaches that employ Markov processes consider the structure of the software, but only simple homogenous architectures have been considered.

Today, complex heterogeneous architectures that meet various requirements are widely seen. Systems that are designed for high performance and fault tolerance fall into this category. It can also be said that most of the software that is seen today is deployed in such systems. There is another important category of architecture, which includes of the shelf components. The question arises whether the existing approaches are sufficient for their reliability analysis. The existing white-box Markov-based model does not cater directly to these architectures. The aim of the model proposed in [1] is to take heterogeneity of software architecture into account and allow it to be applied to various types of software infrastructures at an early stage of software development.

1. Classification of Software Reliability Models

The most popular software reliability models [7] can be classified into data - domain and time - domain models.

Data Domain models are based upon the concept that if all the inputs to the software program are known then studying the reliability of the system breaks down to employing all the possible combinations of input and looking at the output. But in reality it is not be possible to know all the input combinations. So, a sample data set representing these input combinations can be studied and the estimation of failure rates could be done.

The data-domain models can be further classified into:

1. Fault Seeding Models: In this technique, the number of faults in the software is unknown. A known number of faults is “seeded” into the software and testing is done. An estimate of the number of faults in the software is then obtained by the ratio of the discovered seeded faults to discovered faults in the software. An example of this methodology is Mills’ Hyper geometric model [7].
2. Input Domain Models: The software is tested with a set of randomly chosen inputs. The reliability is then estimated by the ratio of the successful inputs to the total number of inputs. An example of this type of model is that proposed by Nelson [7].

Time Domain models rely on the underlying observed failure history to estimate the remaining number of faults in the system and also the time required for testing the software. These models can be categorized into:

1. Homogenous Markov Models: These types of models make the following assumptions

    a. The initial number of faults in the software is unknown but fixed
    b. The number of faults in the system at any time forms the state space of a homogenous Markov chain.
    c. The failure intensity of the software, or the transition rates of the Markov chain depend upon the number of residual faults in the software.

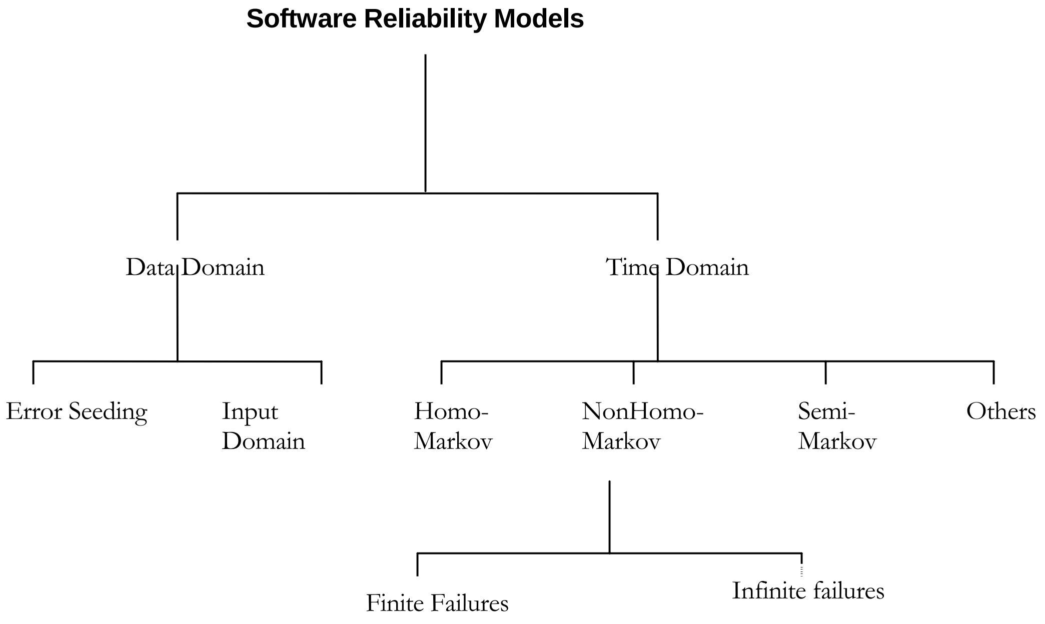
The famous models that fall into this category are Jelsinki-Moranda and Goel-Okumoto imperfect debugging model [7].

2. Non – Homogenous Markov Models: The assumption behind this class of models is that the faults are random variables whose behavior is similar to that of a Non – Homogenous Poisson Process. The Goel -Okumoto model and Delayed S - shaped models are examples of this class

3. Semi – Markov Models: These models assume that the number of faults in a software system is unknown but fixed, and the failure intensity of the software or the rate of transition from a given state depends upon both the number of faults and also the time elapsed in the state. The Schick – Wolverton model is a representative of this class. [7]

4. Other Models: Littlewood – Verall Bayesian model, Keiller- Littlewoood model are some of these models [7].

A graphical classification of the software reliability models from [7] is shown below



Component-based software reliability models are used to predict reliability of component-based systems. The difference in approach is the fact that the system is divided into separate logical units as compared to the traditional systems. It can be seen that COTS (Commercial Off the shelf) could be a part of such systems and also they can be reused.

The characteristics of component-based systems can include [12]:

- Systems that have significant aggregate functionality and complexity.
- Components are self-contained and possibly execute independently.
- Components will be used "as is" rather than modified.
- Components must be integrated with other components to achieve required system functionality.

In general, software reliability models can be classified as being *black box* models and *white box* models. The difference between the two is simply that the white box models consider the structure of the software in estimating reliability, while the black box models do not. The architecture of a component-based system is novel and unique; hence black box testing is not appropriate. So testing should be done for individual components and parts of the system. This changes the rationale of software testing. It implies a reliability model for component-based software should be able to predict the reliability of each component and how it is used in the system i.e. their usage patterns, which will play a very vital role in modeling of system reliability.

Existing component-based software reliability model have been surveyed in [14]. The authors classify the various models in this field into three types: *state-based, path-based, and additive models*. There is a wide-range of modeling techniques available for each of these types.

2. Paradigms that govern the success of Biometric System

Robustness: It is a measure of the extent to which the physical characteristics or trait is subject to change over a period of time. For the biometric system to be successful, the characteristic should not be subject to large changes.

Distinctiveness: The biometric template for any two people should have wide differences.

Accessibility: A Biometric feature that be easily presented to a camera or any imaging device.

Acceptability: It is an indicator that denotes the extent to which people are willing to use a biometric device in day-to-day life.

Testing of Biometric Devices is difficult. The following is a quote from [9]:

"Testing of biometric devices requires repeat visits with multiple human subjects. Further, the generally low error rates mean that many human subjects are required for statistical confidence. Consequently, biometric testing is extremely expensive, generally affordable only by government agencies. Few biometric technologies have undergone rigorous, developer/vendor-independent testing to establish robustness, distinctiveness, accessibility, acceptability and availability in "real-world" (non-laboratory) applications. "

The paper [14] further classifies the methods employed as

- Application Based testing

- Distance Distributions
- Non-Dimensional Measures of Comparison
- Error Bounds
- Operational Testing

This paper clearly indicates that not much investigation has been applied to the potential performance of a biometric system in a large application.

3. Grid Security Policy

The dynamic nature of the Grid is one of the important features that must be taken into consideration before designing a security model. The three key functions that are necessary for any security model designed for the Grid are defined in [17]

1. Multiple Security Mechanisms
2. Dynamic Creation of Services
3. Dynamic establishment of trust domains

The alternatives that are proposed in [11] for multi-site authentication are: Kerberos or Secure Shell. A user delegates his credentials to a program – MyProxy client that runs on the MyProxy server. The client also chooses a username and password for the credential to prevent unauthorized access to the credentials. The user specifies the lifetime of the credential on the MyProxy server [17]. The MyProxy client then acts on behalf of the client. The client could log into the grid from virtually anywhere using the web browser.

The disadvantages due to single sign-on approach are as follows [11]:

1. If the user is mobile then the private key may not be accessible all the time.
2. Human error may lead to exposing the pass-phrase
3. In the scenario where a hacker obtains the encrypted private key, the task for the hacker is to break a password.

From the above background, it can be said the biometric device can successfully replace the pass-phrase, which is used to grant access to the user's private key.

Chapter 2: RELIABILITY ASSESSMENT OF BIOMETRIC DEVICES USING ECRA

Early Component Based Reliability Assessment (ECRA) is a tool that was developed by Smith et al. It is used for the purpose of biometric system reliability assessment. The papers by Cukic et al. [1] [2] are the basis on which the tool has been developed. The concept of Early Reliability Assessment is explained in detail in [1] [2].

The validation of a software system in its developmental stages is very crucial. Early validation of functional requirements is well known. But early validation of the non-functional requirements like reliability is still under research. Statistical validation of reliability requirements makes the approach appealing. The above stated reasons are the premise of the study of early component based reliability assessment. This model has been built in software and it is used in this study.

The ECRA tool serves the following purposes [3]:

1. It provides a probabilistic technique for reliability prediction that is applicable in the early phases of development, before an executable version of the system is available.

2. The ability to study the impact of individual components and interfaces to the reliability of the application, thus allowing a quantifiable method in selecting components when alternative reusable assets are available to result in maximum system reliability.

The tool makes the following three assumptions.

1. Assumption on the existence of knowledge about failure rates for components. Though this information traditionally is not available in component libraries, it is a speculation that over time this information would be presented in data sheets when a software component is purchased.

2. Second, the methodology of the tool is simplified by the assumption of independence of failures among different components. To help realize this assumption, there are some proposals to build applications that include component wrappers to isolate each component [15].

3. The final assumption made further simplifies the tool. It is assumed that component failure follows the principle of regularity, in which components are expected to fail at the same rate whenever invoked.

To work with these goals and assumptions, the ECRA tool models the software system by annotating UML diagrams that can be created using Rational Rose from Rational, Inc. The ECRA tool exploits three distinct diagrams available in UML namely: use case, sequence and deployment diagrams. The explanation on how to annotate these diagrams is presented in the [1].

The requirements of the software in relevance to these diagrams are as follows:

1. The use case diagram requires abstraction of the names of each actor and use case, along with knowledge of each connection that may exist between the two.
2. The sequence diagram involves the name of each diagram and modules within the diagram. Additionally, the tool is required to calculate the number of busy periods for each module in each diagram.
3. Finally, the deployment diagram requires abstraction of the name of each processor and process, along with the knowledge of each connection between individual processors.

1. Why UML?

Three types of UML diagrams are employed namely – use case diagrams, sequence or interaction diagrams and deployment diagrams. Annotations with reliability related attributes in these diagrams help us assess reliability. The design details in early stages provided by UML can be used to predict reliability based on known failure rates of components and connectors. Cukic et al. describe the reasons for success of UML in [2].

The UML notation provides diagrams that capture, under different views, software features that, if using other notations, would remain hidden;

- No standard software process is required to be used with the UML notation. In other words, the designer is free to choose the subset of diagrams that, in each lifecycle phase, allows appropriate modeling of the application under the development;
- The UML diagrams are syntactically related. Consequently, certain types of analyses, such as cross syntax checking, can be performed at any development time. The same notation can be used throughout the software lifecycle.
- The graphical representation of UML diagrams, together with the fact that the UML project is open to notational extensions, gives the potential for introducing annotations. Annotations enrich the software representation with additional information that may support different tasks like, for example, the validation of non-functional requirements.

2. Performance Measures of Biometric Devices

The approach adopted is to use the ideas that have been presented in [1,2] and devise a methodology for biometric reliability assessment. The performance characteristics of Biometric devices have been studied and documented. The following measures are crucial parameters for studying the performance of a biometric device:

a) Robustness of a biometric: It is defined as the stability and repeatability of the biometric. In other words the changes in the physiological or behavioral traits must not have an effect on identification or verification.

b) False-match rates: It is defined as the rate at which the device incorrectly matches a sample with a reference template.

c) False non-match rates: It is defined as the rate at which the device rejects a true match between a sample and the reference template.

d) Receiver Operating Characteristic curves (ROC curves): A graphical representation of false match versus non-match rates, and associated confidence intervals

There are established standards for measures like false match versus non-false match, bin error rate and throughput rate. With these parameters in mind selection of a biometric device can be done appreciably. Thus these parameters provide a benchmark. In applications where a biometric device is used for identification, bin error rates are the suitable measure.

Chapter 3: ASSESSMENT OF RELIABILITY

1. Algorithm

The UML annotations are given as input to the ECRA and the reliability of the system is estimated. This section explains the details on UML annotations. The use case diagram is a high-level view of the system. It describes the interaction between system and the actors. From a use case diagram two important parameters are considered namely:

1. The probability that an actor will use the system represented as q_i, where $\sum q_i = 1$(i = 1 to the number of actors)
2. The probability of an actor using a selected system behavior, represented as P_{ix}, $\sum P_{ix} = 1$ for actor q_i where x denotes each actor/use case connection.

The above two parameters are combined to produce the probability of a system behavior occurring given by [1]:

$$P(x) = \sum_{i=1}^{m} q_i * P_{ix}\,, \tag{1}$$

where m represents the number of actors that use the given system behavior. $P(x)$ gives the knowledge to predict the probability of a sequence diagram occurring.

The sequence diagram shows the interactions between components for performing a given task in time-scale. It is a must that each use case has at least one sequence diagram. In cases where there is multiple sequence diagrams for a use case then the probability $P(x)$ divided by the number of sequence diagrams used for representing the given system behavior. A busy period is defined as the interval of time between starting an interaction and ending with the same corresponding interaction, denoted by bp_{ij} for a component C_i in the sequence diagram *j*. An estimate of θ_{ij}, the probability of component i in the scenario j is obtained by the following equation [1]:

$$\theta_{ij} = Prob\ (failure\ of\ C_{ij}) = \theta_{ij} = 1 - (1 - \theta_i)^{bp_{ij}} \qquad (2)$$

From the deployment diagram, the physical configuration of the software in relevance to the processors and their connections can be understood. The combined knowledge of the use case and sequence diagram in conjunction with the deployment diagram is used to define the architecture of the system. The component failure probability is denoted as θ_i and the connection failure probability is represented as ψ_i. These probabilities (θ_i, ψ_i) are represented by a mean failure probability and the 95 % confidence interval of the failure probability to model the beta probability distribution of the component or the connection. The number of interactions that two-component *l* and *m* exchange in the sequence diagram *j*, is represented as $|Interact(l, m, j)|$.

The communication reliability between these two components ψ_{lmj} can be estimated using the failure probability of the connection represented as ψ_i by [1]:

$$\psi_{lmj} = (1-\theta_i)^{|Interact(l,m,j)|} \tag{3}$$

The ECRA software uses the following equation with the Bayesian reliability prediction algorithm, which is obtained by combining the data from the use case, sequence and the deployment diagrams []:

$$\theta_s = 1 - \sum_{j=1}^{k} P_j \left(\prod_{i=1}^{N} (1-\theta_i)^{bpij} * \prod_{l,j} \left(1-\psi_{lij}\right)^{|Interact(l,i,j)|} \right) \tag{4}$$

where θ_i and ψ_i are random variables that are used to produce θ_s.

By using multiple simulations, ECRA can produce a histogram of the results to represent the predicted system reliability. ECRA will also calculate the mean and 95% confidence interval of the simulation results to produce a beta curve for validation of the Bayesian model.

The input to the ECRA tool consists of the Rational Rose models that have been developed and the reliability records that have been generated. The information that is contained in the reliability record is as follows:

1. Use case Diagram: The probability of the actor using the system modeled is specified. It is important to note here that the total probability of all the actors using the system has to be equal to one. The other information is

respect to the connection probabilities, the total probability summing to be one.

2. Sequence Diagram: Every use case diagram has to correspond to a sequence diagram with a minimum of one. The other requirement placed by the software is the failure probability of confidence interval of each module.
3. Deployment Diagram: A minimum of one process for every module in the sequence diagram has to be present. Also the failure probability and confidence interval of each connection in the deployment diagram has to be specified.

The ECRA software employs MATLAB for calculations. It produces graphs as output. It calculates the parameters of prior beta distributions of each process and connector which includes:

- θ for all the components
- ψ for all the connectors
- a_i and b_i for all components and connectors.

It also provides a histogram plot of all the calculation results. A plot comparing the prior probability density function of the system failure probability θ_s and the normalized histogram from simulation observations is produced. The failure probability and the 95 % confidence level of the system are also calculated.

2. Modeling

The need for studying the reliability of a biometric device in a high performance application is the motivation for this work.

Model: 1

The following model depicts an architecture employing biometric authentication. A client computer requesting access to the remote/local machine has to undergo authentication. The login procedure validates the client based on its credentials and allows access to the resource. The resources available to the client are web server, remote server and a remote application server in addition to the local server. The access to each of these components is granted depending on the level of security clearance a client possesses.

MODEL I

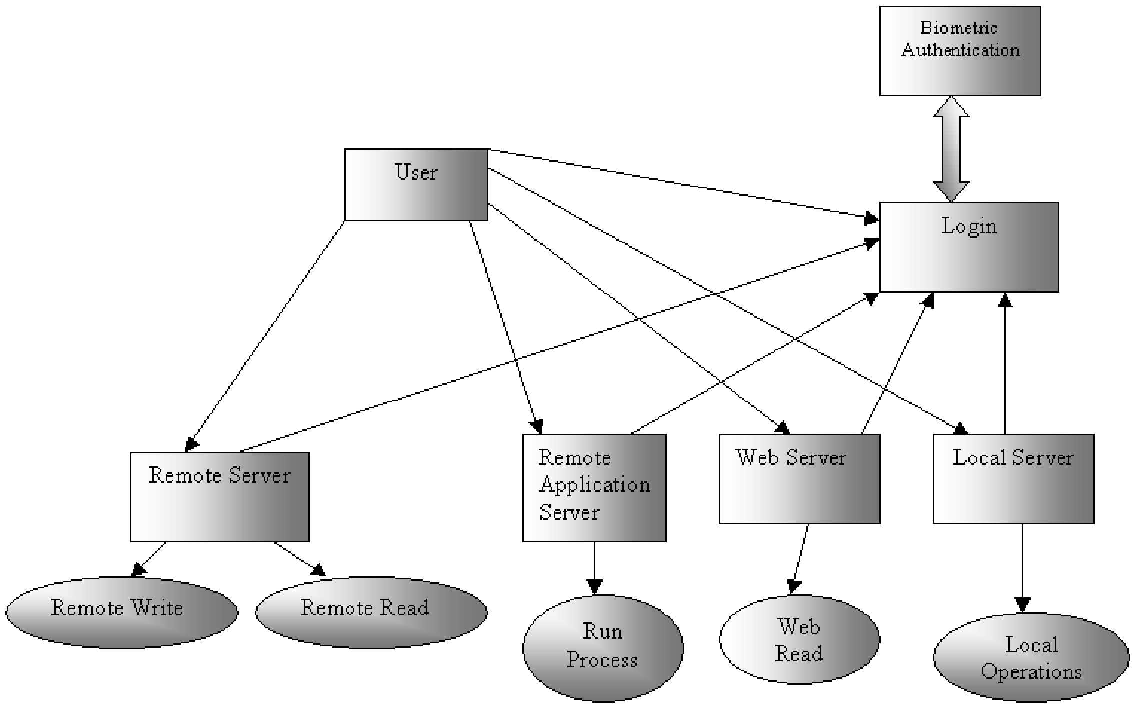


Figure 1: Biometric Application Model I

USE CASE DIAGRAM

The following is the use case diagram for the above model. There are six actors in the system namely: the user, the authentication server, the local server, the web server, remote application server and the remote server. The probability of an actor using a selected system behavior is assigned. To use the resources, the user must go through the login procedure. The login procedure includes the biometric authentication procedure. The login procedure then decides whether the resource the client has requested is available, and also ascertains the client's credentials for the same. If the resource is available and the client has clearance to use the resource then access is given to client.

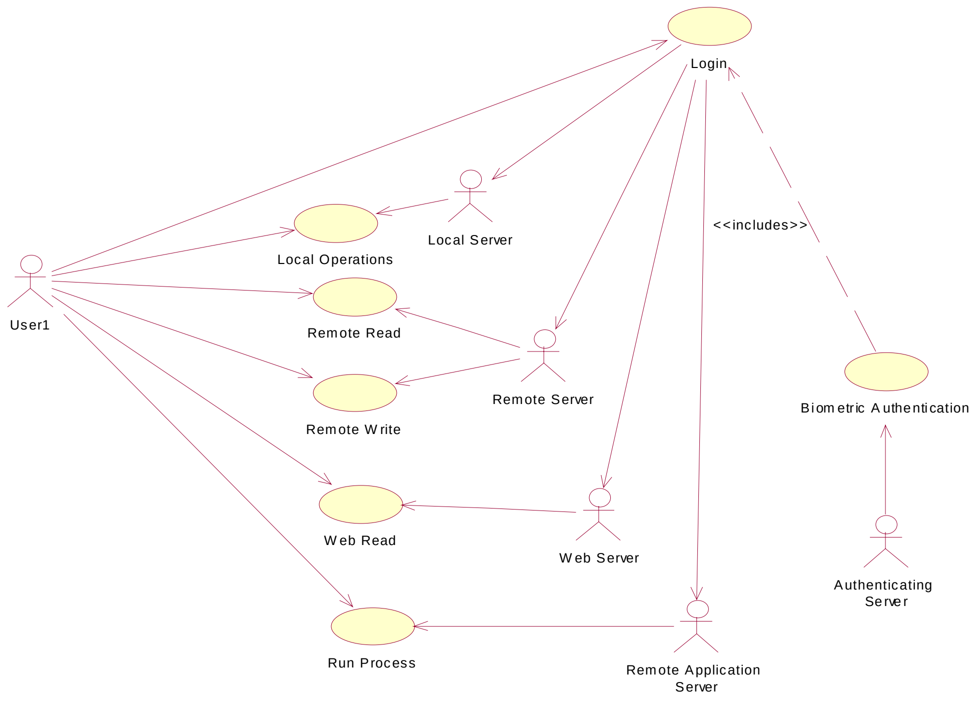


Figure 2: Use Case Diagram I

SEQUENCE DIAGRAM – LOGIN (S1)

The interaction diagram for the login procedure is shown below. The client goes through the authentication procedure. Then the procedure checks whether the client possesses the credentials for the requested service. If the client does possess the credentials then the availability of the resource requested is checked. Based on the availability the client is granted a session on the resource and the procedure logs the client on the resource.

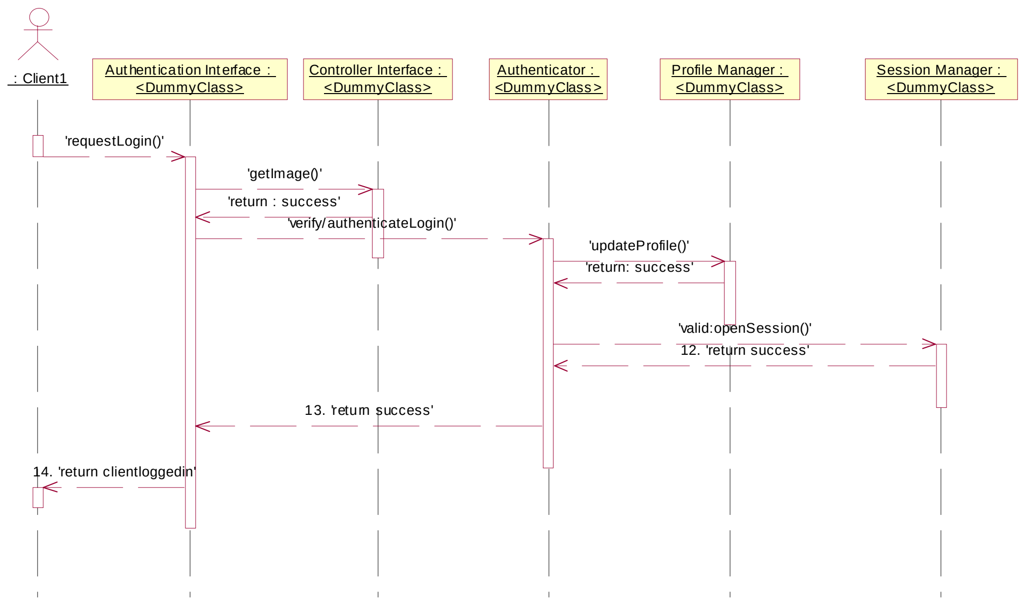

: Client1
Authentication Interface : <DummyClass>
Controller Interface : <DummyClass>
Authenticator : <DummyClass>
Profile Manager : <DummyClass>
Session Manager : <DummyClass>
'requestLogin()'
'getImage()'
'return : success'
'verify/authenticateLogin()'
'updateProfile()'
'return: success'
'valid:openSession()'
12. 'return success'
13. 'return success'
14. 'return clientloggedin'

Figure 3: Sequence Diagram - Login

SEQUENCE DIAGRAM – BIOMETRIC AUTHENTICATION (S2)

The interaction diagram shows biometric authentication. The login procedure starts of the interaction by requesting authentication. The authenticator then initiates the authentication procedure. The sample biometric from the client is procured and converted to a template. Then matching is performed to verify the client and the interaction is completed by either a success or a failure.

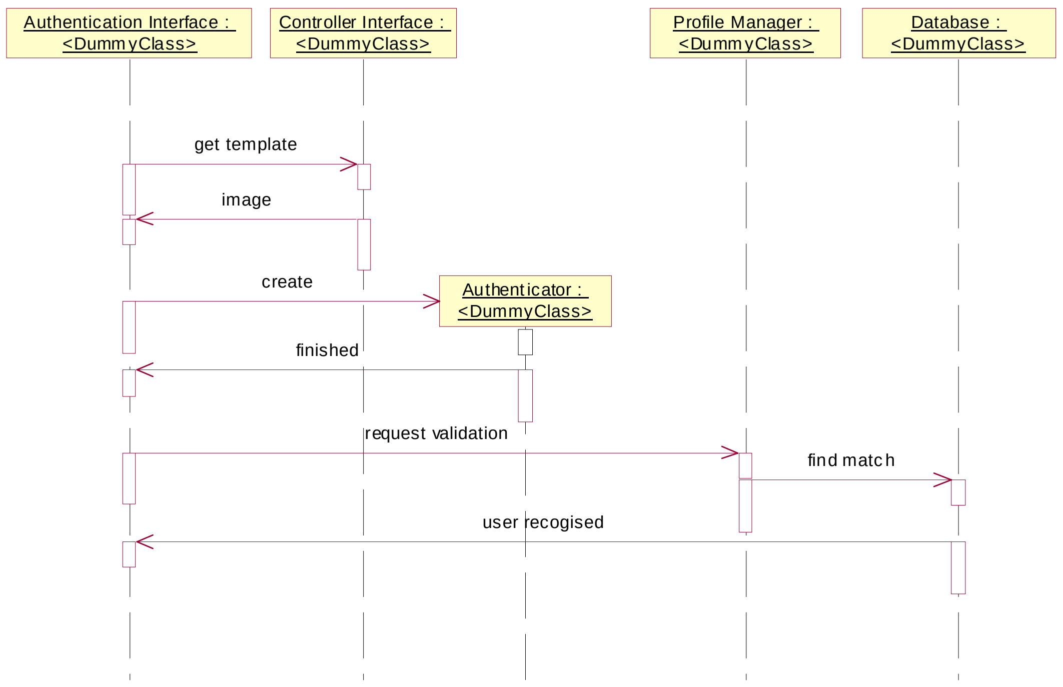

Authentication Interface : <DummyClass>
Controller Interface : <DummyClass>
Profile Manager : <DummyClass>
Database : <DummyClass>
get template
image
create
Authenticator : <DummyClass>
finished
request validation
find match
user recogised

Figure 4: Sequence Diagram - Biometric Authentication

SEQUENCE DIAGRAM – LOCAL OPERATIONS (S3)

This procedure shows how the local operations are performed. The client has to follow the login procedure. The client gets access to the local server after a successful login. After login, the client performs the processing. In this scenario it can be seen that client process is looking for some information on the local server. A search is performed and the data requested is sent back to client.

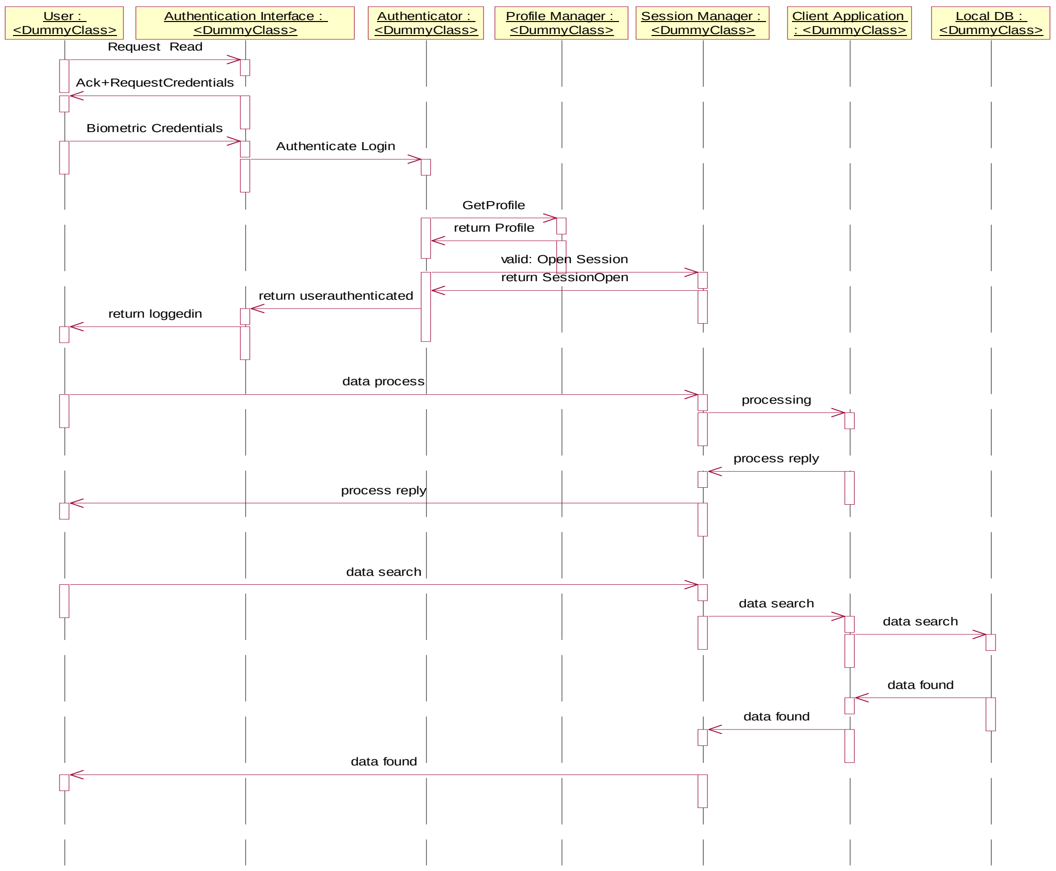


Figure 5: Sequence Diagram - Local Operations

SEQUENCE DIAGRAM – WEB READ (S4)

In this diagram a user is employing the web services in the system. First the user has to undergo authentication. After successful login to the system, the user can request for resources. The client application sends a request to the session manager for the web resource. The session manger in turn verifies the user's profile. Based on the user's credentials, the session manager forwards the session request to the web interface. If the resource is available, a session is granted to the user.

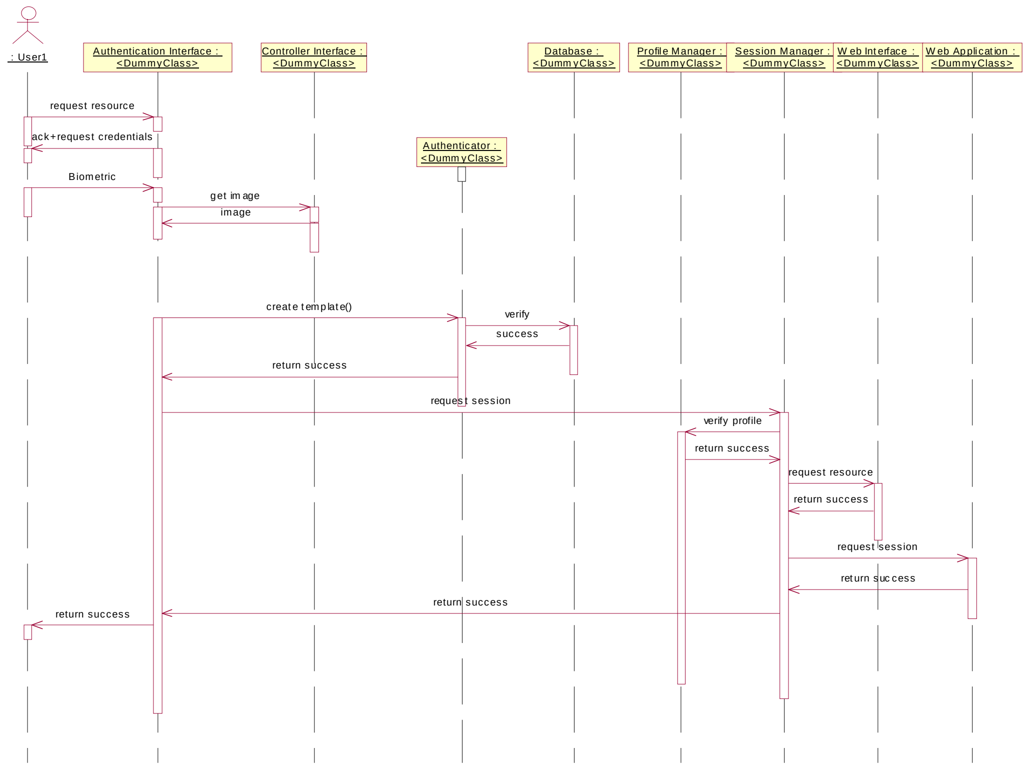


Figure 6: Sequence Diagram - Web Read

SEQUENCE DIAGRAM – REMOTE READ (S5)

The following interaction diagram shows a client performing a remote read operation. The initial procedure is login. The client process undergoes authentication and credentials check. The resource availability is ascertained. Then the client is granted access to the remote server.

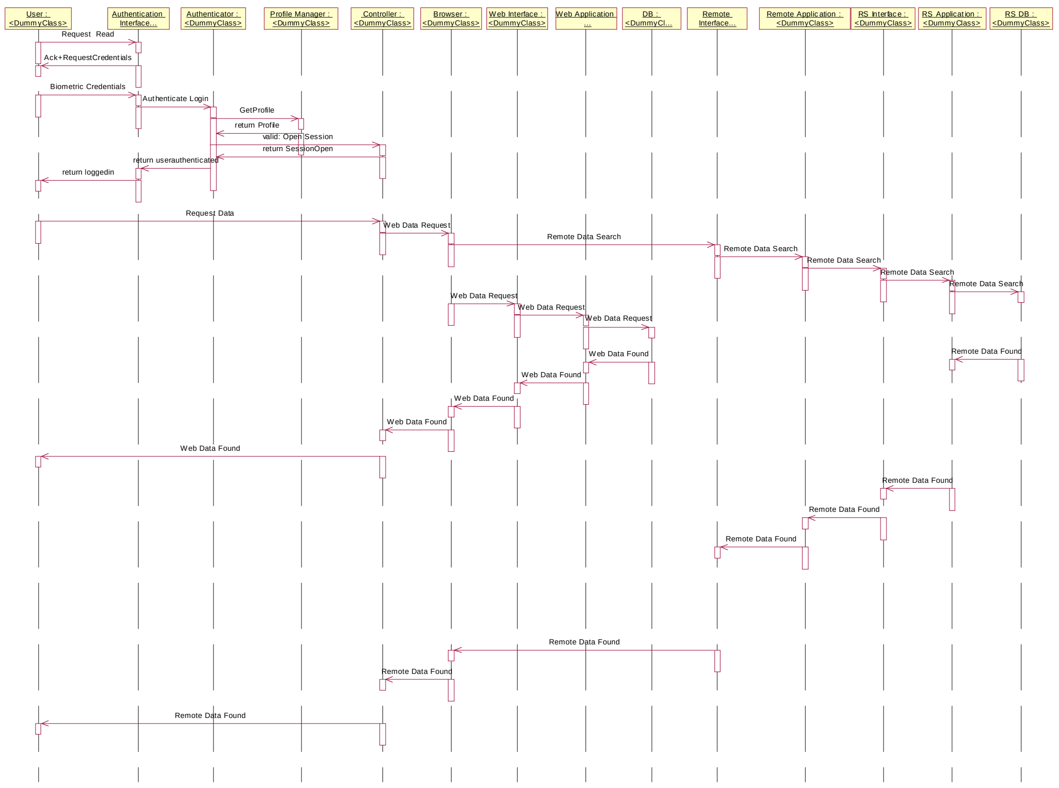


Figure 7: Sequence Diagram - Remote Read

SEQUNCE DIAGRAM – REMOTE WRITE (S6)

This interaction diagram shows how a remote write is performed. The initial steps are the same. Following the success of the authentication and login procedure the client obtains a session on the remote machine. At first, the client requests for data from the server. Then it performs processing on the data that was obtained. Then the processed data is uploaded.

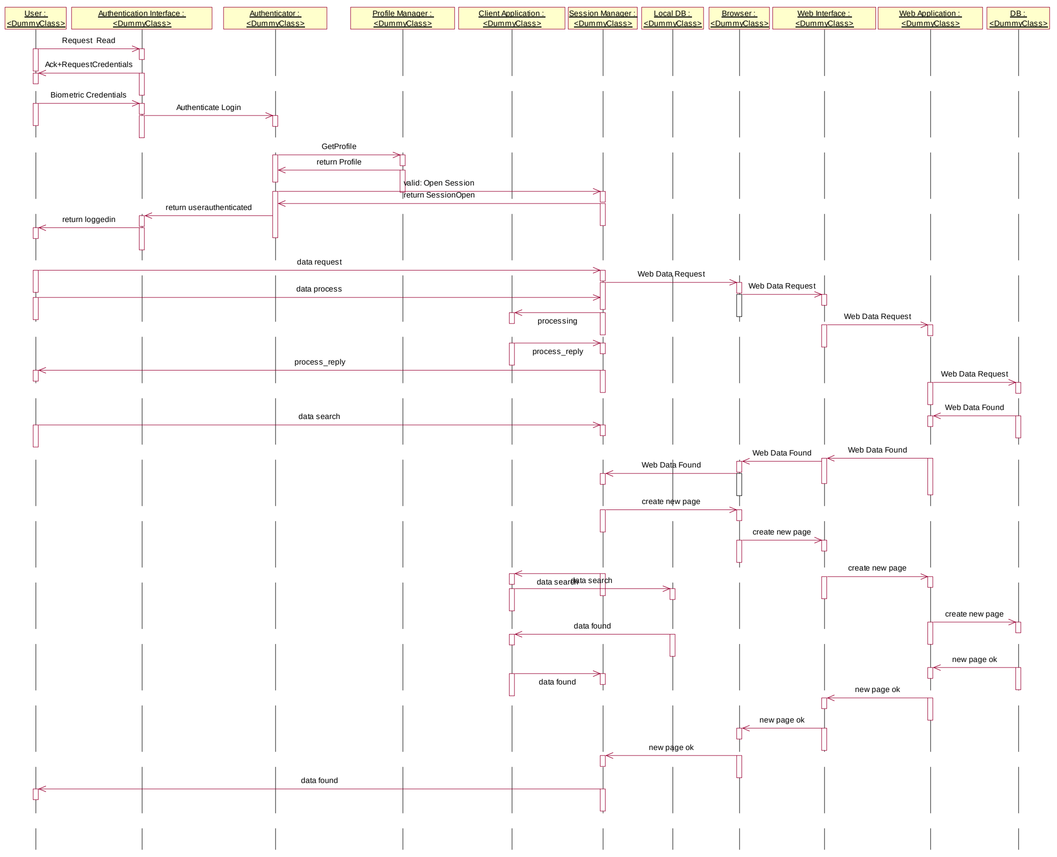


Figure 8: Sequence Diagram - Remote Write

SEQUENCE DIAGRAM – RUN PROCESS (S7)

The following sequence diagram shows a user requesting to execute a program on the remote application server. The user undergoes biometric authentication to login into the system. Then based on the user's security clearance the user is granted access to the remote application server.

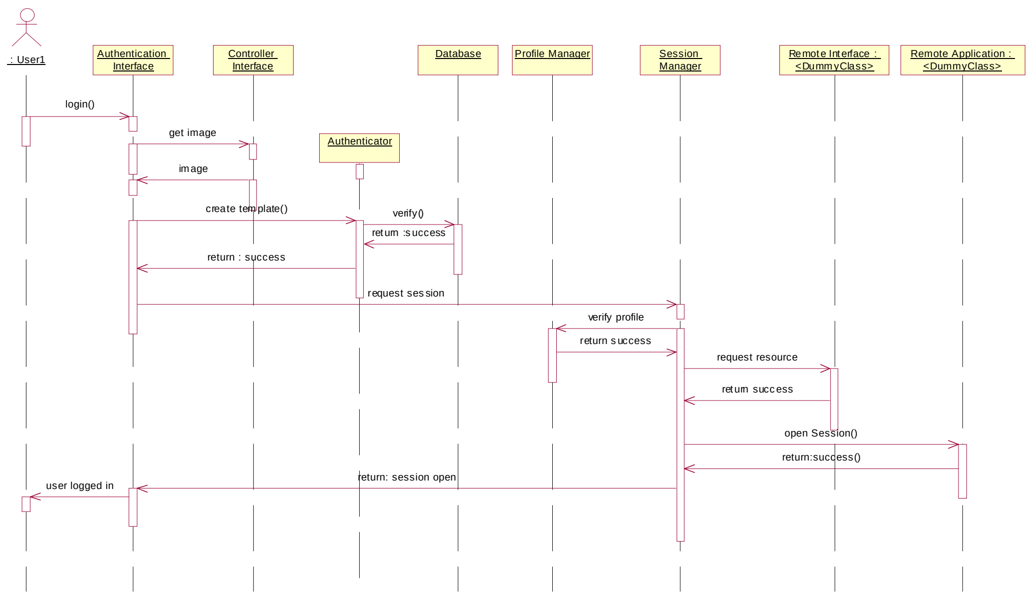


Figure 9: Sequence Diagram - Run Process

DEPLOYMENT DIAGRAM

The deployment diagram depicts the physical resources in the system, including the nodes, components, and associations. A node represents the piece of hardware. A component denotes the software entity and the associations indicate the line of communication between the hardware elements. For this model, there are five nodes namely: Client, Controller, Web Server, Remote Server and Remote Application Server. The connections are denoted as psi1 through psi4.

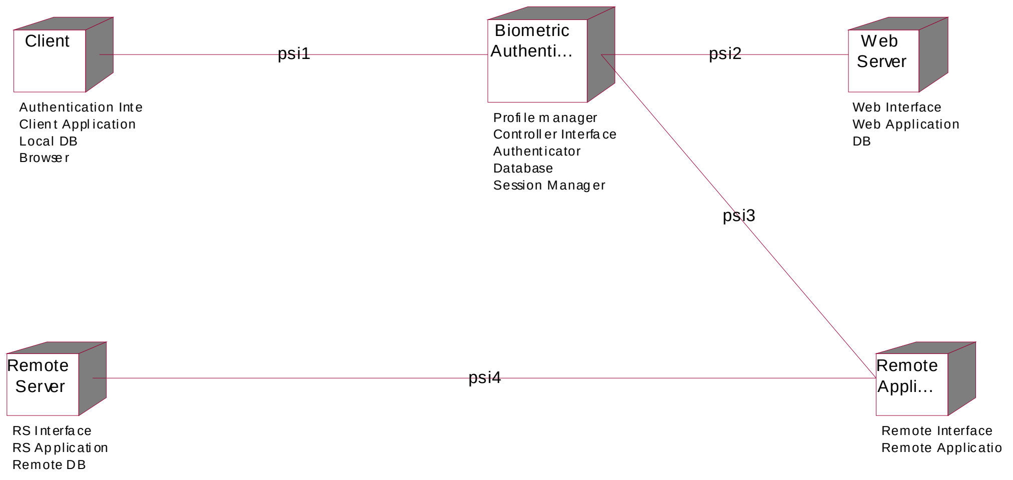


Figure 10: Deployment Diagram I

The following table summarizes an annotation for each component in the case study. The record includes the name of the component, its site, the mean failure probability and its 95% confidence interval. The component reliability information records for the above model are as follows:

TABLE I

Component	Name	Failure Probability	Confidence Interval	Number of Busy Periods						
				S1	S2	S3	S4	S5	S6	S7
C1	Authentication Interface	0.01	(0.007, 0.013)	3	4	3	5	3	3	5
C2	Client Application	0.007	(0.003, 0.01)	0	0	3	0	0	3	0
C3	Local DB	0.003	(0.001,0.005)	0	0	1	0	0	1	0
C4	Browser	0.009	(0.006, 0.012)	0	0	0	0	3	4	0
C5	Authenticator	0.003	(0.001, 0.005)	3	1	3	2	3	3	2
C6	Profile Manager	0.009	(0.006,0.012)	1	1	1	1	1	1	1
C7	Session Manager	0.005	(0.003,0.007)	1	0	0	4	0	0	1
C8	Controller Interface	0.005	(0.003, 0.007)	0	1	5	1	4	8	1
C9	Database	0.009	(0.006,0.012)	0	1	0	1	0	0	1
C10	Web Interface	0.005	(0.003,0.007)	0	0	0	1	2	4	0
C11	Web Application	0.039	(0.025,0.054)	0	0	0	1	2	4	0
C12	DB	0.005	(0.003,0.007)	0	0	0	0	1	2	0
C13	Remote Interface	0.007	(0.003, 0.01)	0	0	0	0	2	0	1
C14	Remote Application	0.005	(0.003, 0.007)	0	0	0	0	2	0	1
C15	RS Interface	0.01	(0.007, 0.013)	0	0	0	0	2	0	0
C16	RS Application	0.005	(0.003, 0.007)	0	0	0	0	2	0	0
C17	Remote DB	0.01	(0.007, 0.013)	0	0	0	0	1	0	0
psi1	(Client, Biometric Authenticator)	0.009	(0.006, 0.012)	-	-	-	-	-	-	-
psi2	(Biometric Authenticator, Web Server)	0.005	(0.003,0.007)	-	-	-	-	-	-	-
psi3	(Biometric Authenticator, RAS)	0.003	(0.001,0.005)	-	-	-	-	-	-	-
psi4	(RAS, Remote Server)	0.007	(0.003, 0.01)	-	-	-	-	-	-	-

Using the UML annotations and the component reliability record from Table I, three sets of experiments were performed. The following section shows the results for the same.

3. Results for Model I

<u>EXPERIMENT I</u>

The first experiment, the reliability of the biometric device (authentication site) was varied from 0.7 to 0.99 in steps of 0.05. The failure probability and the 95 % confidence level of the system are also calculated.

<u>Results for Experiment I</u>

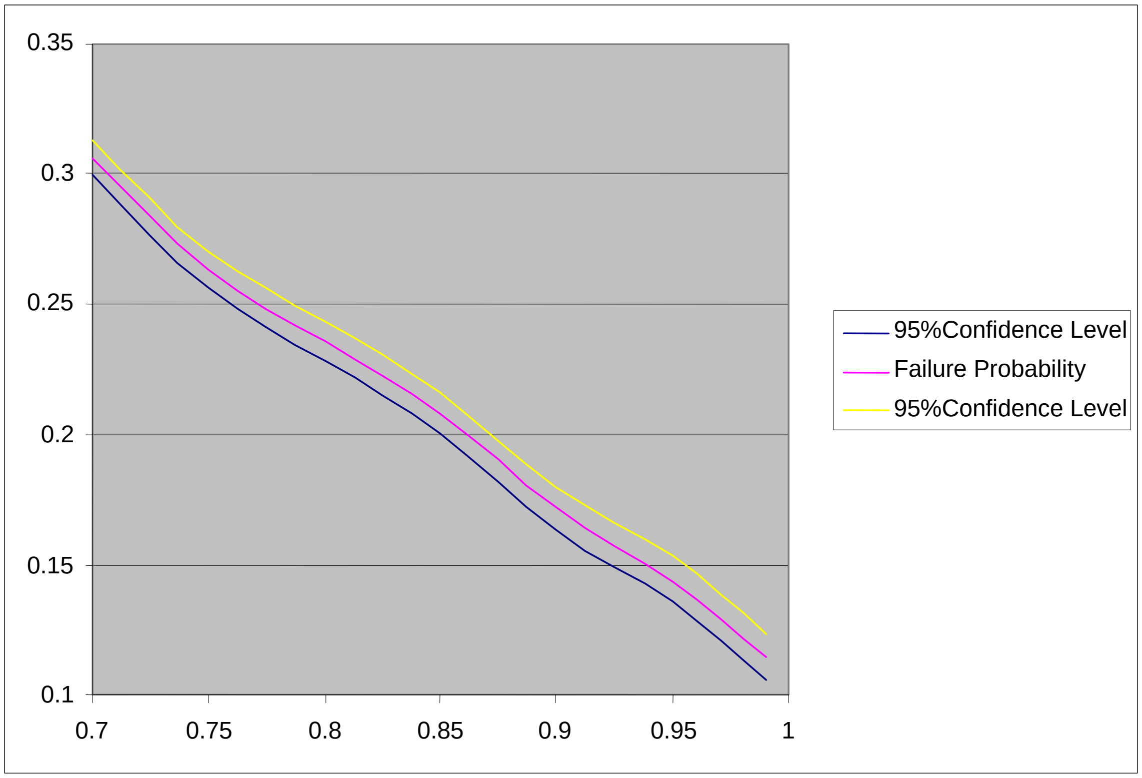


Figure 11: Results for Experiment I –Model I

The failure probability drops down steadily as the reliability of the device increases. In this case the failure probability varies from 0.3 to 0.1.

EXPERIMENT II

The second experiment, four authentication sites were considered. In this case the four device's reliability was varied from 0.7 to 0.99 in steps of 0.05.

Results for Experiment II

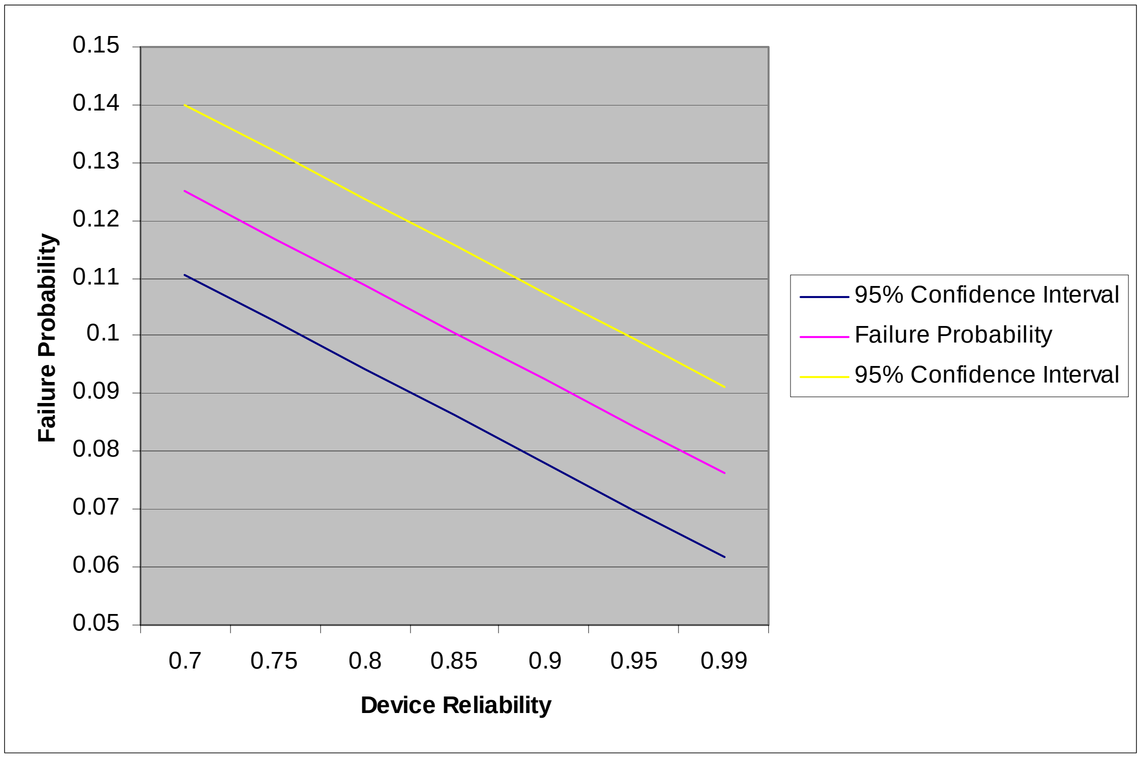


Figure 12: Results for Experiment II – Model I

The failure probability of the four devices decreases as the reliability of the device increases. The failure probability decreases from 0.14 to 0.08 when the device reliability is increased from 0.7 to 0.99 for the four devices.

EXPERIMENT III

For the third experiment, four authentication sites were considered. Of the four the device reliability for three sites were fixed at 0.8, 0.85 and 0.95. The reliability of the fourth site was varied from 0.7 to 0.99 in steps of 0.05

Results for Experiment III

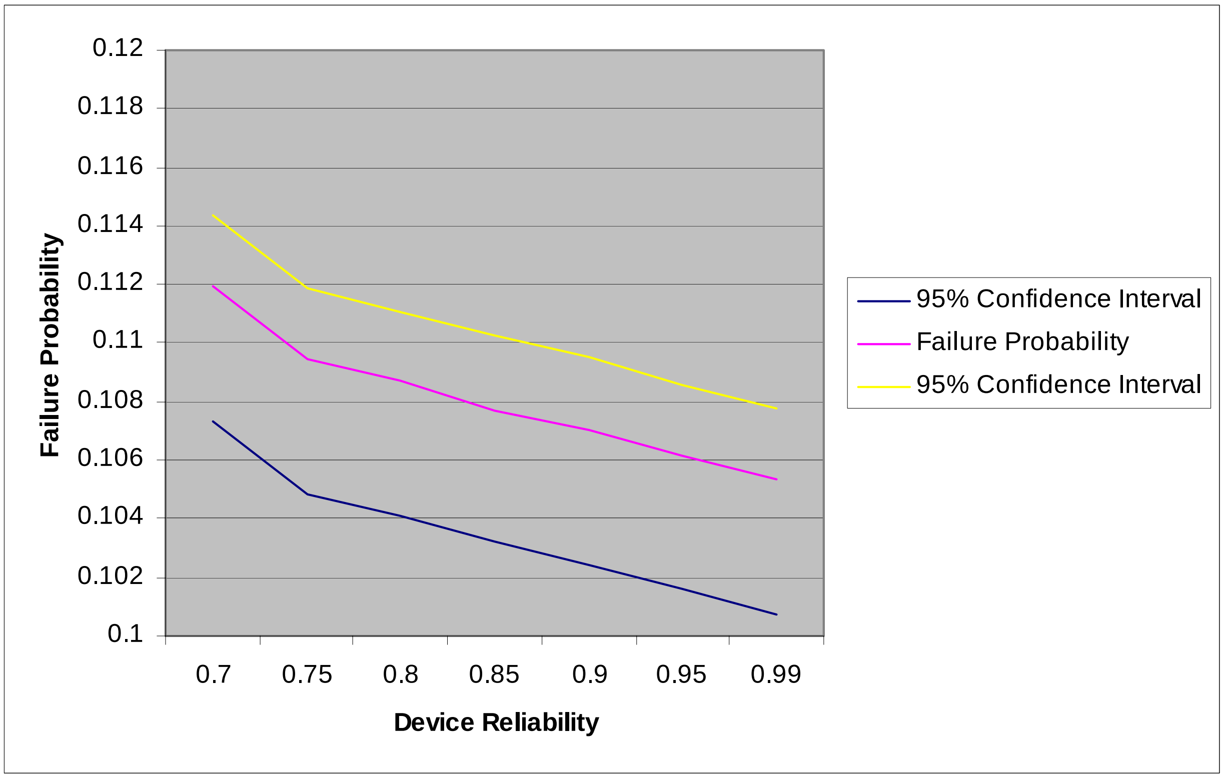


Figure 13: Results for Experiment III – Model I

Having three devices with fixed reliability and varying the reliability of the fourth device gives the above characteristic. The variation in the system failure probability is from 0.112 to 0.107. The three experiments give us the knowledge of system failure probability based on the device reliability.

MODEL II

This model shows a biometric system in the enrollment and authentication modes. Also the biometric device is broken into its sub-systems.

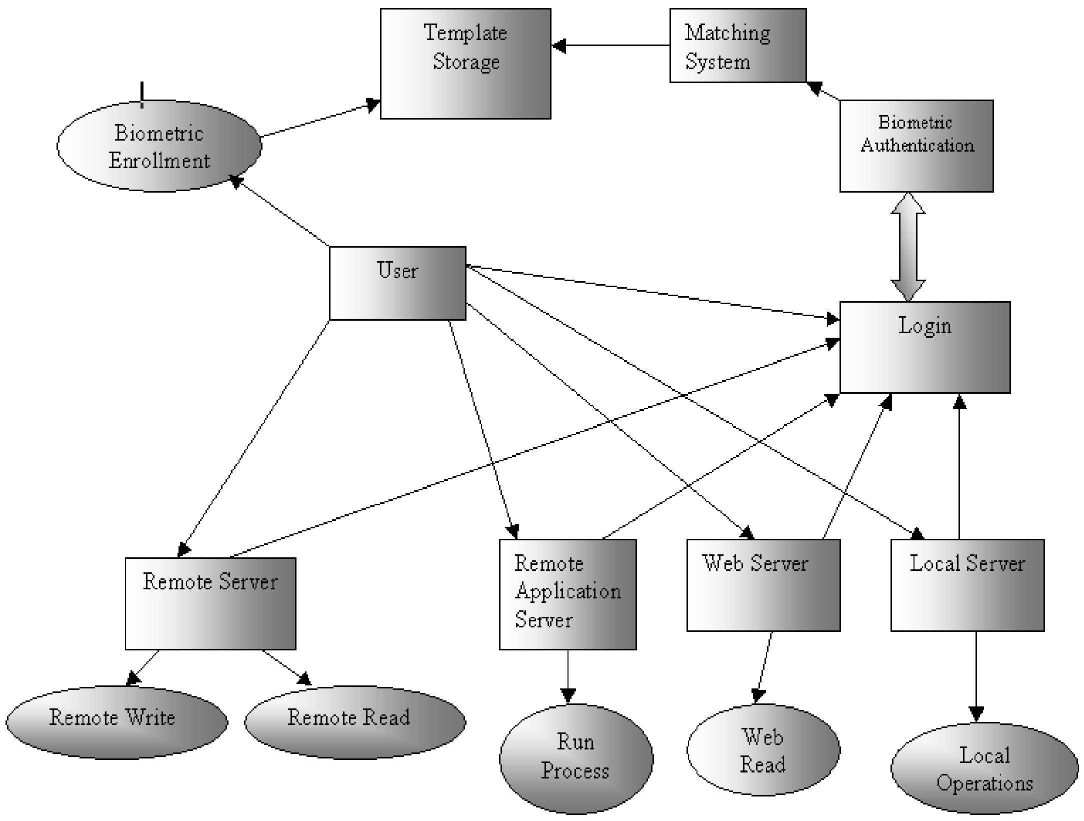


Figure 14: Biometric Application Model II

The principle of biometric authentication is as follows: for a computer to recognize a person (whom they claim to be), the system will need to perform a comparison between a sample of biometric and a sample that was taken earlier, which was stored in a database (Biometric Enrollment). The biometric device is split into subsystems namely: a template creation subsystem, a database, and a matching subsystem. For the authentication procedure the client produces the sample biometric to the scanner. A template is created from this and it is compared to the templates stored in the database. If a match is found then the user is authenticated. The following use case diagram depicts the same.

USE CASE DIAGRAM

There are seven actors in the system namely: the user, the local server, remote server, remote application server, web server, the matching system and the template storage system. For a client to use a resource, the authentication procedure must be carried out. Based on the client’s credentials and the availability of the resource, the client can employ the resources. The probability of an actor using a selected system behavior is assigned.

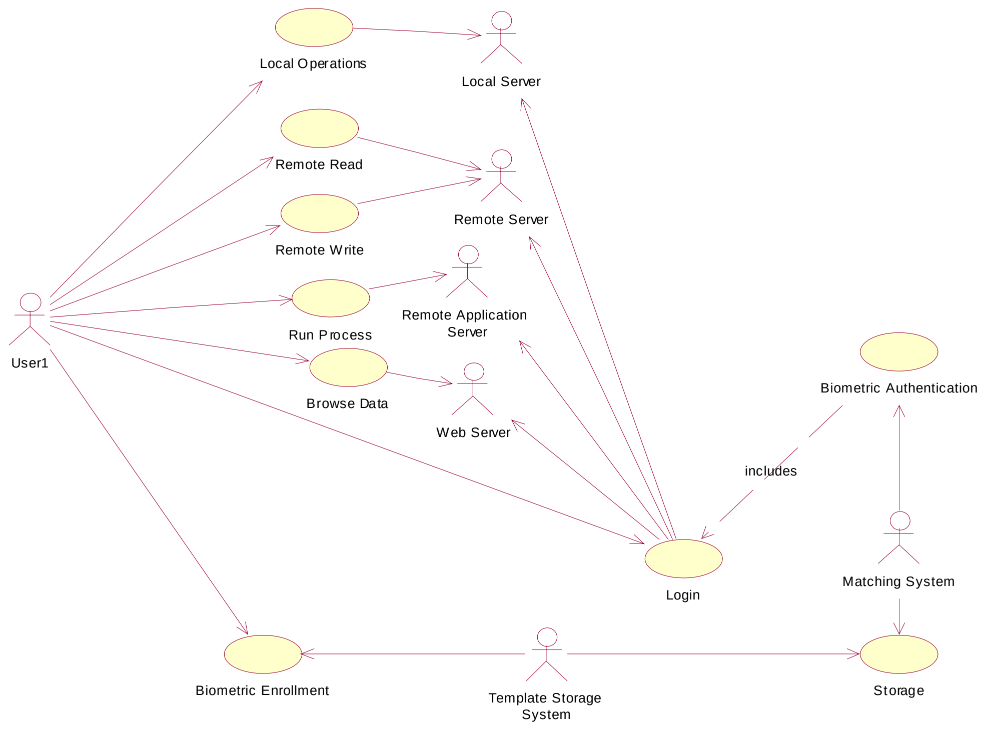


Figure 15: Use Case Diagram Model II

SEQUENCE DIAGRAM – BIOMETRIC ENROLLMENT (S1)

A new user to the system must undergo biometric enrollment. The user presents the live biometric to the sensor. The image is captured and features are extracted. Based on these features, a biometric information record (BIR) is created from the template and this is stored in a database. The user undergoes this process once. This process of registering a new user to the system is called biometric enrollment.

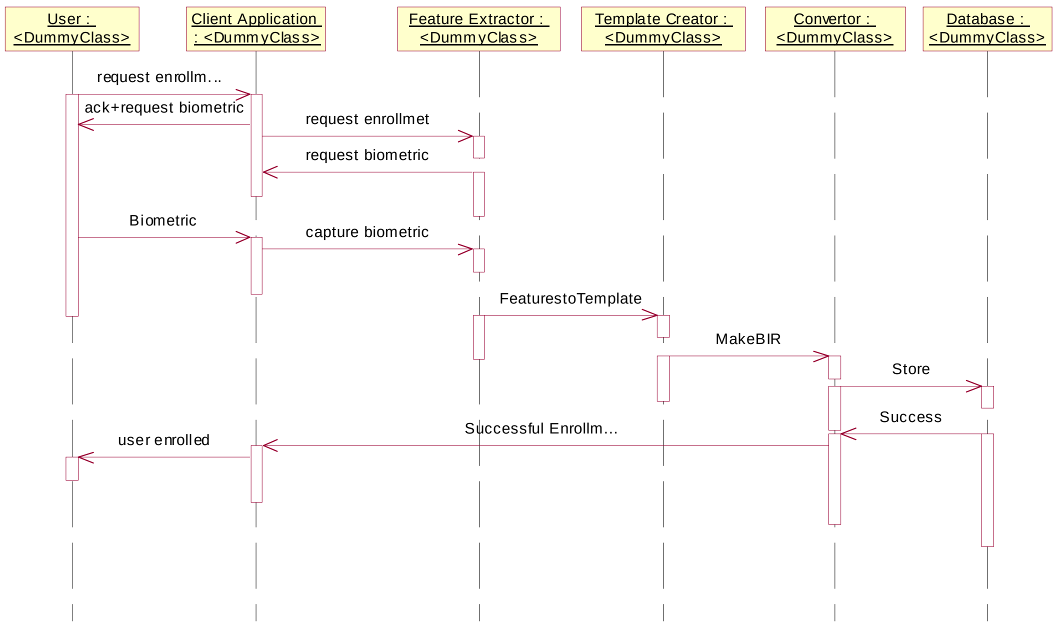


Figure 16: Sequence Diagram - Biometric Enrollment

SEQUENCE DIAGRAM – STORAGE (S2)

The following interaction diagram depicts how a biometric template is stored into the database. A sample biometric is procured from the user and the features are extracted. Based on the extracted features, a template is created. For reasons like interoperability and compatibility, the template is converted into a BIR. This is stored into the database.

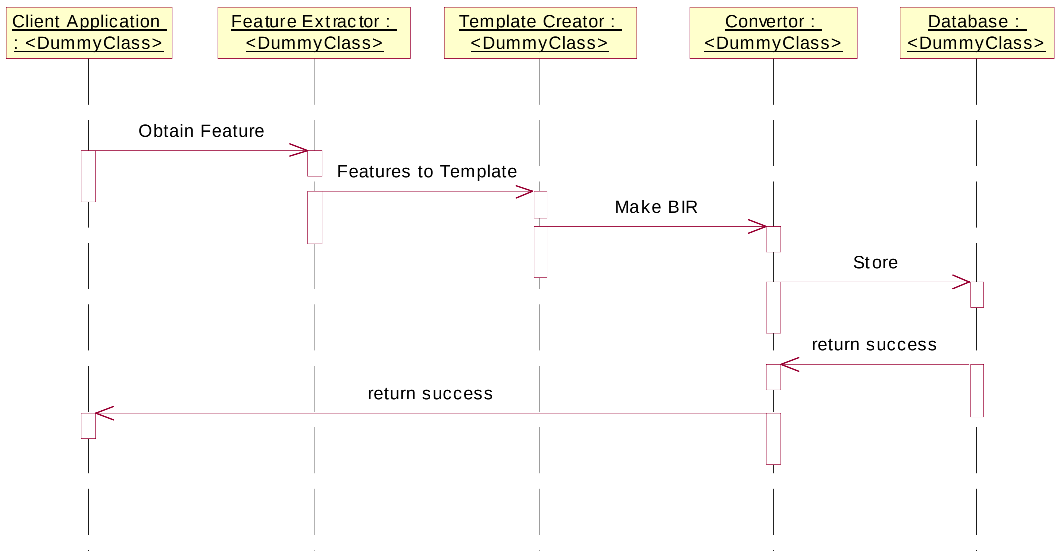


Figure 17: Sequence Diagram – Storage

SEQUENCE DIAGRAM – LOGIN (S3)

The login procedure is necessary for any operation on the system. A user requesting access to a resource must undergo authentication. The sequence diagram shows a simple login scheme. Depending on the user's resource requirement, a session will be granted to the user.

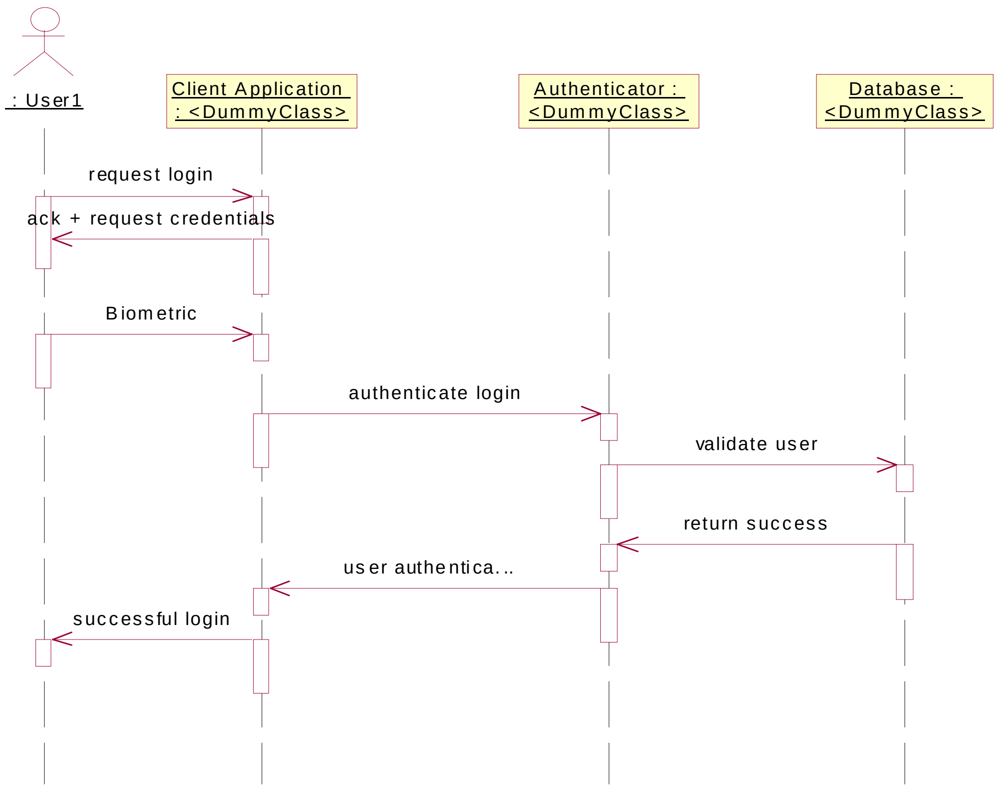


Figure 18: Sequence Diagram – Login

SEQUENCE DIAGRAM – BIOMETRIC AUTHENTICATION (S4)

The interaction diagram shows biometric authentication. The user starts of the interaction by requesting authentication. The authentication interface then initiates the authentication procedure. First the live biometric from the client is procured. The image obtained is then converted into a template. Then matching is performed to verify the client and the interaction is completed by either a success or a failure.

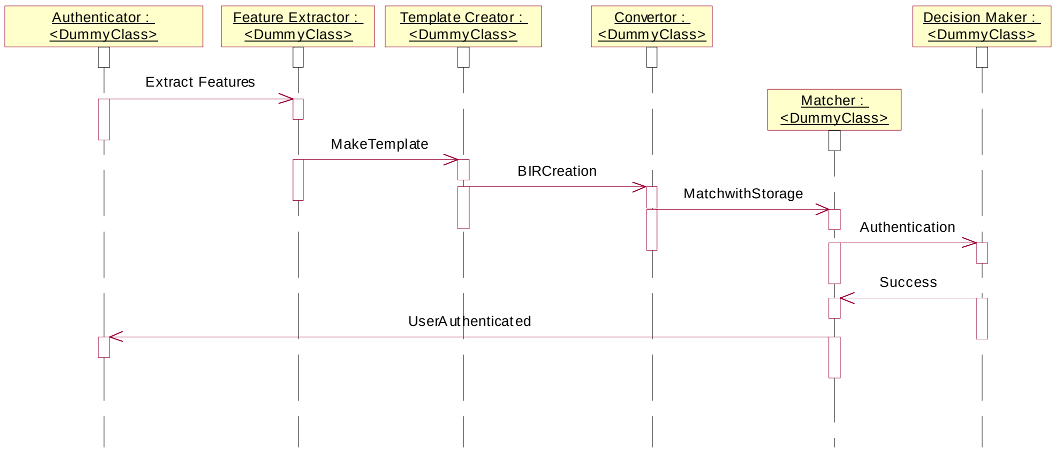


Figure 19: Sequence Diagram - Biometric Authentication II

SEQUENCE DIAGRAM – LOCAL OPERATIONS (S5)

The following sequence diagram shows how local operations are performed. The user presents the credentials to the authentication system and requests access to the local server. The validity of the credentials is checked and the user is given access.

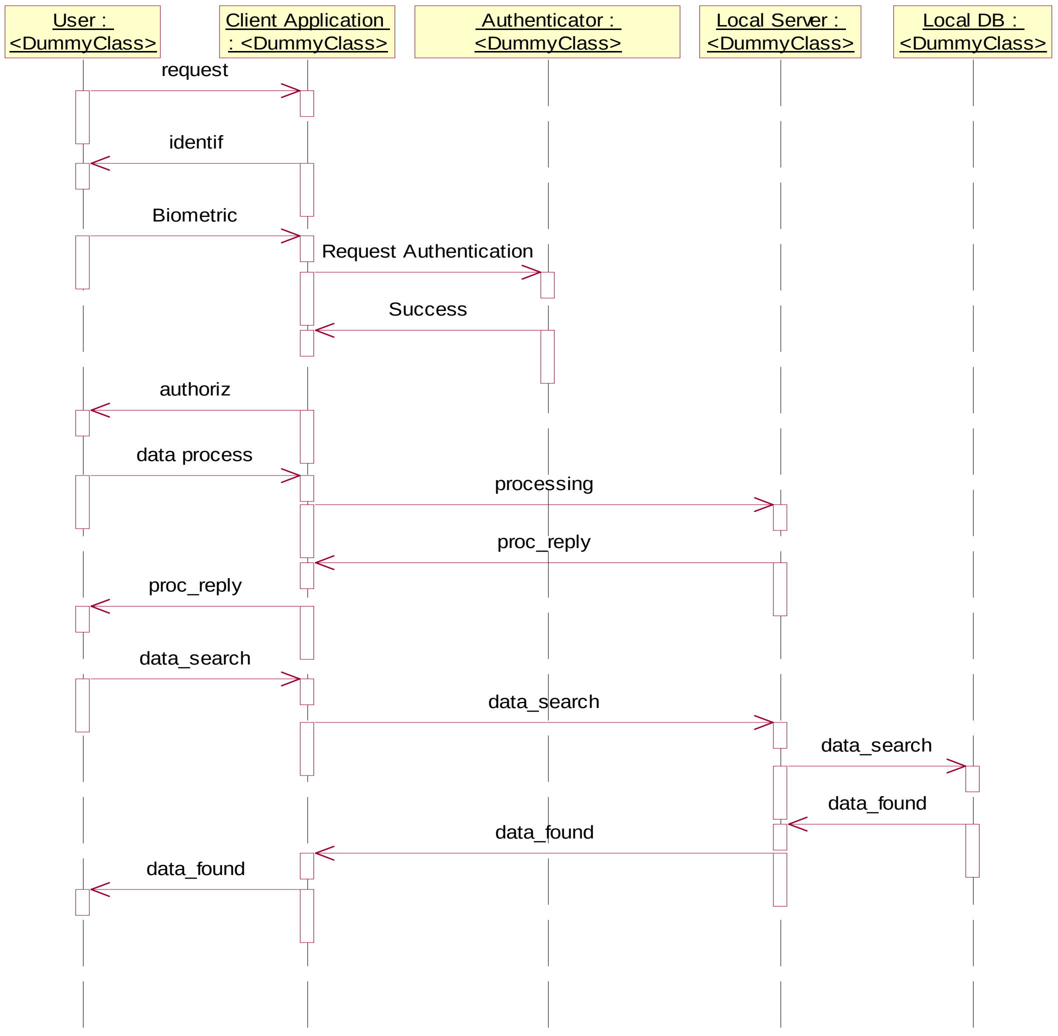


Figure 20: Sequence Diagram - Local Operations

SEQUENCE DIAGRAM – REMOTE READ (S6)

The following interaction diagram shows a client performing a remote read operation. The initial procedure is login. The client process undergoes authentication and credentials check. The resource availability is ascertained. Then the client is granted access to the remote server.

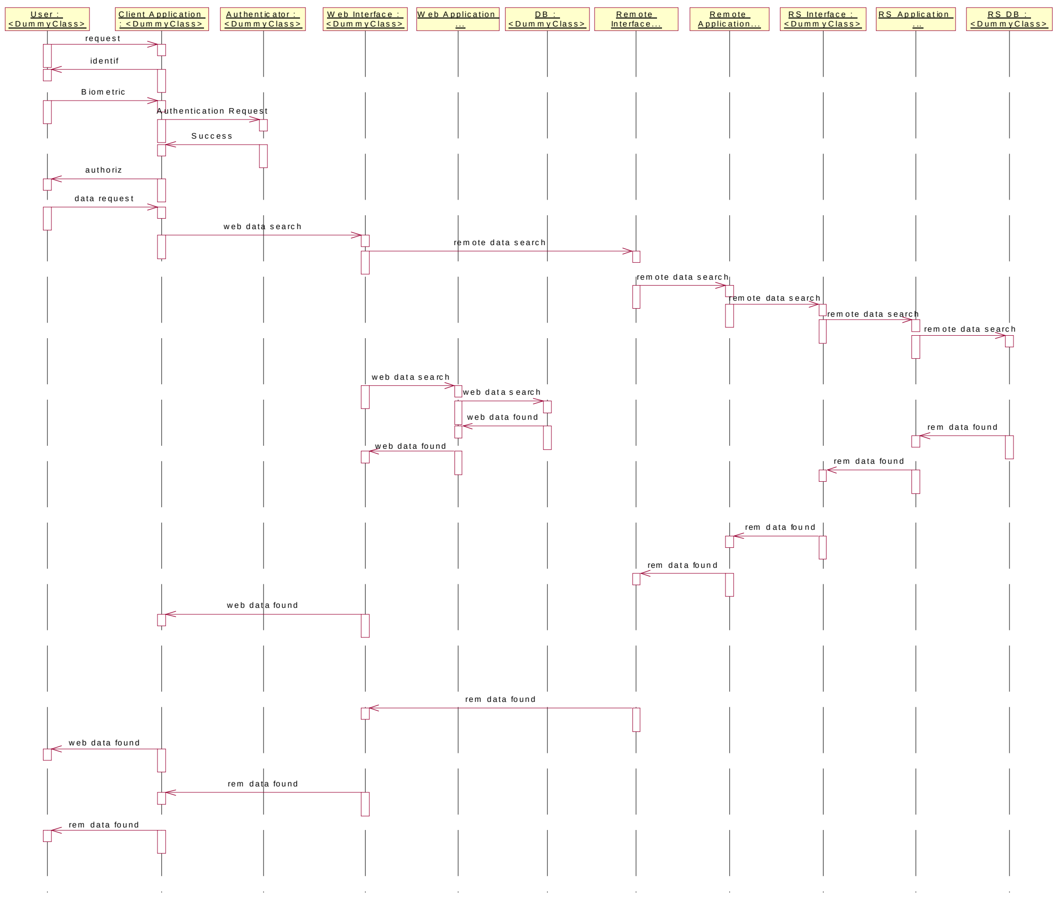


Figure 21: Sequence Diagram - Remote Read

<u>SEQUENCE DIAGRAM – REMOTE WRITE (S7)</u>

This interaction diagram shows how a remote write is performed. The initial steps are the same. Following the success of the authentication and login procedure the client obtains a session on the remote machine. At first, the client requests for data from the server. Then it performs processing on the data that was obtained. Then the processed data is uploaded.

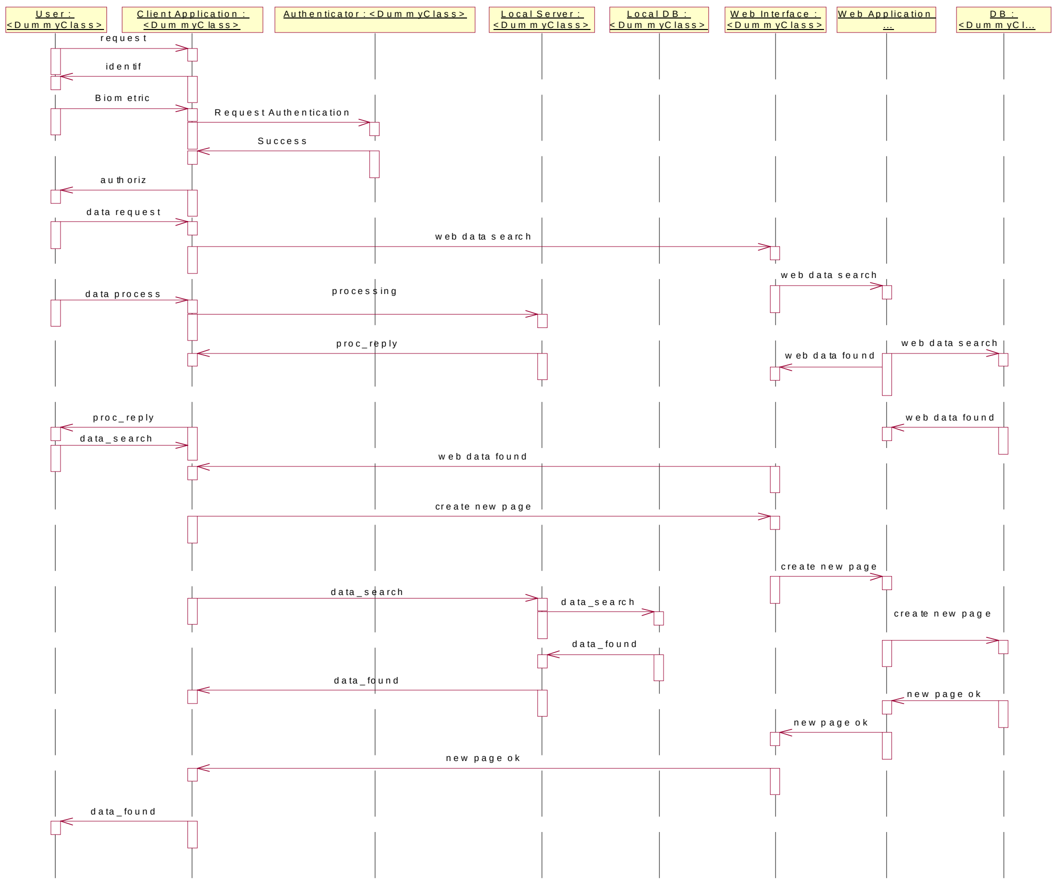


Figure 22: Sequence Diagram - Remote Write

SEQUENCE DIAGRAM – BROWSE DATA (S8)

In this diagram a user is employing the web services in the system. First the user has to undergo authentication. After successful login to the system, the user can request for resources. The client application sends a request to the web interface for the web resource. Based on availability and the user's credentials a session is granted to the user.

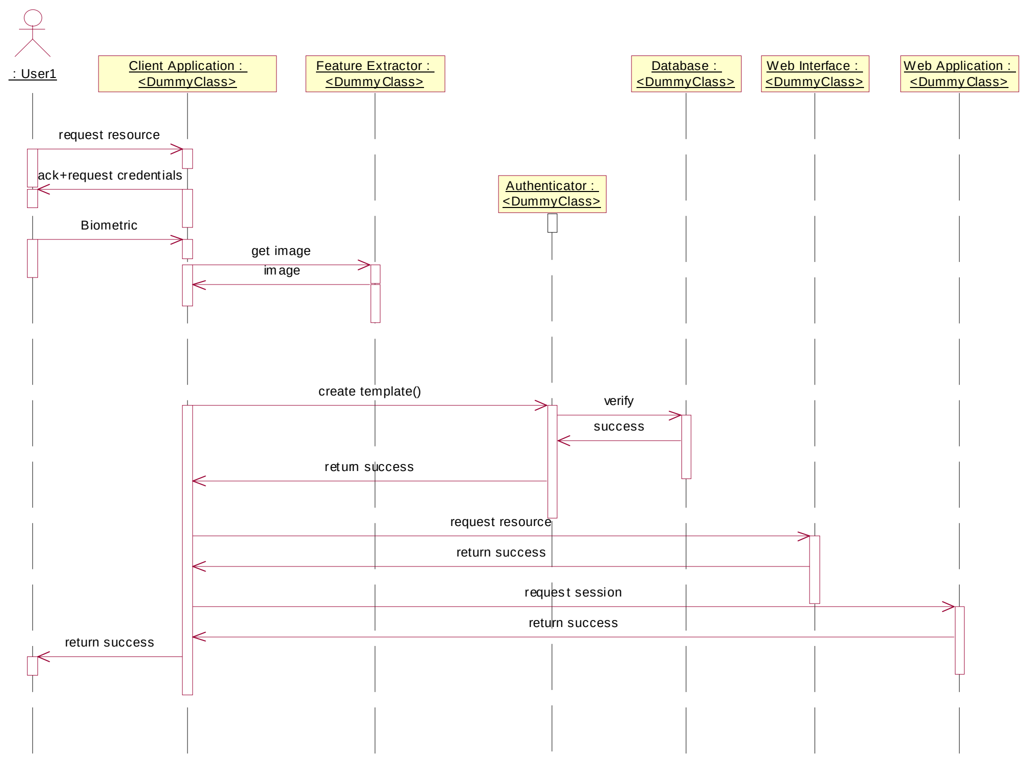


Figure 23: Sequence Diagram - Browse Data

SEQUENCE DIAGRAM – RUN PROCESS (S9)

The following sequence diagram shows a user requesting to execute a program on the remote application server. The user undergoes biometric authentication to login into the system. Then based on the user's security clearance the user is granted access to the remote application server.

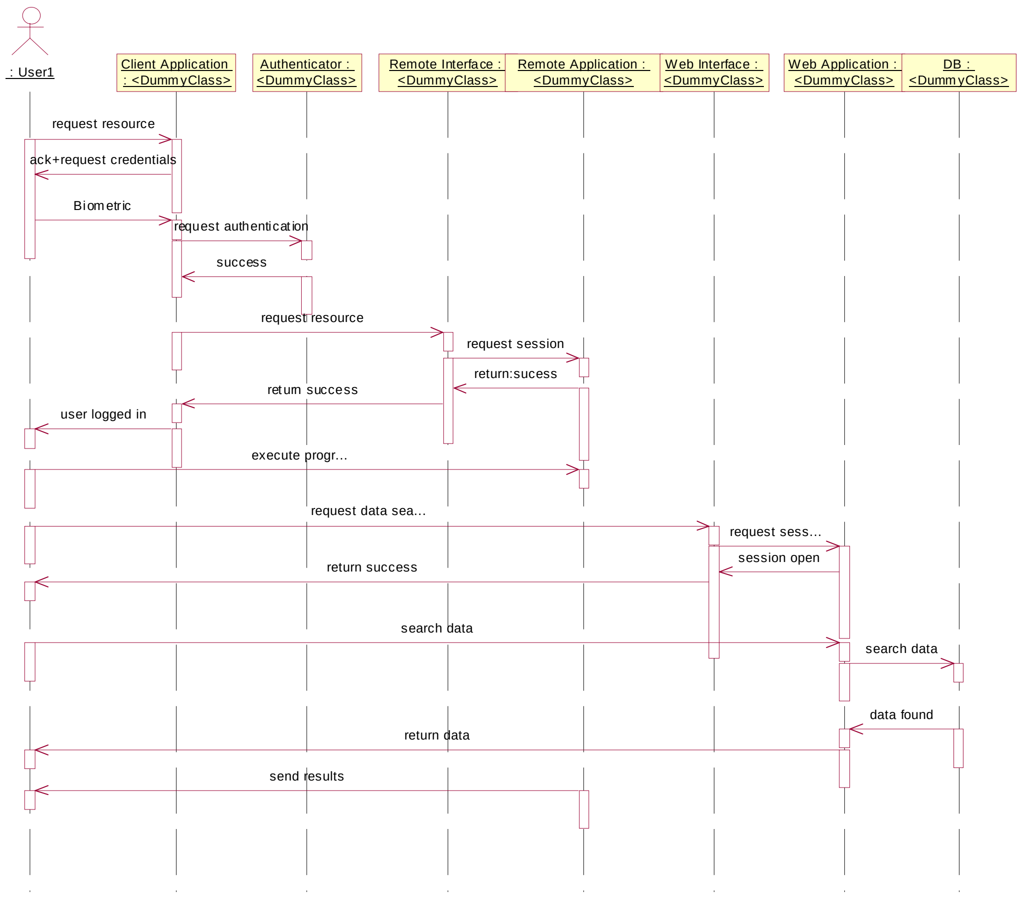


Figure 24: Sequence Diagram - Run Process

DEPLOYMENT DIAGRAM

For this model, there are five nodes namely: client, local server, service provider and the grid portals. The components are denoted by C1 through C12 and psi1 through psi3 indicate associations.

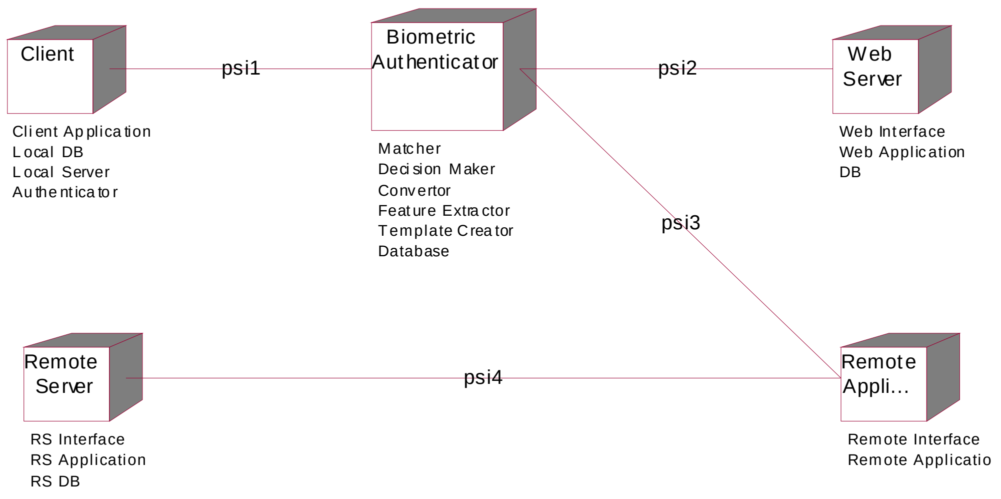


Figure 25: Deployment Diagram II

The following table summarizes an annotation for each component in the case study. The record includes the name of the component, its site, the mean failure probability and its 95% confidence interval. The component reliability information records for the above model are as follows:

TABLE II

Component	Name	Failure Probability	Confidence Interval	Number of Busy Periods								
				S1	S2	S3	S4	S5	S6	S7	S8	S9
C1	Client Application	0.007	(0.005, 0.009)	4	1	3	0	7	6	10	6	4
C2	Local Server	0.009	(0.006, 0.012)	0	0	2	0	3	0	3	0	0
C3	Local DB	0.007	(0.005, 0.009)	0	0	0	0	1	0	1	0	0
C4	Authenticator	0.01	(0.007,0.013)	0	0	0	1	1	1	1	2	1
C5	Feature Extractor	0.005	(0.003, 0.007)	2	1	0	1	0	0	0	1	0
C6	Template Creator	0.003	(0.001, 0.005)	1	1	0	1	0	0	0	0	0
C7	Convertor	0.01	(0.007, 0.013)	2	2	0	1	0	0	0	0	0
C8	Matcher	0.009	(0.006, 0.012)	0	0	0	2	0	0	0	0	0
C9	Decision Maker	0.005	(0.003, 0.007)	0	0	0	1	0	0	0	0	0
C10	Database	0.007	(0.005, 0.009)	1	1	1	0	0	0	0	1	0
C11	Web Interface	0.039	(0.025, 0.054)	0	0	0	0	0	3	4	1	2
C12	Web Application	0.009	(0.006, 0.012)	0	0	0	0	0	2	4	1	3
C13	DB	0.007	(0.005, 0.009)	0	0	0	0	0	1	2	0	1
C14	RS Interface	0.039	(0.025, 0.054)	0	0	0	0	0	2	0	0	0
C15	RS Application	0.005	(0.003, 0.007)	0	0	0	0	0	2	0	0	0
C16	RS DB	0.007	(0.005, 0.009)	0	0	0	0	0	1	0	0	0
C17	Remote Interface	0.009	(0.006, 0.012)	0	0	0	0	0	2	0	0	2
C18	Remote Application	0.003	(0.001, 0.005)	0	0	0	0	0	2	0	0	2
psi1	(Client, Biometric Authenticator)	0.009	(0.006, 0.012)	-	-	-	-	-	-	-	-	-
psi2	(Biometric Authenticator, Web Server)	0.005	(0.003, 0.007)	-	-	-	-	-	-	-	-	-
psi3	(Biometric Authenticator, RAS)	0.003	(0.001, 0.005)	-	-	-	-	-	-	-	-	-
psi4	(RAS, Remote Server)	0.007	(0.005, 0.009)	-	-	-	-	-	-	-	-	-

4. Results for Model II

EXPERIMENT I

In this study, three experiments for each model were performed. The first experiment, the reliability of the biometric device (authentication site) was varied from 0.7 to 0.99 in steps of 0.05. The failure probability and the 95 % confidence level of the system are also calculated.

Results for Experiment I

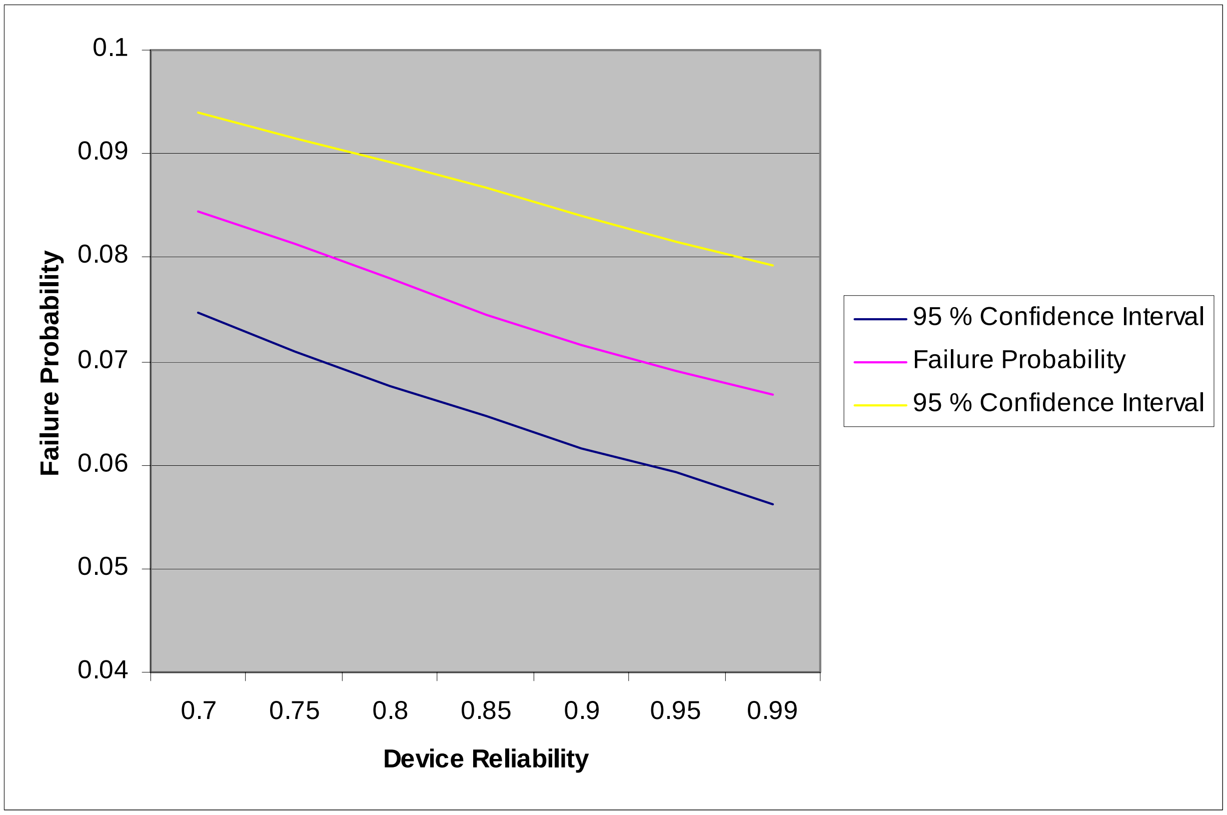


Figure 26: Results for Experiment I - Model II

The failure probability drops down steadily as the reliability of the device increases. The line in the center shows the failure probability and the other two lines denote the 95 % confidence levels.

EXPERIMENT II

The second experiment, four authentication sites were considered. In this case the four device's reliability was varied from 0.7 to 0.99 in steps of 0.05.

Results for Experiment II

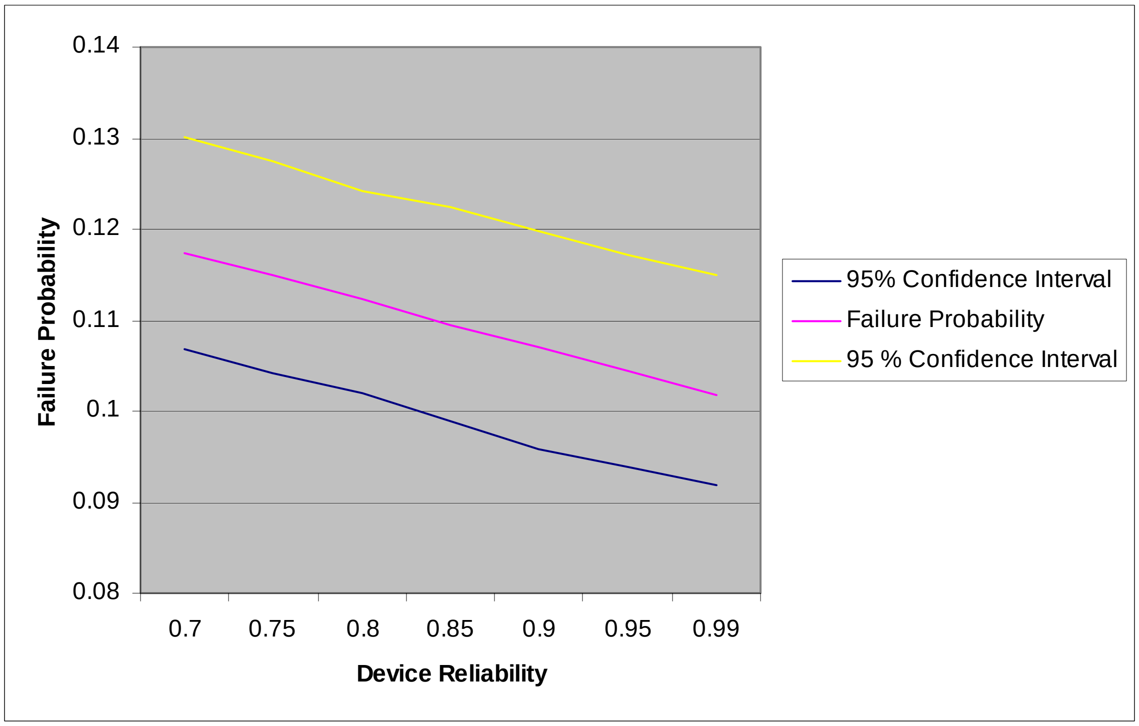


Figure 27: Results for Experiment II - Model II

The failure probability of the four devices decreases as the reliability of the device increases. The failure probability decreases from 0.11 to 0.09, when the device reliability is increased from 0.7 to 0.99 for the four devices.

EXPERIMENT III

For the third experiment, four authentication sites were considered. Of the four the device reliability for three sites were fixed at 0.8, 0.85 and 0.95. The reliability of the fourth site was varied from 0.7 to 0.99 in steps of 0.05

Results for Experiment III

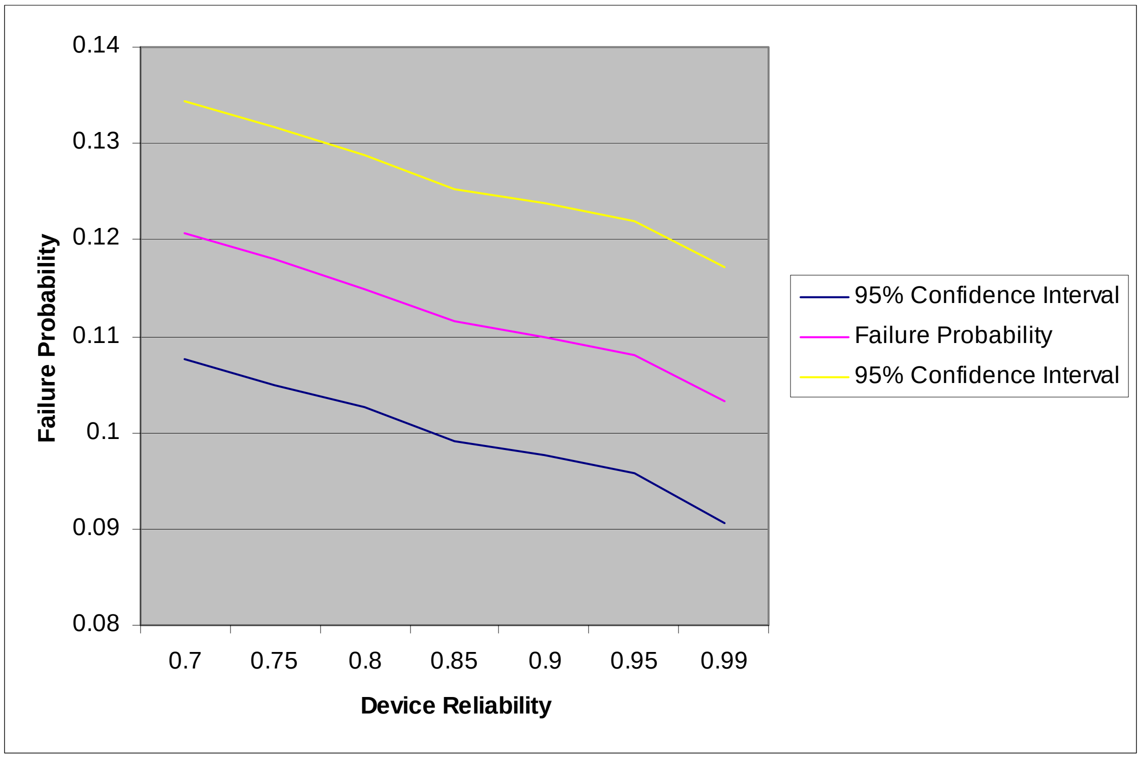


Figure 28: Results for Experiment III - Model II

The above plots have device reliability on X -axes and failure probability on the Y-axes. It can be seen that as the device reliability increases the failure probability decreases. Based on the component based reliability theory, it is possible to model every component in a software model. By performing these experiments, it was learnt that early component based reliability assessment could be applied to any software model.

4. CONCLUSIONS AND FUTURE WORK

Current and future applications of biometrics would benefit from undergoing reliability modeling. We believe that a thorough study of the performance of biometric systems is a must before deploying the system. The requirement for such a study needs a data sheet that gives failure probability of components and connectors. It is our hope that through the development of the ECRA tool and model, practitioners will be able to apply software reliability assessment techniques to biometric systems at earlier stages of the life cycle model than currently available. U*sing the UML diagrams and the ECRA tool, the reliability of a large system containing biometric devices can be predicted and analyzed.*

The proposed approach can be applied in a very early phase of system design when the use-cases and the sequence diagrams are available. Reliability estimation prior to system integration can be very useful in investigating the quality related consequences of configuring a system. When repeated during testing the success of the prediction can be seen. As standard software reliability engineering practices dictate, UML annotations are used for defining operational profiles.

Future work could be looking at eliminating the assumptions made in the reliability estimation algorithm. It would be an interesting study to see what the impacts are to the system by removing the independence and regularity assumptions in the model. Building practical tests cases will demonstrate the success of the ECRA approach. It would also help testing and refining the approach that has been adopted. It is our hope that the research that has been completed will provide a methodology for studying the reliability impact of biometric devices to large systems.

Bibliography

1. B.Cukic et al., "*Early Reliability Assessment of UML Based Software Models*", Proc. International Workshop on Software Performance, Rome, Italy, July 2002.
2. B.Cukic et al., "*A Bayesian Approach to Reliability Prediction and Assessment of Component Based Systems*", Proc. 12th International Symposium on Software Reliability Engineering (ISSRE'01), Hong Kong, November 2001.
3. William .B. Smith, "*Early Component-Based Reliability Assessment using UML based Software Models* ", MS Thesis, LDCSEE, WVU, May2002
4. Lyu, M., "*Handbook of Software Reliability Engineering*", McGraw-Hill, New York, NY, 1996
5. Engineering Statistics Handbook, http://www.itl.nist.gov/div898/handbook/apr/section1/apr167.htm
6. Joe Lambert, "*Engineered Software*", http://www.cse.psu.edu/~lambert/420/big/node7.html. 05/06/1999
7. Gokhale et. al., Important Milestones in Software Reliability Modeling, *Proc. Software Engineering and Knowledge Engineering (SEKE) '96*, Lake Tahoe, NV.
8. Ganesh J. Pai, "*A Survey of Software Reliability Models*", Project Report, http://www.cs.virginia.edu/~jck/cs651/project.reports/SRMsurvey.pdf, Fall 2002.

9. J.L. Wayman, "*Technical Testing and Evaluation of Biometric Identification Devices*" in A. Jain, et al (eds), Biometrics: Personal Identification in a Networked Society, (Boston, Kluwer Academic Press, 1999)

10. I. Foster, C. Kesselman, S. Tuecke. ," *The Anatomy of the Grid: Enabling Scalable Virtual Organizations.* ", InternationalJ. Supercomputer Applications, 15(3), 2001.

11. R. Butler, D. Engert, I. Foster, C. Kesselman, S. Tuecke, J. Volmer, V. Welch." *A National-Scale Authentication Infrastructure*" IEEEComputer, 3(12): 60-66, 2000.

12. A. Ross and A. K. Jain, "*Information Fusion in Biometrics*", Pattern Recognition Letters, Vol. 24, Issue 13, pp. 2115-2125, Sept 2003.

13. Brown, A., Wallnau, K., "*Engineering of Component-Based Systems,*" Component-Based Software Engineering: Selected Papers from the Software Engineering Institute, IEEE Computer Society, 1996, pp. 7-15

14. Goseva-Popstojanova, K., Trivedi, K.S., "*Architecture Based Approaches to Software Reliability Prediction*", International Journal Computers & Mathematics with Applications

15. J. M. Voas, "*COTS and High Assurance: An Oxymoron*?" Proc. Of the 4th IEEE International Symposium on High-Assurance System Engineering, Bethesda, MD, 1998.

16. V. Welch, F. Siebenlist, I. Foster, J. Bresnahan, K. Cajkowski, J. Gawor, C. Kesselman, S. Meder, L. Pearlman, S. Tuecke." *Security for Grid*

Services” Twelfth International Symposium on High Performance Distributed Computing (HPDC-12), IEEE Press, to appear June 2003.

17. J. Novotny, S. Tuecke, V. Welch. “*An Online Credential Repository for the Grid: MyProxy*”Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), IEEE Press, August 2001.
18. I. Foster, C. Kesselman, G. Tsudik, S. Tuecke. “*A Security Architecture for ComputationalGrids*”Proc.5th ACM Conference on Computer and Communications Security Conference, pp.83-92, 1998.
19. Carl Ellison and Bruce Schneier, “*Ten Risks of PKI: What You’re not Being Told about Public Key Infrastructure*”, http://www.counterpane.com/pki-risks.pdf.
20. Roger S. Pressman,” *Software Engineering*”, McGraw-Hill, New York, NY, 2001

Curriculum Vitae

Karthikeyan Mahadevan

1866 B University Ave. (304)-685-9046

Morgantown, WV-26505 mahadeva@csee.wvu.edu

___

Summary

- Graduate Research Assistant, West Virginia University
- Graduate Teaching Assistant, West Virginia University
- Associate Engineer E-Security and Networking Consultant, Odyssey Technologies, Nov '00- Jun '01

Areas of Expertise:

Operating Systems: Windows 95/98/NT/2000, DOS, Unix (& Flavors- Red hat Linux, Solaris)

Languages: C, C++, Perl, HTML

Packages: VC++, MS-Office, Rational Rose RT, MATLAB.

Work Experience: Associate Engineer E-Security and Networking Consultant, Nov '00 – Jun '01

- Secure Shell, a replacement for the Telnet. Team Lead and was responsible for design and implementation the project. The majority of the code was in C and C++ and the GUI was in Visual C++ (Dec '00 – Feb '01)

- Single Sign-On applications using Smart Cards were a series of projects developed for various Multinational Banks wherein the users can login from any part of the world to access their account and transact with security. The application was developed entirely in C (Jan '00 – Apr '01)

- Deployment of SOCKS Server and Security Mechanisms for Citibank. (Mar '00 – Jun '01)

- Developed and Designed PKCS#12 library
- Awarded the Best RAD (Rapid Application Development) for Secure Shell

Research: Reliability Assessment of Biometric Devices in Large Scale Applications

- Integration of CORBA API and BIO API
- NASA IPG project and developing a secure Grid using Biometrics
- Biometrics for Privacy and Authentication issues in Large Scale Distributed Systems
- Developing an algorithm for secure data transfer using Biometrics. Most of the coding is in C.
- Also currently implementing a Self-Certifying File System on DAFS Server for Linux and Linux like file systems. This is done in C and C++.
- Iris Recognition: We are looking at developing using Wavelets and security with XML

Education:

Masters in Computer Engineering, West Virginia University 08/01 – 08/03

Bachelors in Electronics and Communication Engineering, University of Madras, India.08/’96 – 07/’00 76.2% (grades not-convertible)

Diploma in Network Centered Computing – National Institute of Information technology, Chennai India 04/’97 – 07/’99

Academic Honors:

Ranked Ist in the University of Madras for the subject of Computer Programming.

First Prize in National Code Debugging Contest.

References available upon request