FLORIDA DEPARTMENT OF FINANCIAL SERVICES

PUBLIC HURRICANE RISK AND LOSS MODEL

PHRLM

PRIMARY DOCUMENT BINDER

The Document

This binder contains a complete set of documents specifying the model structure, detailed software description, and functionality.

Project Supervisors

Dr. Shu-Ching Chen
Associate Professor,
School of Computer Science
Florida International University

Dr. Mei-Ling Shyu
Associate Professor,
Electrical and Computer Engineering
University of Miami

Development Team Members

Min Chen
(Team Leader)
Ph. D. Candidate,
School of Computer Science
Florida International University

Kasun Wickramaratna
MS. Candidate,
School of Computer Science
Florida International University

Na Zhao
Ph. D. Candidate,
School of Computer Science
Florida International University

Xiaosi Zhou
Ph. D. Candidate,
School of Computer Science
Florida International University

Testing Team Members

Indika Priyantha
MS. Candidate,
Electrical and Computer Engineering
University of Miami

Shaminda Subasingha
MS. Candidate,
Electrical and Computer Engineering
University of Miami

Khalid Saleem
Ph. D. Candidate,
School of Computer Science
Florida International University

Kasturi Chatterjee
Ph. D. Candidate,
School of Computer Science
Florida International University

Table of Contents

Section 1

The Public Hurricane Risk and Loss Model (PHRLM)

1.1. General Description of PHRLM Model

The PHRLM model is a probabilistic model designed to estimate the damage and insured losses due to the occurrence of hurricanes in Atlantic Basin. The PHRLM estimates the full probabilistic distribution of damage and loss for any significant storm event. The modeling methodology of it can be partitioned into four major components:

- Storm Forecast Module
- Wind field Module
- Damage Estimation Module
- Loss Estimation Module

The high-level flow chart is shown in Figure 1.1

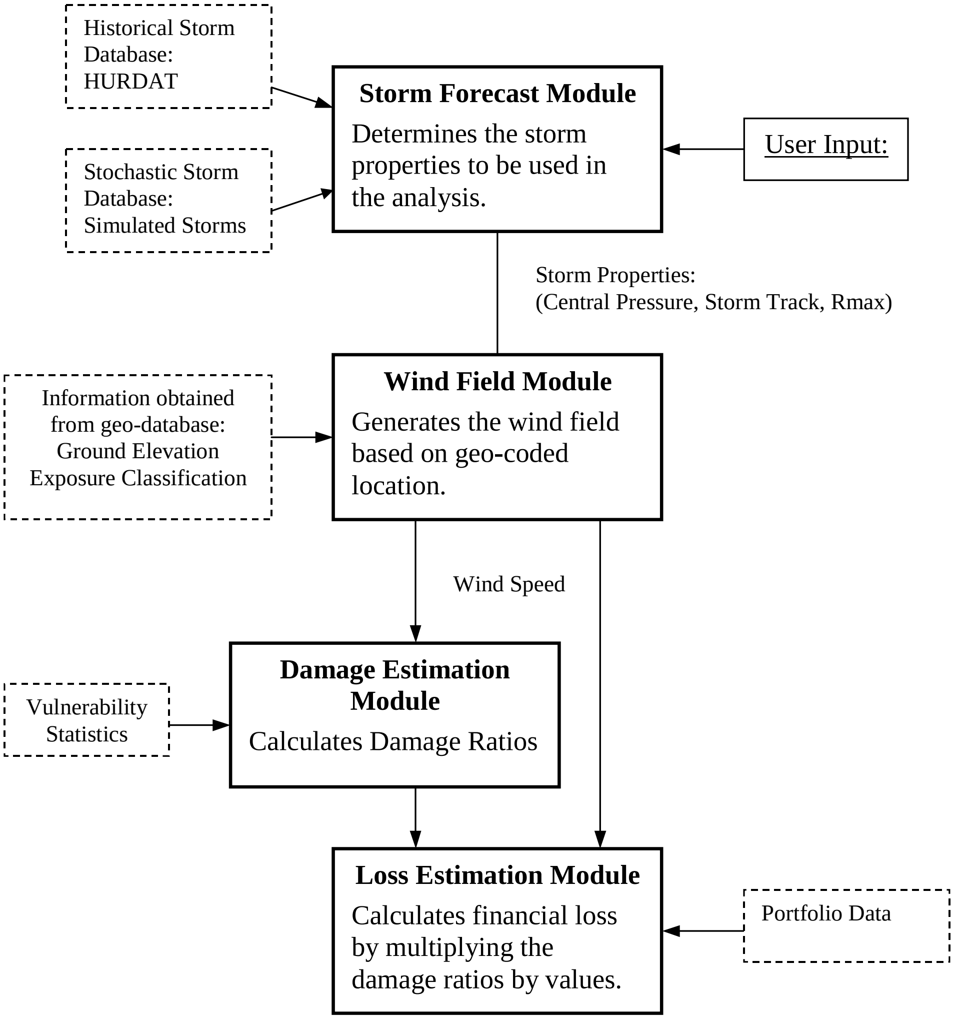

Historical Storm Database: HURDAT
Stochastic Storm Database: Simulated Storms
Storm Forecast Module
Determines the storm properties to be used in the analysis.
User Input:
Storm Properties: (Central Pressure, Storm Track, Rmax)
Information obtained from geo-database: Ground Elevation Exposure Classification
Wind Field Module
Generates the wind field based on geo-coded location.
Wind Speed
Vulnerability Statistics
Damage Estimation Module
Calculates Damage Ratios
Loss Estimation Module
Calculates financial loss by multiplying the damage ratios by values.
Portfolio Data

Figure 1.1: Model Flowchart

1.2. Computer Model and Implementation

1.2.1. Use Case View of the System

Use case diagram is one diagram in UML for modeling the dynamic aspects of a system. Use case diagrams are central to modeling the behavior of a system, a subsystem, or a class. Figure 1.2 presents the use case diagram of our computer model for the PHRLM, and it shows a set of use cases and actors and their relationships.

A. Actors:

There are two actors in this system, the scientists and the statisticians. The scientists can access all use cases related to the Storm Forecast Module and Wind field Module, while the statisticians can interact with the Damage and Loss Estimation Modules.

B. Use Cases:

- **Use Case I: Annual Hurricane Occurrence**
  Use Case I is used to estimate the probability distribution for annual hurricane occurrence and to generate a series of simulated years along with their associated number of storms according to the selected the probability distribution.

- **Use Case II: Storm Genesis Time**
  Use Case II is used to generate the probability distribution of the origin dates for the historical storms and simulated storms (produced by Use Case I).

- **Use Case III: Storm Track Generation**
  Use Case III is used to generate the storm tracks for simulated storms based on data obtained from Use Case II and stochastic algorithms.

- **Use Case IV: Wind Field Generation**
  Use Case IV is used to generate wind fields for storms based on the data generated in Use Case IV for the year range specified by the user.

- **Use Case V: Wind Speed Correction**
  Use Case V is used to refine open terrain wind speed produced by the hurricane wind model with respect to the actual terrain (based on land use – land cover).

- **Use Case VI: Wind Speed Probability**
  Use Case VI is used to calculate the probabilities of the 3s gust wind speeds affecting each of the zip codes.

- **Use Case VII: Insurance Loss Module**
  Use Case VII is used to calculate the expected loss values.

C. Use Case Diagram:

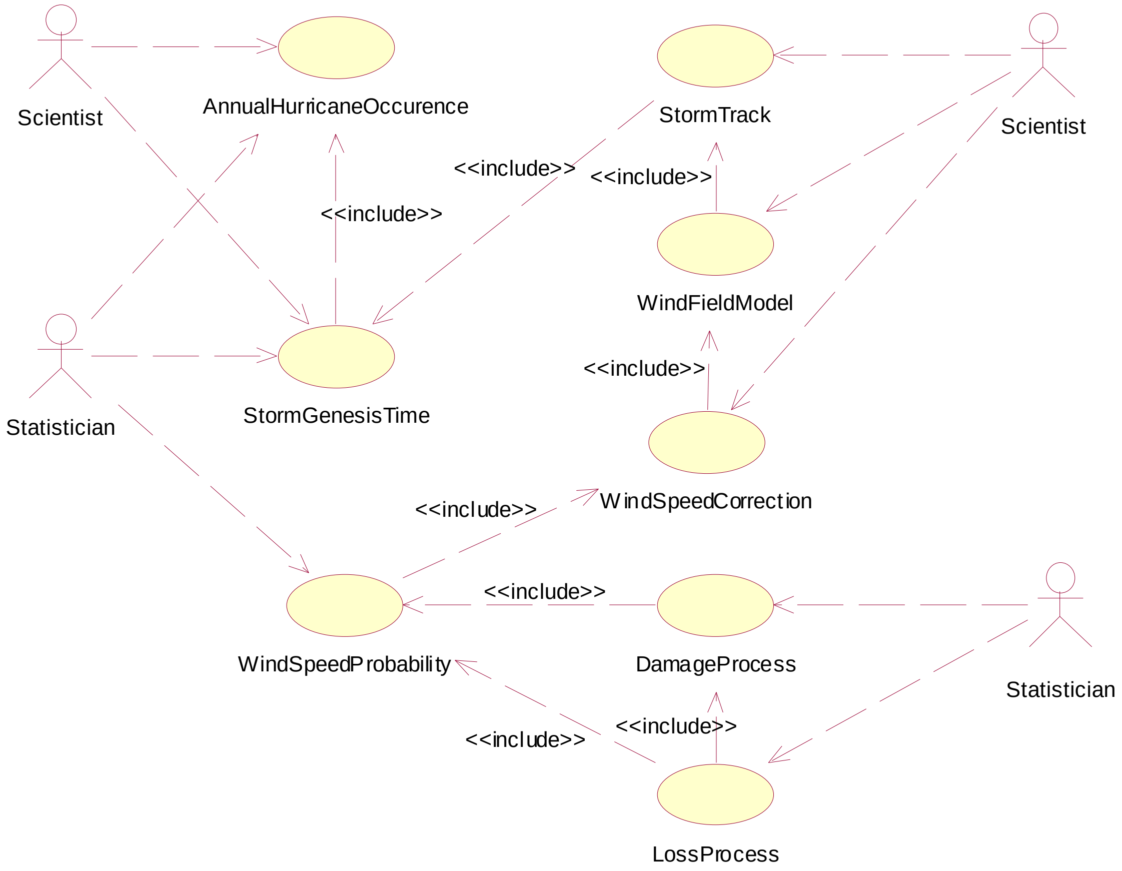


Figure 1.2: Use Case Diagram of the System

1.3. System Architecture Design

Figure 1.3 gives a high-level system architecture abstraction which follows the popular three-layer architecture.

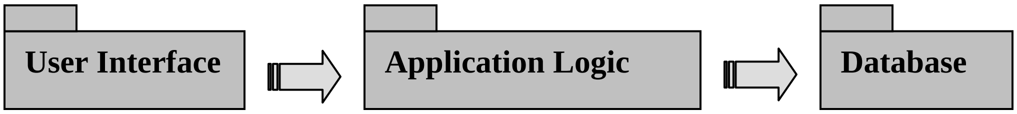


Figure 1.3: The Three-layer System Architecture

A. User Interface:

User Interface is the first layer of the system, and also the only layer visible to the user. Due to the popularity and convenience of the Internet, a web interface is preferred so that the users are able to access the system online.

B. Application Logic:

The second layer is used to glue the user interface and the underlying database. OC4J is chosen to serve as the second layer.

C. Database:

The database layer adopts Oracle*9i* database due to its advanced features for extensibility, availability, high performance and management.

1.3.1. Detailed System Architecture Design

Figure 1.4 is the general system organization. There are five major components: client, OC4J container, Java application, Oracle database, and math model.

♦ **Client Side**

The users can gain access to the system through any commonly used commercial browser such as Internet Explorer, Netscape and etc. The user interface should be friendly and can offer the user-required functionalities as best as possible. JSP (Java Server Page) technique is used to dynamically generate the content in the web page. The basic idea of JSP is to allow Java Code to be mixed together with static HTML or XML templates. The Java logic handles the dynamic content generating while the markup language controls structuring and presentation of data.

♦ OC4J

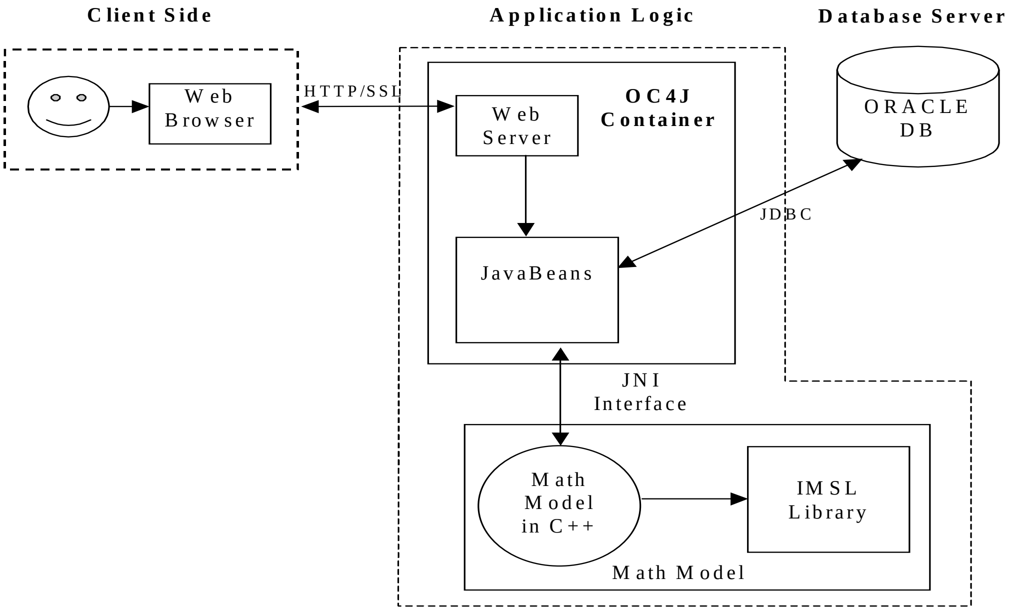


Figure 1.4. Detailed system architecture

OC4J is short for *Oracle9iAS Containers for J2EE*. It is a complete J2EE 1.2 container that includes a JSP Translator, a Java servlet engine, and an Enterprise JavaBeans (EJB) container. OC4J also supports the Java Messaging Service and several other Java specifications.

Advanced techniques such as JavaBeans and JNI are employed in the second layer. JavaBean is a Java class that defines properties and that communicates with other Beans via events. Properties can be defined within the JavaBean class definition, or they can be inherited from other classes. JNI stands for Java Native Interface; it is part of the Java Developer Kit. The actual mathematical and statistical computations are implemented in C/C++ language for the sake of speed; JNI then serves as a bridge between java side and native side of an application.

♦ JDBC

JDBC is a Java program that provides a way for the user to invoke SQL statements to access the database. JDBC API is used to build the communication between the Java program and the database server. Multiple database drivers for connecting to different databases are supported by JDBC.

Actually, JDBC technology allows users to access virtually any tabular data source from the Java programming language. It provides cross-DBMS connectivity to a wide range of SQL databases.

Through JDBC API, developers can take advantage of the Java platform's “Write Once, Run Anywhere” capabilities for industrial strength, cross-platform applications that require access to enterprise data. With a JDBC technology-enabled driver, a developer can easily connect all corporate data even in a heterogeneous environment.

♦ JNI

Java is one of the most popular languages with strong support for web application, however the math model is implemented using C++ for the sake of speed and the stronger functionalities supported in the IMSL library C++ version. To bridge the gap between the Java application and the math model, the JNI is employed.

JNI stands for Java Native Interface. JNI is a standard programming interface for writing Java native methods and embedding the Java virtual machine into native applications. The primary goal is binary compatibility of native method libraries across all Java virtual machine implementations on a given platform. Native programs writing in languages other than java, such as C/C++ can be integrated into Java applications and it is ensure these programs are completely portable across all platforms. By programming through the JNI, you can use native methods to create, inspect, and update Java objects (including arrays and strings), to all Java methods, to perform runtime type checking.

Section 2

Storm Forecast Module
Module I

Section 2.1

Annual Hurricane Occurrence (AHO)
Use Case I

2.1.1. General Description Of AHO

AHO, short for Annual Hurricane Occurrence, is the first use case in the FIU/IHRC Public Hurricane Risk and Loss model. It aims at estimating the probability distribution for annual hurricane occurrence and generating a series of simulated years along with their associated number of hurricanes according to the probability distribution that has the best goodness of fit.

Threat Area

In our latest PHRLM model, only the hurricanes fall in the threat area are considered. Here, the threat area is defined as a radius of 900 km centered at 29.0 North (Latitude) and 83.0 East (Longitude), which is actually the region of the interest. The threat area surrounding Florida is shown in s1. In other words, a hurricane will be considered if it ever passed the threat area and it ever had the wind speed of larger than 74mph (at least Category 1 hurricane) when it was in the threat area. The wind speed ranges for category 1-5 hurricanes is shown in Table 2.1.1. For instance, hurricane Andrew 1992 [12] as shown in Figure 2.1.1 is one of the qualified hurricanes, which are considered in our model.

Table 2.1.1: The wind speed ranges for category 1-5 hurricanes

Category	Wind Speed (mph)
1	74 <= WD < 95
2	95 <= WD < 110
3	110 <= WD < 130
4	130 <= WD < 155
5	155 <= WD

In our database (1851-2003), there are totally 1274 historical tropical cyclones, which include all hurricanes and tropical storms that were not hurricanes. After the filtering process using the threat area definition, only 309 of them are considered as the valid historical hurricane records.

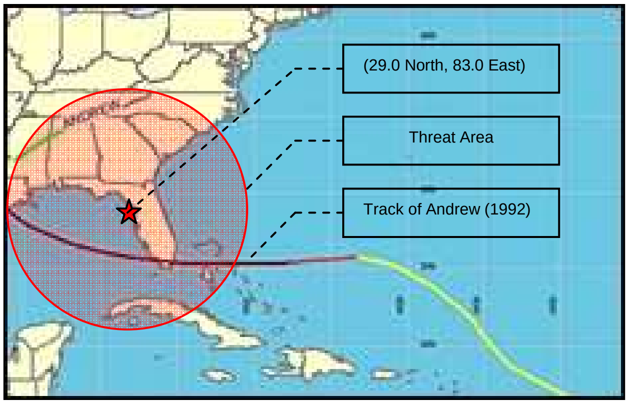


Figure 2.1.1. Threat area and the storm track of hurricane Andrew 1992

2.1.2. AHO Design Requirements

Name: Annual Hurricane Occurrence (Threat Area Only)

Description: The user enters a choice of year range and the system generates the following:

(1) A probability distribution for the number of hurricanes per year.

(2) A simulated number of years with their associated numbers of hurricane occurrences.

1. The user enters a year range from the following selections:

1851-2003
1900-2003
1944-2003
Multi-Decadal
ENSO

NOTE:

Neutral Years: All non-ElNino and non-LaNina years are considered Neutral or average.

Multi-Decadal: Warm (and active), Cold (quiet). See Table 2.1.2 for a listing of Multi-Decadal year ranges.

ENSO: EL Nino, La Nina; see Table 2.1.3 for a listing of El Nino and La Nina years.

Table 2.1.2: Listing of Multi-Decadal year ranges and temperature

Temperature (Warm)	Temperature (Cold)
1870-1902	1903-1925
1926-1970	1971-1994
1995-2003	

Table 2.1.3: Listing of El Nino and La Nina years

El Nino - Year	La Nina - Year
1925	1933
1929	1938
1930	1942
1940	1944
1941	1945
1951	1948
1953	1949
1957	1950
1963	1954

1965	**1955**
1969	**1956**
1972	**1961**
1976	**1964**
1977	**1967**
1982	**1970**
1986	**1971**
1987	**1973**
1990	**1974**
1991	**1975**
1993	**1978**
1994	**1988**
1997	**1995**
	1998
	1999
	2000
	2002

2. Based on the selection of year range inputted by the user from Step 1, the system queries the database and returns data of the years within the desired year range and their associated numbers of tropical cyclones occurrences. Table 2.1.4 illustrates the content of the returned data.

Table 2.1.4: Matrix of Number of Hurricanes Per Year

Year (Y_0 - Y_n)	**# Hurricanes (H_0 - H_n)**
1900	**1**
1901	**3**
Y_i	**H_i**
Y_n	**H_n**

NOTE:

In the initial development, the model considered all the tropical cyclones which include all hurricanes and tropical storms that were not hurricanes. In the latest version of model, only the hurricanes in the threat area are considered.

3. The system uses the data retrieved in Step 2: the years and their associated numbers of hurricanes, and the Statistician's equation to generate the parameters of probability distribution.

Detailed steps are as follows:

3.1. The system fits the distribution for the historical data from Table 2.1.4. To do so, system uses historical data from Table 2.1.4 to calculate the **mean** and the **standard deviation**.

3.2. The system stores the output (mean & standard deviation) from 3.1 in the database.

3.3. The system determines the distribution fits for each range using the Poisson model:

3.3.1. For Poisson model, the system calculates the Mean "u".

3.3.2. The system determines the goodness of fit for the Poisson model.

3.3.2.1. The system calculates “k”, the maximum number of hurricanes in the data set.

3.3.2.2. The system calculates the number of hurricanes (X_0 – X_k) and the observed frequencies (O_0 – O_k), where “O_i” represents the number of years in which there were “i” hurricane occurring.

3.3.2.3. The system calculates “n”, the sum of the observed frequencies generated in 3.3.2.2.

3.3.2.4. The system uses the Mean “u”, and IMSL library functions to calculate the expected frequencies for the number of hurricanes (E_0 – E_k) where “E_j” represents the expected number of years in which ‘j’ hurricanes will occur.

3.3.2.5. The system generates a frequency table for the number of hurricanes. The frequency table is a matrix consisting of 3 columns: The number of hurricanes (X_0 – X_k), the observed frequencies (O_0 – O_k), and the expected frequencies (E_0 – E_k). See Table 2.1.5.

3.3.2.6. The system stores the frequency tables (Table 2.1.5) generated back to the database.

3.3.2.7. The system reconstruct the frequency table to make sure that no expected value is less than 1 and no more than 20% are less than 5. If either of the two conditions is violated, then some categories are combined so that the conditions are always satisfied.

3.3.2.8. The system calculates chi-squared statistics, i.e., the goodness of fit statistics.

3.3.2.9. The system calculates the p-value by calling the IMSL CHIDF routine.

3.3.2.10. The system stores the resulted chi-squared statistic and p value (Table 2.1.7) back to the database.

3.4. The system determines the distribution fits for each range using the Negative Binomial model.

3.4.1. For Negative Binomial model, the system calculates "m" and "k" estimates.

3.4.2. The system determines the goodness of fit for the Negative Binomial modal:

3.4.2.1. Repeat Steps 3.3.2.1. Through 3.3.2.3.

3.4.2.2. The system uses $X_0 – X_k$, $O_0 – O_k$, “k”, and “n” to calculate the expected frequencies for the number of hurricanes (E_0 – E_k) where “E_j” represents the expected number of years in which ‘j’ hurricanes will occur. The system calls the IMSL gamma function to calculate the expected values.

3.4.2.3. The system repeats Steps 3.3.2.5. — 3.3.2.6. for the Negative Binomial model.

3.4.2.4. The system generates a frequency table for the number of hurricanes. The frequency table is a matrix consisting of 3 columns: The number of hurricanes (X_0 – X_k), the observed frequencies (O_0 – O_k), and the expected frequencies (E_0 – E_k). See Table 2.1.6.

3.4.2.5. The system stores the generated frequency tables (Table 2.1.6) back to the database.

3.4.2.6. The system stores the chi-squared statistics (Table 2.1.7) back to the database.

3.5. The system selects the distribution that gives the highest p value to be the final selected distribution for the number of hurricanes per year.

3.6. The system plots the observed frequencies versus the fitted frequencies in a histogram.

Table 2.1.5: Frequency table of number of hurricanes, yearly frequencies, and expected frequencies for the Poisson Model

# Hurricanes ($X_0 – X_k$)	Observed Frequency ($O_0 – O_k$)	Expected Frequency ($E_0 – E_k$)
0	O_0	E_0
1	O_1	E_1
2	O_2	E_2
X_k	O_k	E_k

Table 2.1.6: Frequency table of number of hurricanes, yearly frequencies, and expected frequencies for the Negative Binomial Model

# Hurricanes ($X_0 – X_k$)	Observed Frequency ($O_0 – O_k$)	Expected Frequency ($E_0 – E_k$)
0	O_0	E_0
1	O_1	E_1
2	O_2	E_2
X_k	O_k	E_k

Output from Step 3:

Table 2.1.7: Probability distribution

Year Range: 1851-2003

Type of fit	Mean	Variance	p-value	Goodness of fit
Poisson				
Negative Bin.				

4. The system presents the following question to the user:

4.1 How many years would you like for your simulation?

4.1.1. The user input the number of years for simulation. For example, 100,000 years.

5. The system uses **IMSL** routines to generate a random sample from the chosen distribution obtained in Step 3 and generates a number of simulated years (**SY_0 - SY_n**) and their associated numbers of hurricanes (**SH_0 - SH_n**).

Output from Step 5:

Table 2.1.8: Simulated years and their associated numbers of hurricanes

Year (SY_0 - SY_n)	**# Hurricanes (SH_0 - SH_n)**
SY_0	**SH_0**
SY_1	**SH_1**
SY_i	**SH_i**
SY_n	**SH_n**

6. The system stores simulated years and number of hurricanes (see Table 2.1.8).

<u>Note:</u>

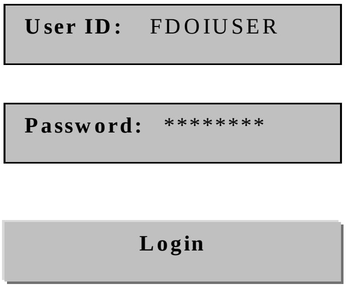
Steps-2 – 4 are repeated for each year range that the user requests.

2.1.3. AHO Interface Design Requirements

This part designs the GUI (Graphic User Interface) for the Annual Hurricane Occurrence (AHO). The user interface design aims at providing a friendly and easy-to-use environment for the users to log in to the PHRLM system and access the AHO use case.

A. The first step: the user logs in the system

Figure 2.1.2 depicts the Login Interface. The user enters a User ID and a corresponding password, and then submits the login request to the system. The system verifies the user ID and password. If the User ID and Password are correct, the access right is granted to the user; otherwise, the system gives the "invalid user/password" error message and asks the user to reenter the User ID and password.



User ID: FDOIUSER

Password: ********

Figure 2.1.2: Login interface

B. The second step: the user selects a year range

Figure 2.1.3 illustrates the Year Range Selection Interface. The system presents a list of year ranges to the user. The valid year ranges are shown at Table 2.1.9. The user selects a year range and submits to the system.

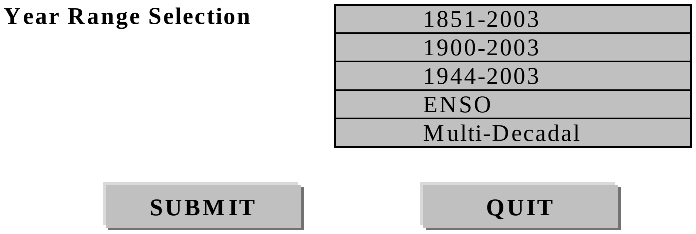


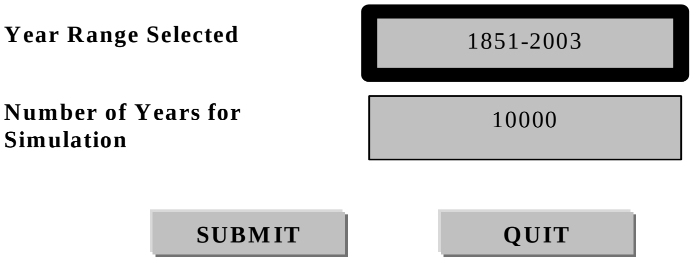
Figure 2.1.3: Year range selection interface

Table 2.1.9: list of the valid year ranges

Valid Year Range
1851-2003
1900-2003
1944-2003
ENSO
Multi-Decadal

C. The third step: the user specifies the number of years for simulation

Figure 2.1.4 shows the Simulation Selection Interface. The system displays the year range selected at step 2. The user then specifies the number of years and submits to the system.



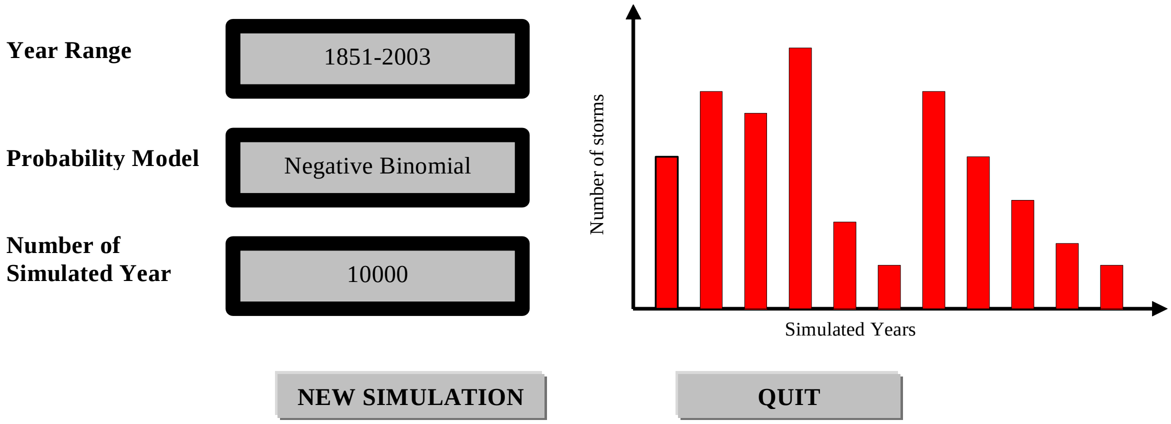
Year Range Selected: 1851-2003

Number of Years for Simulation: 10000

Figure 2.1.4: Simulation selection interface

D. The fourth step: simulation results display

Figure 2.1.5 portrays the Simulation Results Display Interface. The system displays the year range selected at step 2, the probability model used, the number of simulated years user designated at step 3 and the simulation results.



Year Range: 1851-2003

Probability Model: Negative Binomial

Number of Simulated Year: 10000

Figure 2.1.5: Simulation results display interface

2.1.4. Computer Model Design

2.1.4.1. Use Case View of AHO

A. Actors:

There is one actor (scientists) in AHO. Scientists use this use case to find a statistic modal with satisfying goodness of fit, conduct the simulation and observe the simulation results.

B. Use Case:

Use case AnnualHurricaneOccurrence is used to estimate the probability distribution for annual hurricane occurrence and to generate a series of simulated years along with their associated numbers of hurricanes occurrences with respect to the probability distribution that has the best goodness of fit.

C. Use CaseDiagram:

Figure 2.1.6 shows the use case diagram for AHO.

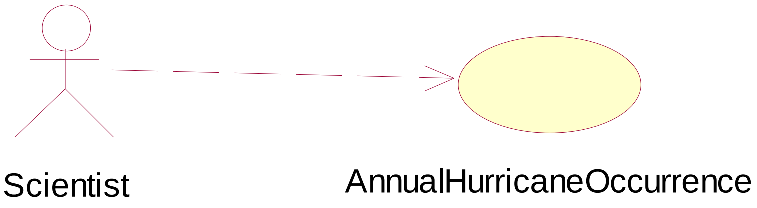


Figure 2.1.6: Use Case Diagram for AHO

2.1.4.2. System Design

This part describes the system design. Appropriate diagrams are provided to describe the system classes, activities and the overall flow chart of AHO.

2.1.4.2.1. Program Flow Chart of AHO

The overall flow chart of AHO is illustrated as follows.

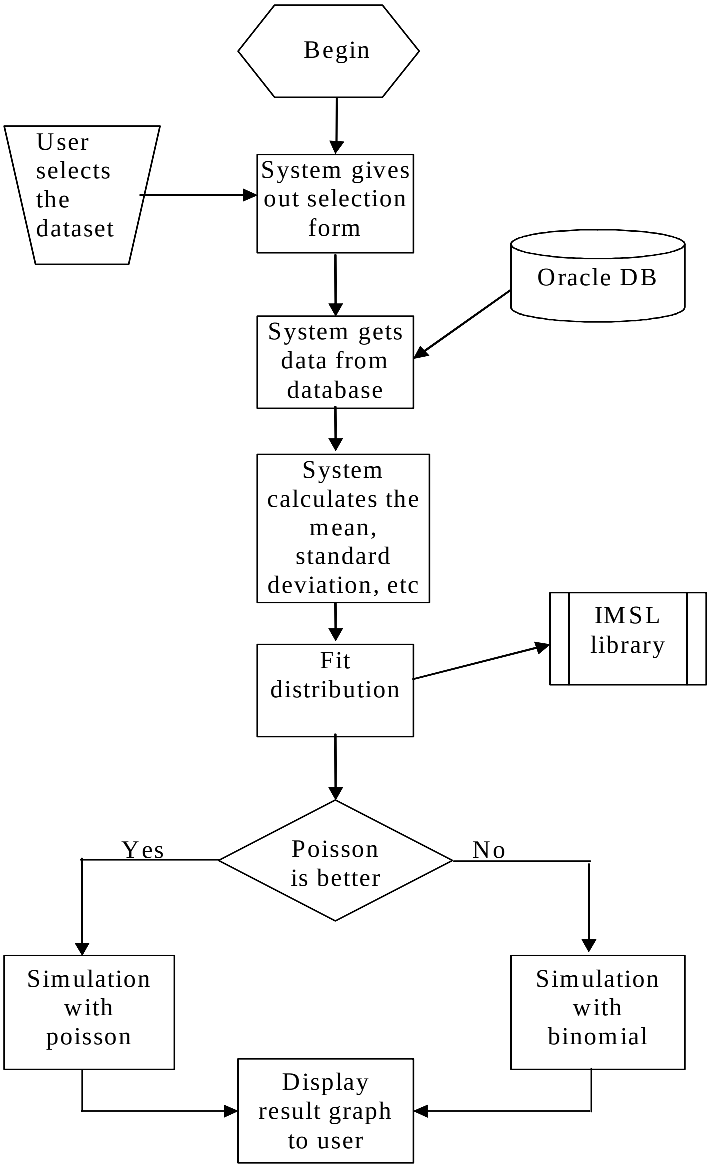

Begin
User selects the dataset
System gives out selection form
Oracle DB
System gets data from database
System calculates the mean, standard deviation, etc
IMSL library
Fit distribution
Poisson is better
Yes
No
Simulation with poisson
Simulation with binomial
Display result graph to user

Figure 2.1.7: Flow chart of AHO

2.1.4.3. Class Diagram and Description

A. Class Diagram

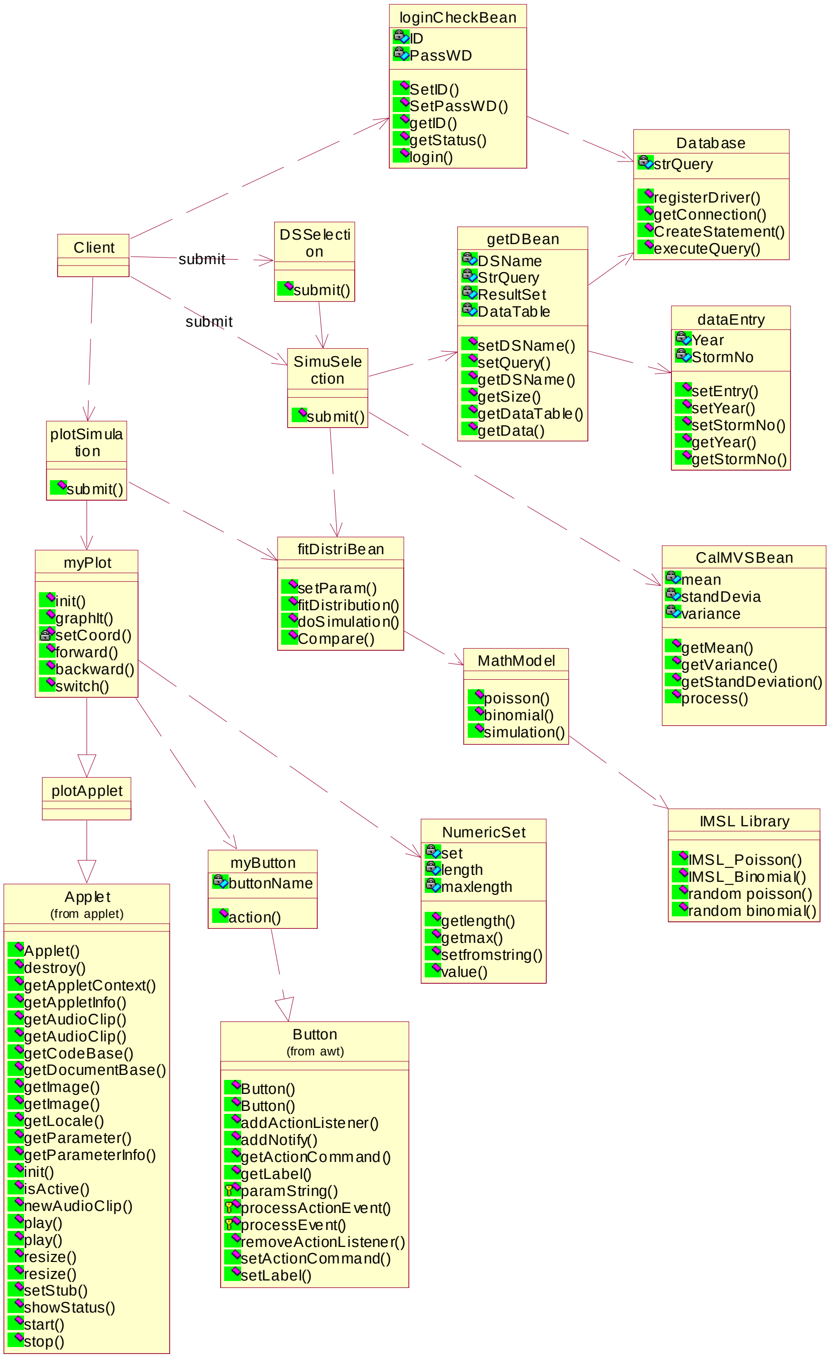


Figure 2.1.8: Class Diagram for AHO

B. Classes Descriptions

This section addresses the major classes used and their functionalities.

- **Client**:
  This class is a virtual class. It refers to the user who uses this system. No need to implement it.

- **LoginCheckBean**:
  This class is for the user login authorization purpose. It gets the username (ID) and the associated password, verifies the information with data stored in Oracle Database.

- **DSSelection/SimuSelection**:
  This class is used to get the user's selection of year range. It then passes control to the classes that can get data from database and do the simulation.

- **GetDBean**:
  This class is used to get hurricane data from Oracle database.

- **DataEntry**:
  This class is used to hold data records.

- **CalMVSBean:**
  This class is used to calculate statistic characteristics of a data set such as mean value and standard deviation.

- **Database**:
  This class is an abstract concept. It includes all the systems that can provide database operations.

- **FitDistriBean**:
  The class is used to interface with the actual math model.

- **MathModel**:
  This C++ class using IMSL library functions to fit distribution and generate the simulation result. It communicates with the Java main application using JNI interface.

- **IMSL Library**:
  This is a statistical and mathematic functions library provided by IMSL.

- **PlotSimulation**:
  This class is used to visualize the simulation result.

- **MyPlot**:
  This class gets the simulation result, and draws the simulation result figure. It also provides buttons to allow the end user to change the graph type, move back/forward in the figure.

- **NumericSet**:
  This class is used to store the simulation result data; it is used by the class **myPlot**.

- **MyButton**:
  This class is for new button customization, which is used by the class **myPlot** to let the end user to change the graph type, move back/forward in the result figure.

- **PlotApplet**:
  This class provides some basic functions in plotting and is the base class for class **myPlot**. (Note: This class is implemented by MIT CS department.)

- **Applet/Button:**
  A base class provided by Java API.

2.1.4.4. Sequence Diagram

Sequence diagrams are helpful in understanding the relations among the classes. This section shows sequence diagrams that describe four major activities in use case AHO, which are **login**, **fit distribution**, **simulation** and **plot** respectively.

A. Login Process

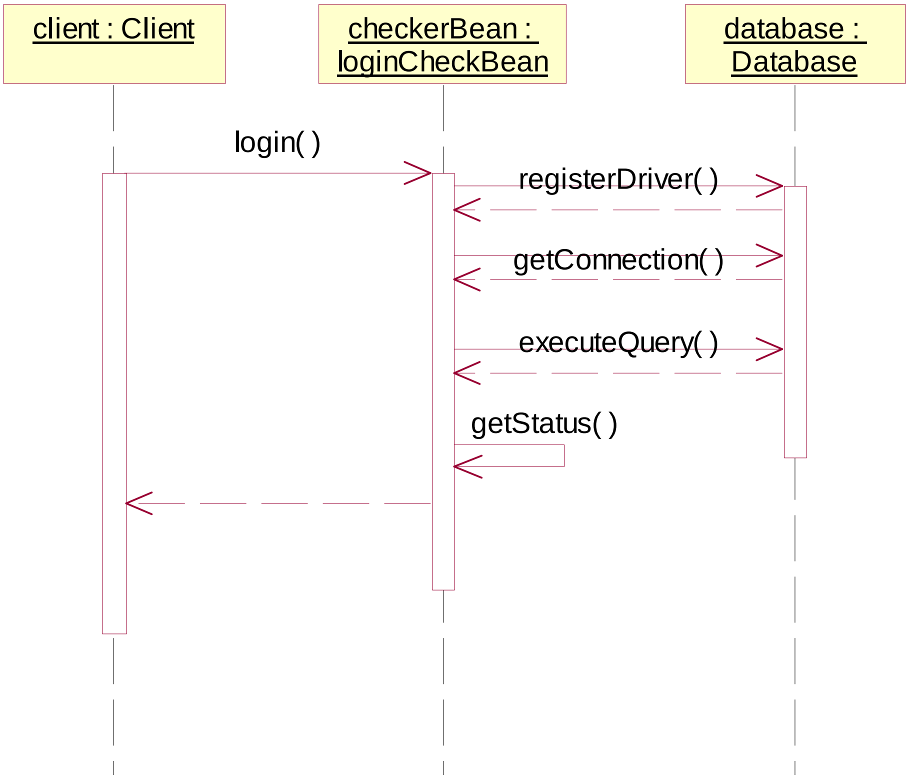


Figure 2.1.9: Sequence diagram for login process

- **Step 1**: The user enters the user ID and password in the web browser and click “login” button

- **Step 2:** The login information is passed to the loginCheckBean class, which communicates with the database. If the password or username is not matched with the information stored in the database, the user gets an error message and is asked to login again. If the username and password are matched, the user can continue to access the system.

B. Simulation Process

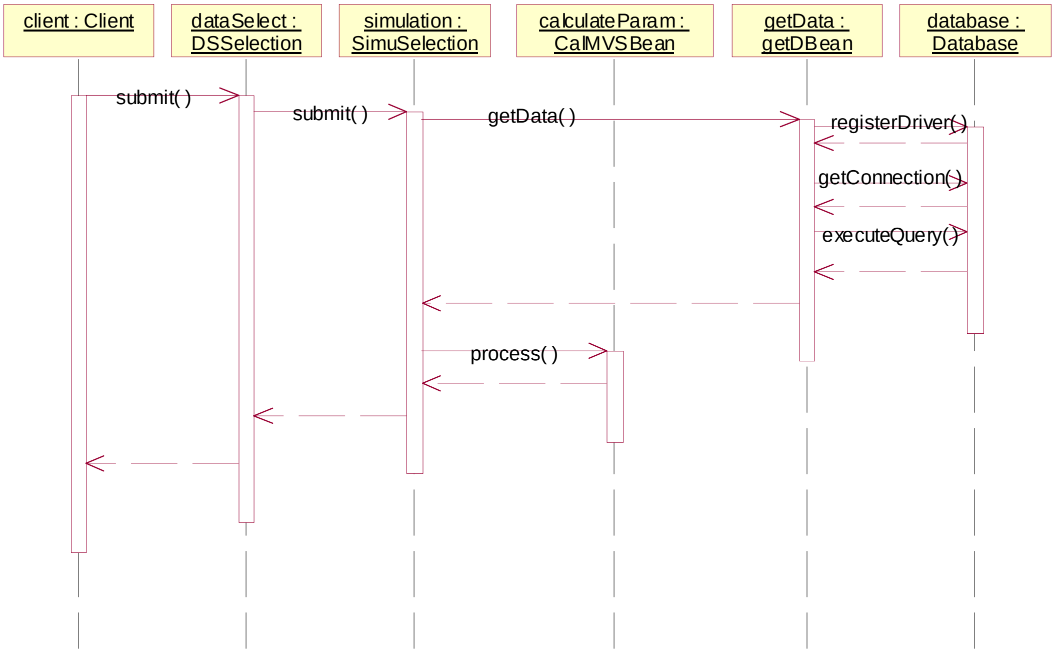


Figure 2.1.10: Sequence diagram for simulation process

- **Step 1:** The user selects a data set (e.g. a year range), and then clicks the "Submit" button.

- **Step 2:** The **dataSelect** object captures the data set selected by the user, and then calls the **simulation** object to get the data from database and processes the retrieved data.

- **Step 3:** The **simulation** object connects with the database, creates the query, and then gets the desired data from database.

- **Step 4:** The **simulation** object calls the **calculateParam** object to calculate some statistic values of the data set such as mean, variance and standard deviation values.

- **Step 5:** The **simulation** object returns the mean, variance and standard deviation values back to the user (displayed in the web browser).

C. Fit Distribution Process

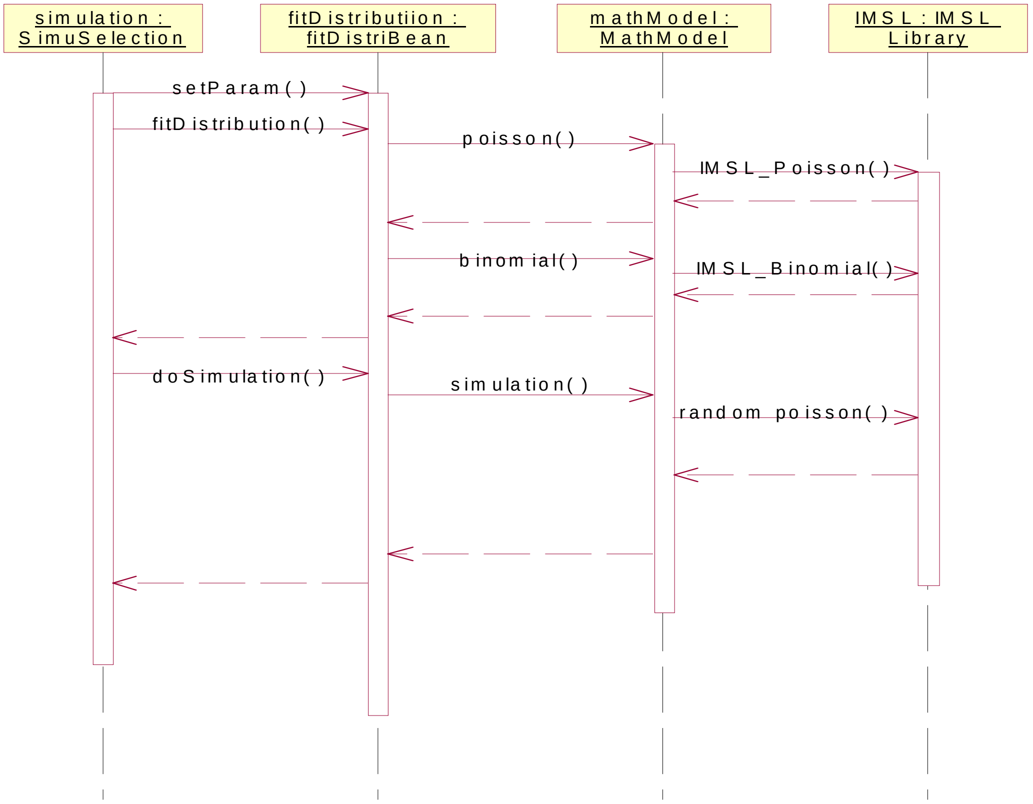


Figure 2.1.11: Sequence diagram for fit distribution process

- **Step1:** The **simulation** object passes the calculated mean, variance, standard deviation values and other parameters to the **fitDistribution** object.

- **Step 2:** The **simulation** object then calls the **fitDistribution** object to fit the distribution using Poisson and negative binomial model and then identifies the better one.

- **Step 3:** The **fitDistribution** object achieves the distribution-fitting task by calling the math models written in C++.

- **Step 4:** The math model calls the IMSL libraries to get the results.

- **Step 5:** After identifying the better model, the **simulation** object calls the **fitDistribution** object to do the actual simulation.

- **Step 6:** The math model does the actual simulation work and return the result set back to the original caller.

- D. Plot Process

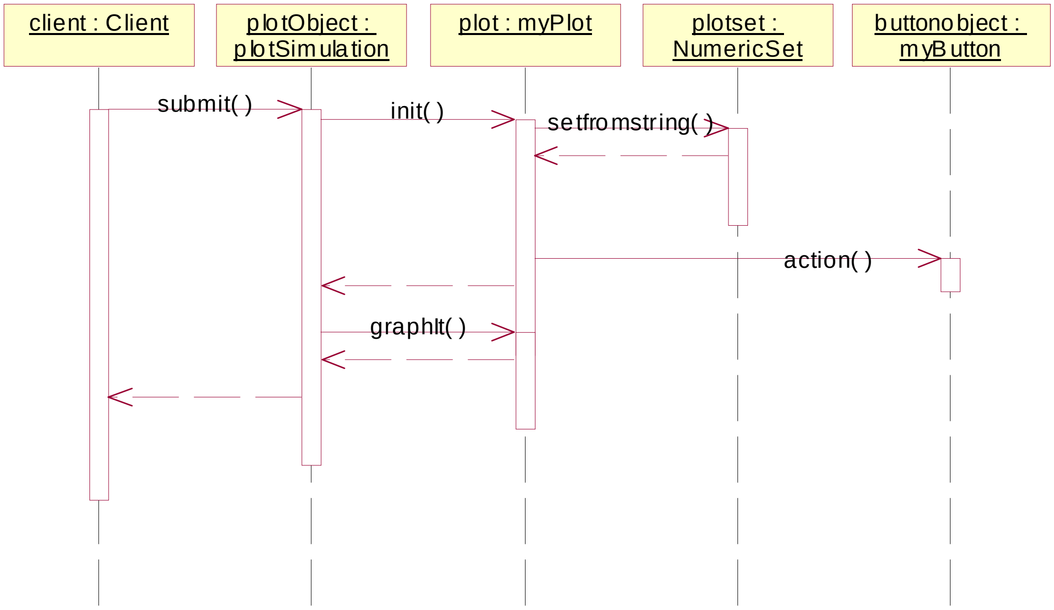


Figure 2.1.12: Sequence diagram for plot process

- **Step 1**: The user submits request for result visualization.

- **Step 2**: The **plotObject** object initializes an instance of **myPlot** class to do the plot task.

- **Step 3**: The initialized **plot** object creates a **plotset** object to store the result obtained from the simulation. Also, it creates several buttons (instance of **myButton** class) to give user the choices to move forward/backward, or change the figure type.

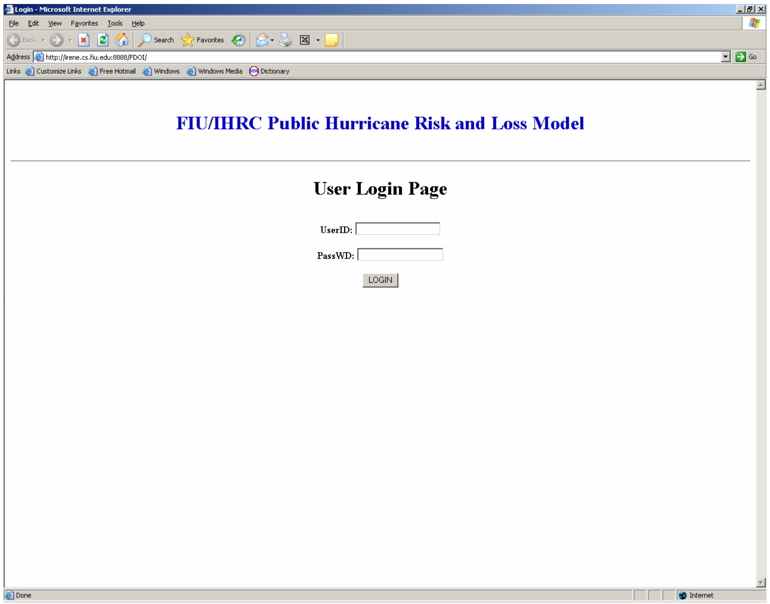
- **Step 4**: The **plotObject** object then calls the **plot** object to plot the resulted simulation data set, and displays it to the user (In user's web browser).

2.1.5. Implementation of AHO

The implementation for use case AHO has already been finished. The demo is online at http://www.cs.fiu.edu/PHRLM.

2.1.5.1. Login page:

The users need a username and a password to access the FIU/IHRC Public Hurricane Risk and Loss Model. Following is the snapshot of the web page for login.



Login - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://irene.cs.fiu.edu:8888/FDOI/ Go
Links Customize Links Free Hotmail Windows Windows Media Dictionary
LOGIN
Done
Internet

FIU/IHRC Public Hurricane Risk and Loss Model

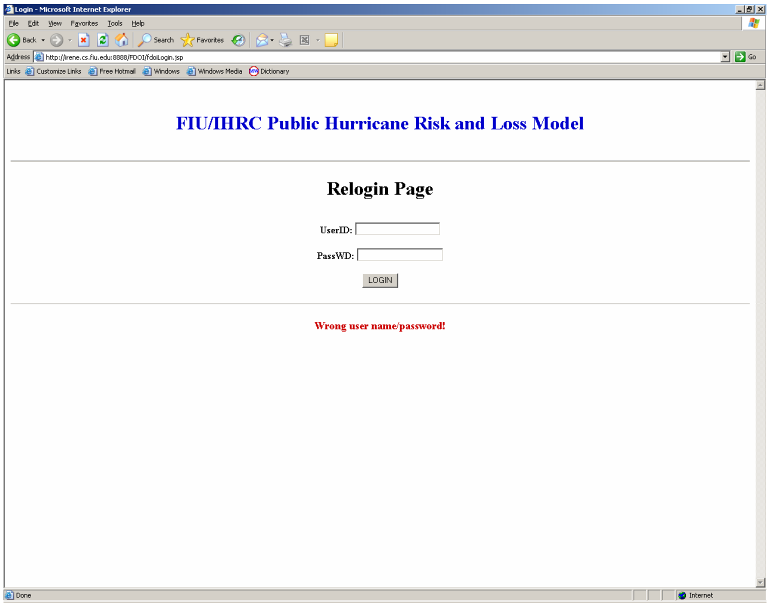
User Login Page

UserID:

PassWD:

Figure 2.1.13. Snapshot of the Login page

If the username/password is wrong, error message is given and the user is required to input the username and password again.



Login - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://irene.cs.fiu.edu:8888/FDOI/fdoiLogin.jsp
Go
Links Customize Links Free Hotmail Windows Windows Media Dictionary
Done
Internet

FIU/IHRC Public Hurricane Risk and Loss Model

Relogin Page

UserID:

PassWD:

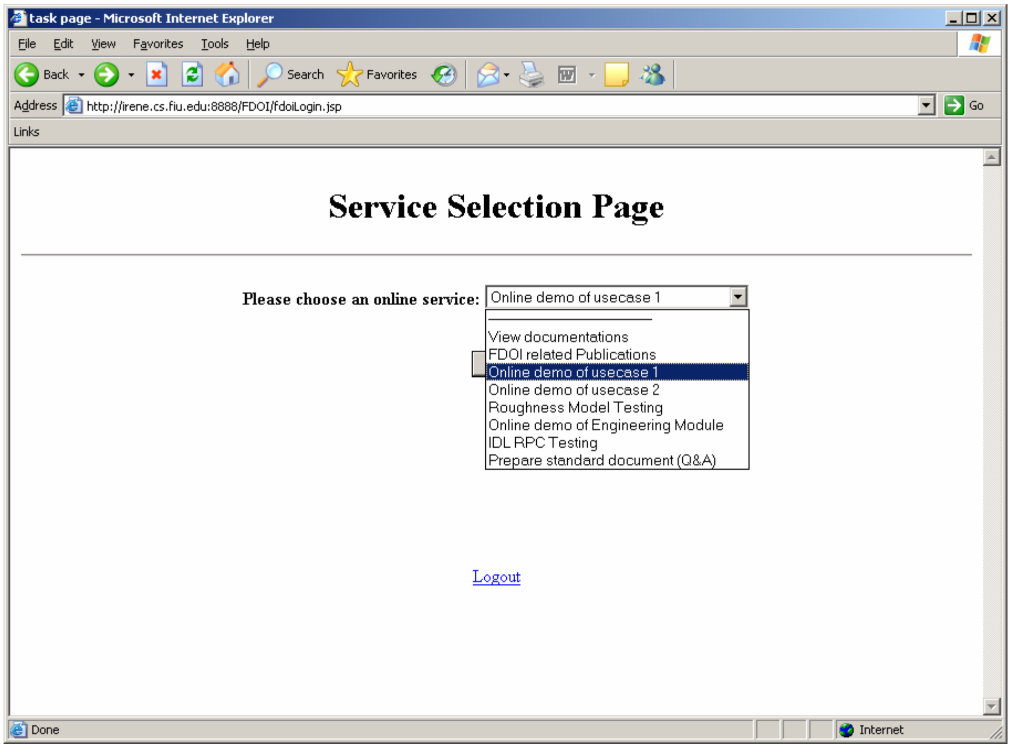
LOGIN

Wrong user name/password!

Figure 2.1.14. Snapshot of the Login Error Message

2.1.5.2. AHO page:

If the login is successful, the user can go to access Use Case One: Annual Hurricane Occurrence via selecting the option "Online Demo of Use Case 1" in the Service Selection Page as illustrated below.



task page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://irene.cs.fiu.edu:8888/FDOI/fdoiLogin.jsp
Go
Links
Done
Internet

Service Selection Page

Please choose an online service: Online demo of usecase 1

- View documentations
- FDOI related Publications
- Online demo of usecase 1
- Online demo of usecase 2
- Roughness Model Testing
- Online demo of Engineering Module
- IDL RPC Testing
- Prepare standard document (Q&A)

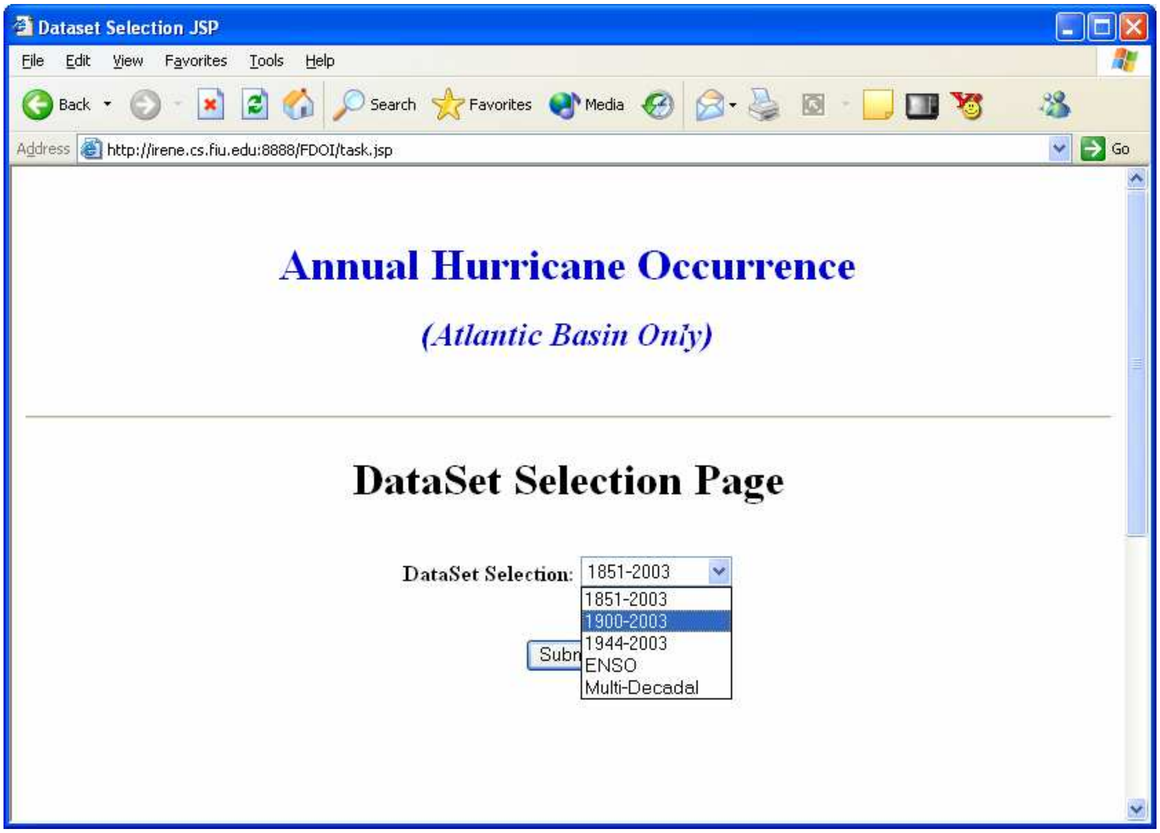
Logout

Figure 2.1.15. Snapshot of the service selection page

AHO is used to estimate the probability distribution for number of hurricanes per year and to generate a number of simulated years with its associated number of hurricanes based on the estimated probability distribution. Several steps are conducted to achieve that task.

Step 1:

To accomplish the above task, first the users need to select a year range. The Dataset Selection Page is designed for that purpose, which is the first page for AHO. Figure 2.1.16 illustrates the snapshot of the Dataset Selection Page.



http://irene.cs.fiu.edu:8888/FDOI/task.jsp

Annual Hurricane Occurrence

(Atlantic Basin Only)

DataSet Selection Page

DataSet Selection: 1851-2003

1851-2003
1900-2003
1944-2003
ENSO
Multi-Decadal

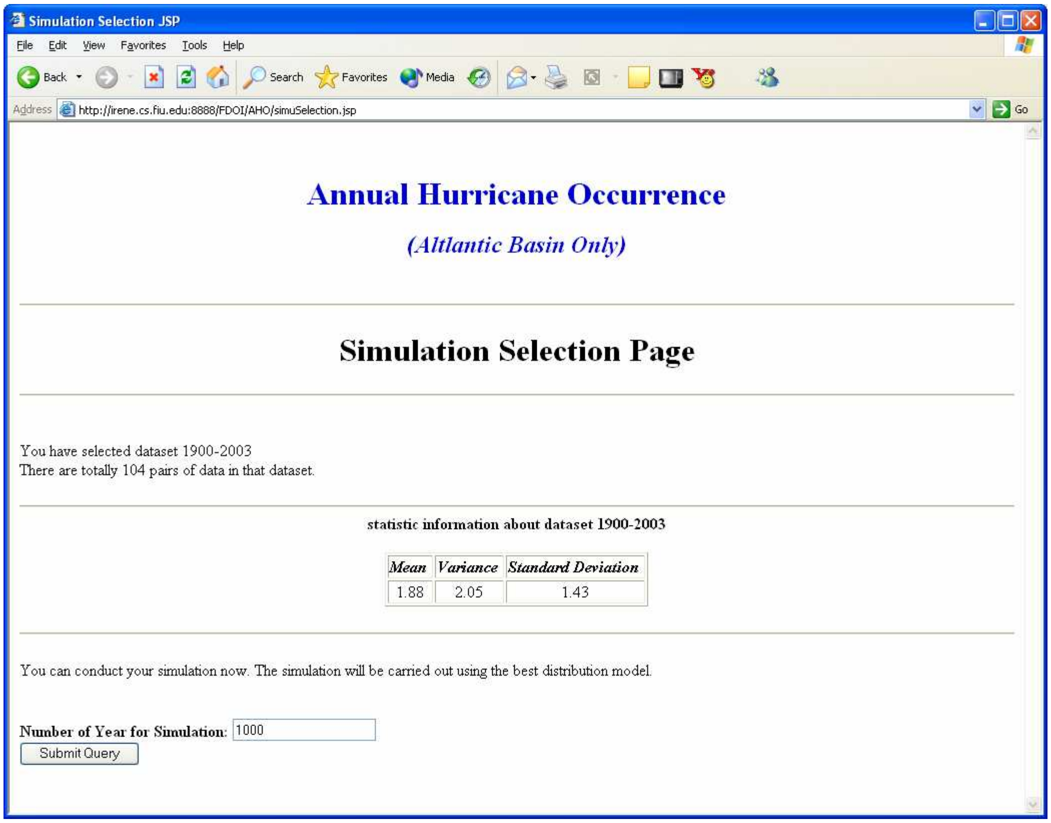
Figure 2.1.16. Snapshot of the first web page for AHO

A dropdown list containing all valid year ranges is provided to make the selection simpler to the users and to avoid any illegal year range specified by the user. There are five possible year ranges: 1851-2003, 1900-2003, 1944-2003, ENSO, and Multi-Decadal. (For detailed explanation of these terms, see to AHO design requirement part.) The user selects a year range from the dropdown list and submits his/her selection to the system.

Step 2:

Upon the user’s year-range selection, the system constructs a query and questions the underlying Oracle database to get the data set pertaining to the user’s year-range selection.

A statistical computation is carried out upon the retrieved data set to analyze its numerical characteristics; several statistical values of that data set such as mean value, variance and standard deviation are derived through this process and are displayed in the dynamically generated web page. Based on these retrieved historical data, the system also utilizes several stochastic probability distributions to fit the occurrence frequency.


Simulation Selection JSP
File Edit View Favorites Tools Help
Back
Search
Favorites
Media
Address
http://irene.cs.fiu.edu:8888/FDOI/AHO/simuSelection.jsp
Go

Annual Hurricane Occurrence

(Altlantic Basin Only)

Simulation Selection Page

You have selected dataset 1900-2003
There are totally 104 pairs of data in that dataset.

statistic information about dataset 1900-2003

Mean	*Variance*	*Standard Deviation*
1.88	2.05	1.43

You can conduct your simulation now. The simulation will be carried out using the best distribution model.

Number of Year for Simulation: 1000

Submit Query

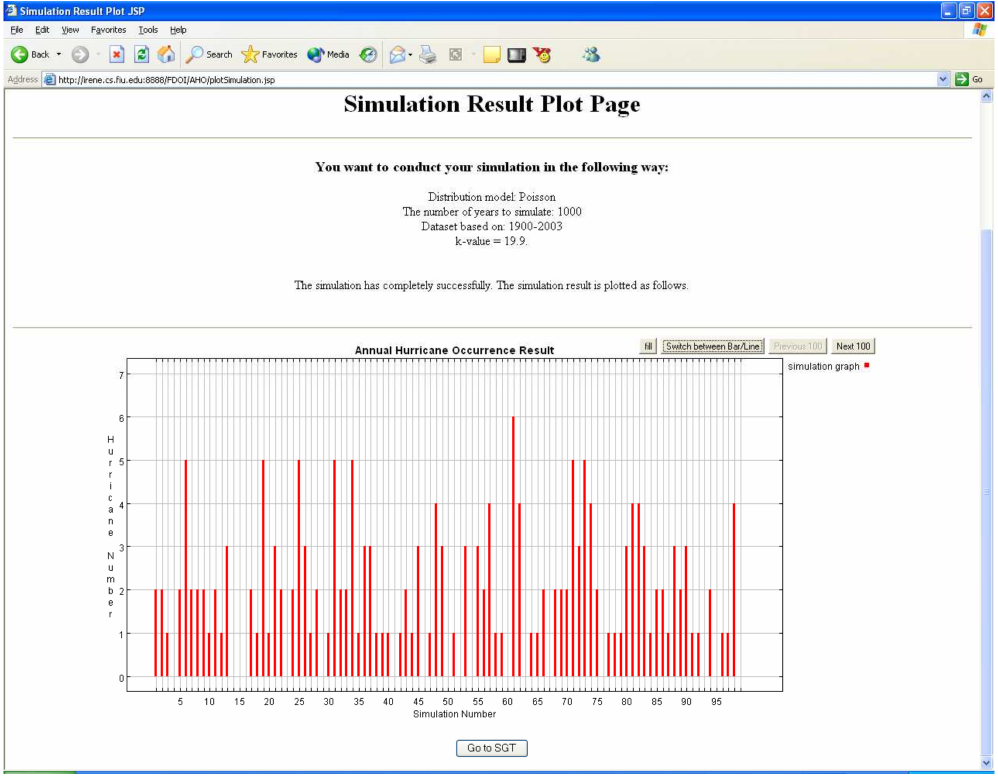
Figure 2.1.17: Snapshot of the second web page for AHO

The snapshot of the second web page is given in Figure 2.1.17. The upper part of the Simulation Selection Page has a table, which contains some statistical features about the selected data set. The lower part of the Simulation Selection Page offers the user a platform to compose and submit the simulation request. The user can specify his desired number of years for simulation. The simulation request is submitted to the system.

Step 3:

The system conducts the simulation with respect to the probability distribution that has the best goodness of fit and the number of simulated years that is determined by the user in the previous step. A series of years and their associated number of hurricane occurrences in that year are generated.

There is a new page for the simulation purpose; in that page the simulation result is plotted to offer better visual effect. The result is visualized 100 pairs of data per screen; the user can use the forward button to browse more and use the backward button to go back. Two different types of plots are supported: bar chart and line chart. Figure 2.1.18 – 2.1.19 illustrate respectively the snapshots of the bar plot example and line plot example.


Simulation Result Plot JSP
File Edit View Favorites Tools Help
Back
Search
Favorites
Media
Address http://irene.cs.fiu.edu:8888/FDOI/AHO/plotSimulation.jsp
Go

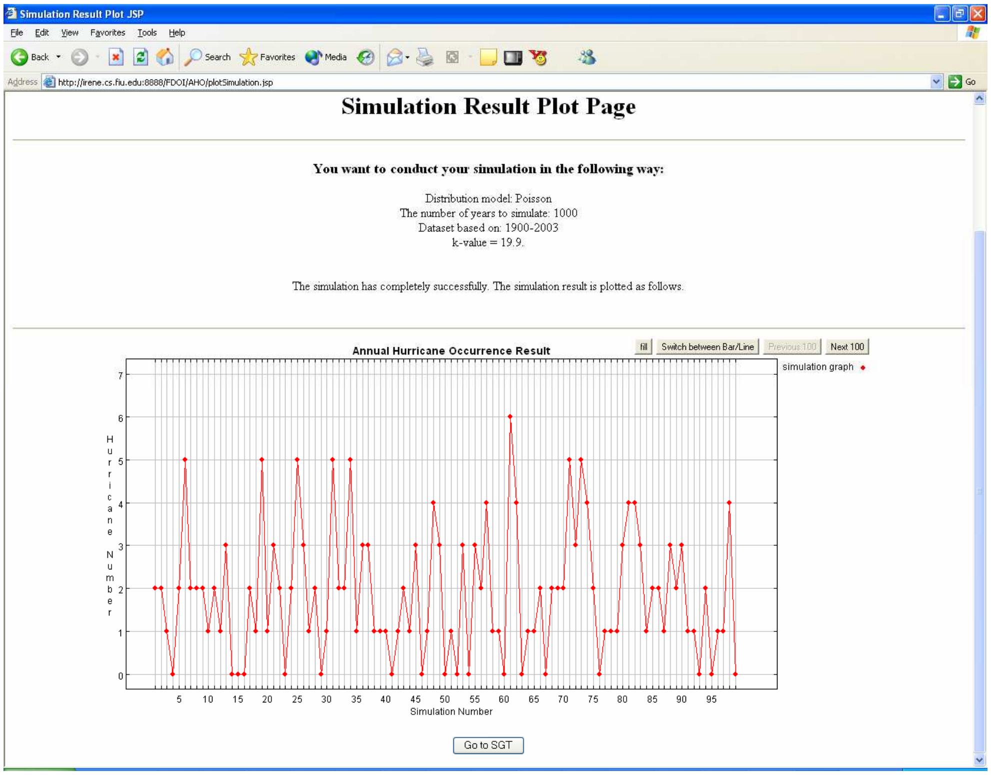
Simulation Result Plot Page

You want to conduct your simulation in the following way:

Distribution model: Poisson
The number of years to simulate: 1000
Dataset based on: 1900-2003
k-value = 19.9.

The simulation has completely successfully. The simulation result is plotted as follows.

Figure 2.1.18: Example of bar plot of the simulation result



Simulation Result Plot JSP
File Edit View Favorites Tools Help
Back
Search
Favorites
Media
Address http://irene.cs.fiu.edu:8888/FDOI/AHO/plotSimulation.jsp
Go

Simulation Result Plot Page

You want to conduct your simulation in the following way:

Distribution model: Poisson
The number of years to simulate: 1000
Dataset based on: 1900-2003
k-value = 19.9.

The simulation has completely successfully. The simulation result is plotted as follows.

Figure 2.1.19: Example of line plot of the simulation result

Section 2.2

Storm Genesis Time (SGT)
Use Case II

2.2.1 General Description Of SGT

SGT, short for Storm Genesis Time, is the second use case of the FIU/IHRC Public Hurricane Risk and Loss model. It aims at estimating the probability distribution for storm genesis time and generating the genesis time of a series of simulated hurricanes generated in Use Case One.

In this use case, only the historical hurricanes falling in threat area are considered. For the detailed documentation on threat area, please check use case 1 (AHO) documentation.

2.2.2 SGT General Requirements

Name: Storm Genesis Time (Threat Area Only)

Description: The end user enters a range of years and the system generates the following:

(1) A probability distribution for SGT (Storm Genesis Time).

(2) Genesis time of simulated hurricanes generated in Use Case One.

1. The end user enters a year range from the following selections:

1851-2003
1900-2003
1944-2003
Multi-Decadal
ENSO

NOTE:

Neutral Years: All non-ElNino and non-LaNina years are considered Neutral or average.

Multi-Decadal: Warm (and active), Cold (quiet). See Table 2.1.1 for a listing of Multi-Decadal year ranges.

ENSO: EL Nino, La Nina; see Table 2.2.2 for a listing of El Nino and La Nina years.

Table 2.2.1: Matrix of Multi-Decadal year ranges and temperature

Temperature (Warm)	Temperature (Cold)
1870-1902	1903-1925
1926-1970	1971-1994
1995-2003	

Table 2.2.2: Matrix of El Nino and La Nina years

El Nino - Year	La Nina - Year
1925	1933
1929	1938
1930	1942
1940	1944
1941	1945
1951	1948
1953	1949
1957	1950
1963	1954

1965	**1955**
1969	**1956**
1972	**1961**
1976	**1964**
1977	**1967**
1982	**1970**
1986	**1971**
1987	**1973**
1990	**1974**
1991	**1975**
1993	**1978**
1994	**1988**
1997	**1995**
	1998
	1999
	2000
	2002

2. Based on the user input from step 1, the system queries the database and the query results contain fix data for all the hurricanes. The query results consist of 9 columns. The names of these columns are as following: Storm ID, Storm Name, Genesis Date, Julian Date, Genesis Fix Time, Lat, Lon, Max Wind Speed, and Pressure. For example, in year 1851 there was one hurricane in the threat area. Table 2.2.3 illustrates the content of the returned data.

Table 2.2.3: Record of First Fix Data

Storm Id	Storm Name	Genesis Date*	Julian Date	Genesis Time	Lat	Lon	Max W/S	Pre
4	**Not Named**	**20-Aug-1851**	**2397355**	**180000**	**21.9**	**80.4**	**70**	**0**

3. The system uses data from the output of step 2 to calculate the hours between the genesis of each hurricane (in 6 hour resolution) and 0:00 hours May 01. The system generates a new matrix consisting of data from the output of step 2 and the calculated hours of each hurricane. The matrix also contains 9 columns. The column names are the same as those in step 2 (See Table 2.2.4).

   Each day storm data is collected in the one of the following intervals: I1=[0:00, 6AM), I2=[6AM, 12 Noon), I3=[12Noon, 6PM), I4=[6PM, midnight). For the sake of simplicity, each interval is associated with its starting point. So, for example, since the hurricane with Storm ID 4 happened in the interval I3 on August 20, 1851, the number of hours recorded for this hurricane will be 24*(2397355 (Julian date of 20-Aug-1851) - 2397243 (Julian date of 1-May-1851)) + 18 = 2706.

Table 2.2.4: New Record of First Fix Data

Storm ID	Storm Name	Genesis Date*	SGT	Genesis Time	Lat	Lon	Max W/S	Pre
4	**Not Named**	**20-Aug-1851**	**2706**	**180000**	**21.9**	**80.4**	**70**	**0**

4. The system stores in the database the calculated hours between the genesis of hurricanes and 0:00 hours May 01.

5. The system uses the data from the output generated in step 3 to estimate the probability distribution of SGT. In the following, we denote the random variable of SGT by T .

   5.1. The system calculates the number of years of the year range the user entered at step 1. Let it denoted by *M*.

   5.2. The system calculates the number of hurricanes in each year in the year range. Let n_i denote the number of hurricanes in year i .

   5.3. The total number of hurricanes we have is $N = \sum_i n_i$

   5.4. The system sorts all the hurricanes in ascent order according to their SGT. Assume now that these hurricanes occurred at times $0 \le T_1 \le T_2 \le T_3 \cdots \le T_W$, where $W \le N$. The system also calculates the number of hurricanes that occurred at time T_i . Let it denoted by f_i .

   5.5. The system calculates the empirical CDF (Cumulative Distribution Function) for T , an estimate of the true CDF $F(t) = P(T \le t)$ using the following equation:

$$F_N(t) = \begin{cases} 0, & \text{if } t < T_1 \\ \dfrac{f_1 + f_2 + \cdots + f_i}{N} & \text{if } T_i \le t < T_{i+1}, i = 1,2,\cdots,W-1 \\ 1, & \text{if } t > T_W \end{cases}$$

   5.6. The system calculates the smooth estimator of $F(t)$. For a suitable kernel function K and a positive bandwidth sequence $h_N(t)$, (Note that h_N is a function of the point t and the sample size N).

5.7. This estimator, denoted by $\hat{F}_N$ is defined as:

$$\hat{F}_N(t) = \int_0^\infty \frac{1}{h_N(t)} K((t-x)/h_N(t)) F_N(x)dx$$
$$= \sum_{j=1}^{W} S_j K^*\left(\frac{t-T_j}{h_N(t)}\right)$$

Where S_j is the jump of F_N at T_j, that is,

$S_j = F_N(T_j) - F_N(T_{j-1}), j = 2,3,\cdots,W$ and $S_1 = F_N(T_1)$. Also $k^*(u)$ is the integral of $K(x)$, that is, $k^*(u) = \int_{-\infty}^{u} K(x)dx$

5.8. In the above function, one has a wide variety of choices available for the kernel function and the corresponding bandwidth. We will try the following kernel function and bandwidth:

a) The kernel function is the Epanechnikov kernel K, that is,

$$K(x) = 3/4\sqrt{5}\,(1 - x^2/5), -\sqrt{5} < u < \sqrt{5}$$

b) The LOCAL bandwidth $h_N(t) = \frac{S}{2} * \left(\frac{1}{N}\right)^{1/3}$, where S is the standard deviation of the calculated SGTs of all the hurricanes.

6. The system presents a list of simulated events sets.
7. The user selects a set of simulated events and submits it to system.

8. The system checks the selected simulated events. If they already exist in database, the system query the database to get the data of selected simulated years; or, the system triggers Use Case One (AHO) to generate a set of simulated events.

9. The system uses **IMSL** routines to generate the SGT for each hurricane of the selected simulated events. The selected simulated events give the number of hurricanes in a given year. Assume there are M_i hurricanes in year i. The system will sample M_i hurricanes from CDF $\hat{F}_N(t)$ to get the genesis time for those M_i hurricanes in year i.

10. The system stores the generated SGT into database.

11. The system displays the generated SGT and the corresponding simulated events on screen.

Note:

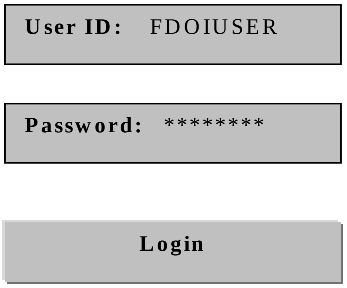
Steps-2 – 11 are repeated for each year range that the user requests.
Steps-7– 11 are repeated each set of simulated events the user selects at step 6.

2.2.2.1 SGT Interface Design Requirements

This part designs the GUI (Graphic User Interface) for the Storm Genesis Time (SGT). It describes the process by which scientists or statisticians log in to the PHLRM system to view the genesis time of events generated in Use Case One (AHO) if they exist or to trigger the events if they do not exist.

A. The First step: the user logs in the system

Figure 2.2.1 is the Login Interface. The user enters the User ID, enters a password and submits to the system. The system checks the user name and password and let the user log into the system if the User ID and Password are correct. Or, the system gives the "invalid user/password" error to the user and asks the user to reenter the User ID and password.



User ID: FDOIUSER

Password: ********

Figure 2.2.1: Login Interface

B. The second Step: the user selects a year range

Figure 2.2.2 is the Year Selection Interface. The system presents a list of year ranges to the user. The correct year ranges are shown at Table 2.2.5. The user selects a year range and submits to the system. The system also should provide the user the option to go back to the first step or to quit the system.

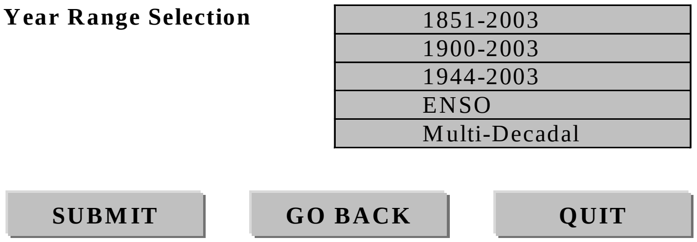


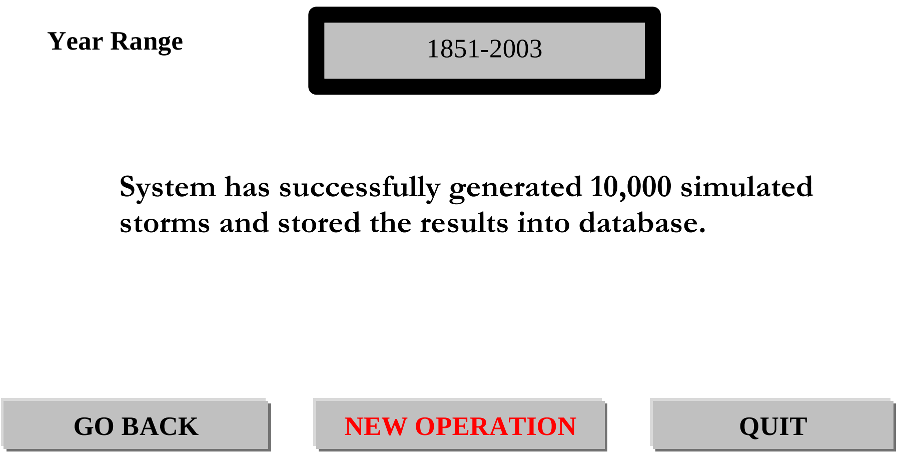
Figure 2.2.2: Dataset Selection Interface

Table 2.2.5: Valid year ranges

Valid Year Range
1851-2003
1900-2003
1944-2003
ENSO
Multi-Decadal

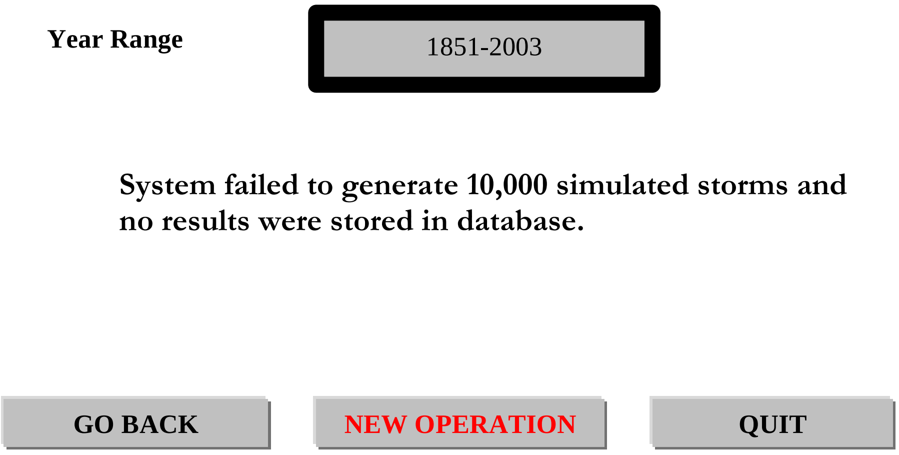
C. The third step: storm genesis time results display

After the user submits the selected year range, the system generates the storm genesis time for each simulated hurricane, store the results into database and display the success message to the user if succeeded or display error message when failed. The system also should provide the user the option to go back to the third step, to start a new operation or to quit the system. Figure 2.2.3 is the Simulation Results Display Interface in the case of success. Figure 2.2.4 is the Simulation Results Display Interface in the case of success.



System has successfully generated 10,000 simulated storms and stored the results into database.

Figure 2.2.3: Storm Genesis Time Results Display Interface (in case of success)



Year Range 1851-2003

System failed to generate 10,000 simulated storms and no results were stored in database.

Figure 2.2.4: Storm Genesis Time Results Display Interface (in case of failure)

2.2.3 Computer Model Design

2.2.3.1 Use Case View of SGT

A. Actors:

There is one actor (scientists) in SGT. They will use this use case to estimate the probability distribution model and to generate the storm genesis time of the simulated hurricanes generated in Use Case One (AHO).

B. Use Case:

SGT is used to estimate the probability distribution model for HBG (Hours between Genesis) and generate the genesis time of a series of simulated hurricanes generated in Use Case One.

C. Use Case Diagram:

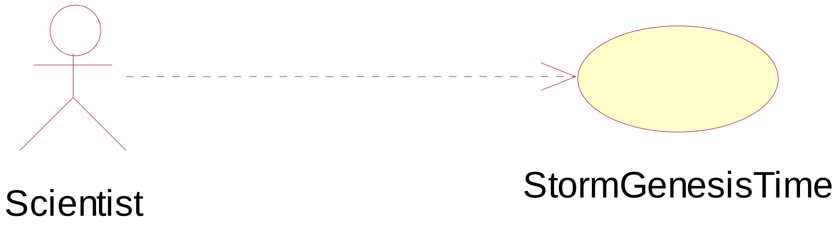


Figure 2.2.5: Use Case Diagram for SGT

2.2.3.2 System Design

This part includes the appropriate diagrams to describe the system classes, components, activities and the overall flow chart of SGT.

2.2.3.3 Program Flow Chart of SGT

Here we give out the flow chart of SGT. we could see clearly from the chart the relations of different parts.

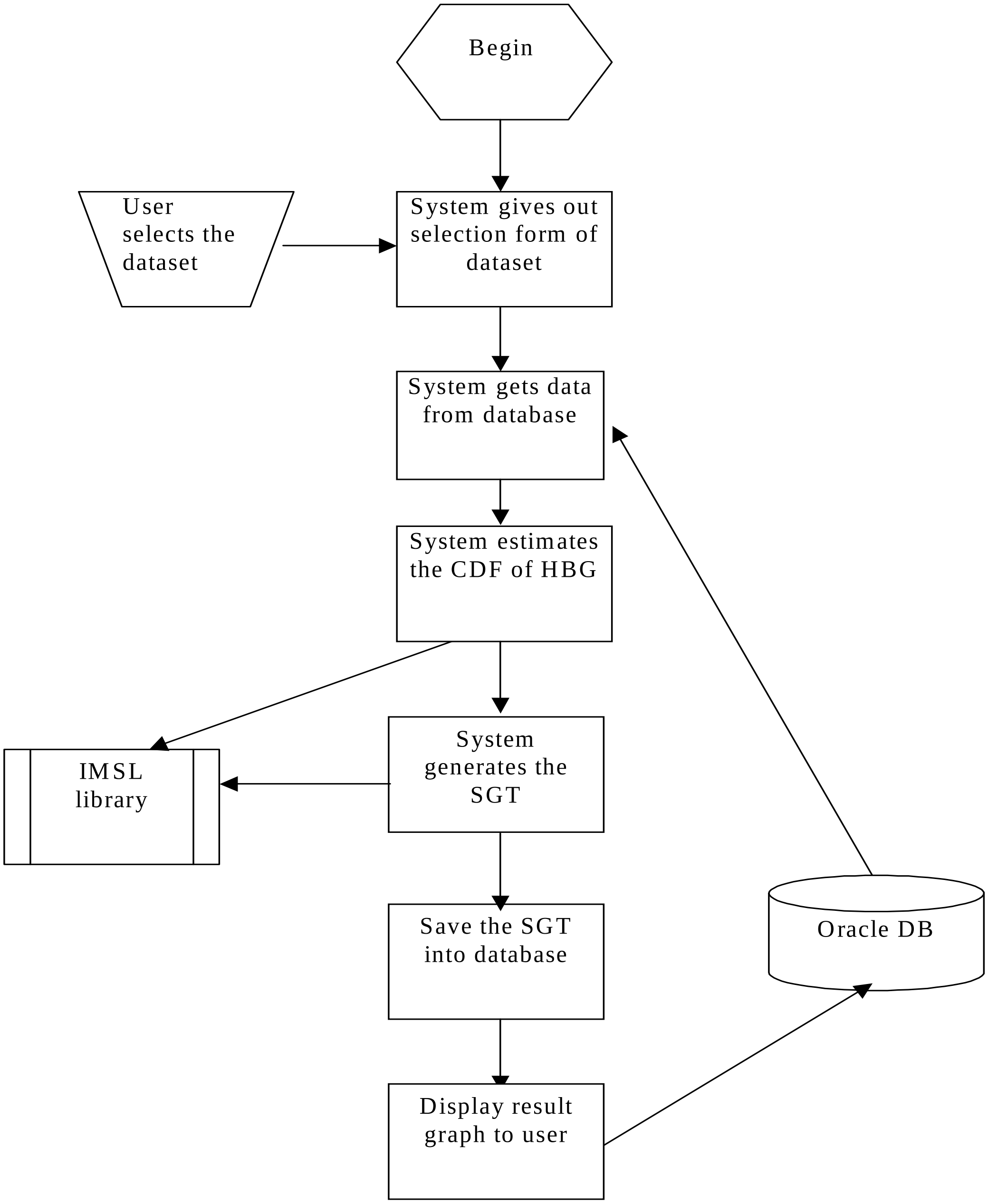

Begin
User selects the dataset
System gives out selection form of dataset
System gets data from database
System estimates the CDF of HBG
IMSL library
System generates the SGT
Save the SGT into database
Oracle DB
Display result graph to user

Figure 2.2.6: Flow chart of SGT

2.2.3.4 Class Diagram and Description

A. Class Diagram

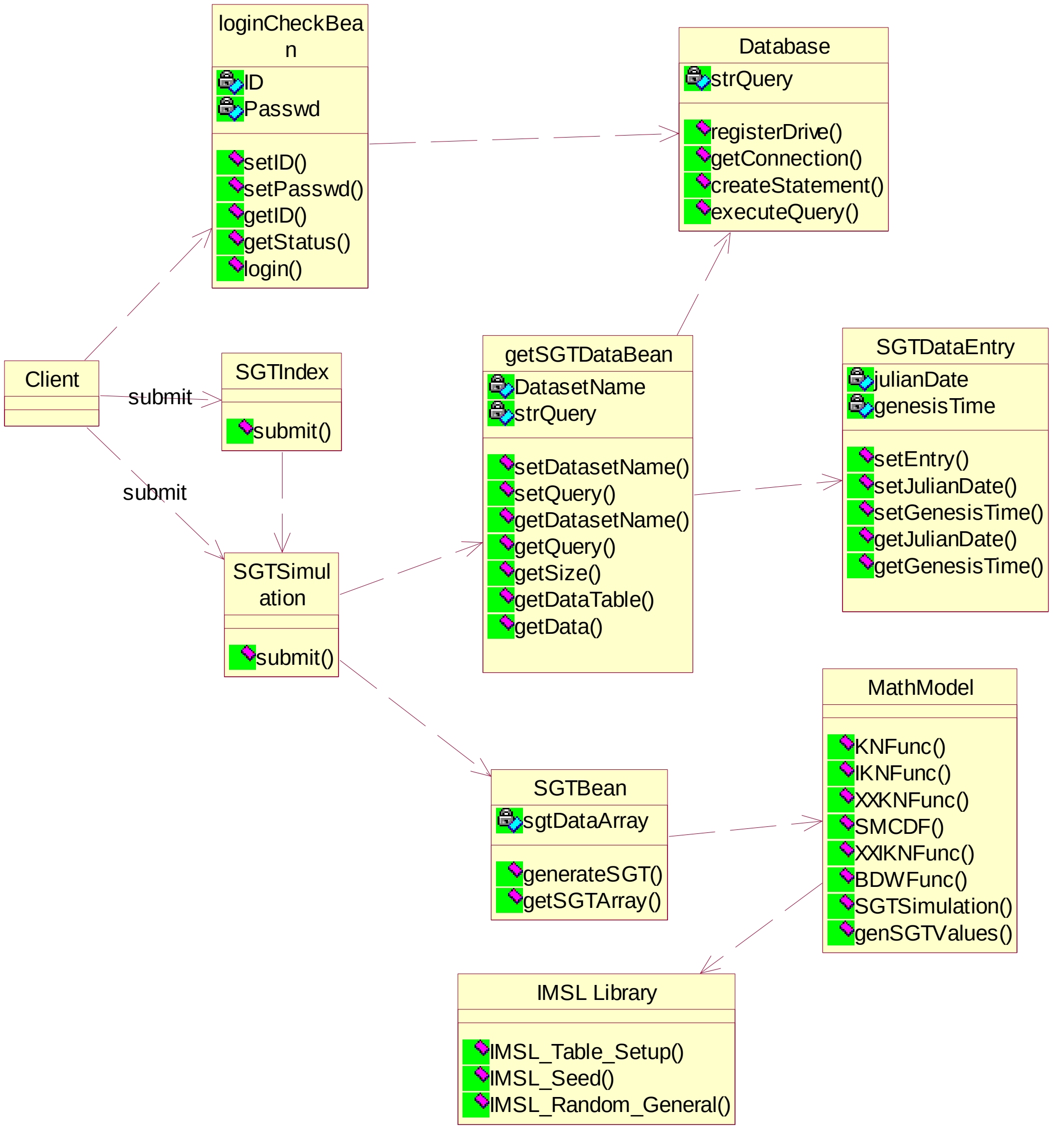


Figure 2.2.7: Class Diagram for SGT

B. Classes Descriptions

Here we would like to give a brief introduction of the functions of the classes we use. Generally, our design follows the flow chart we developed.

- **Client**:
  A virtual class. It means the user who uses this system. Or, we could say, it is the web browser of user machine. No need to implement it.

- **LoginCheckBean**:
  This class is in charge of user login. It gets the username (ID) and password the user enters, and then checks with the information stored in Oracle Database. We will explain the login process later with a sequence diagram.

- **SGTIndex**
  This class is used to get user selections (e.g. year range). It will then pass control to classes that will get data from database.

- **SGTSimulation**:
  This class will call other classes to get the needed data from database, and then call the related classes to generate the SGT, and then display the results using a table on end user's web browser.

- **getSGTDataBean**:
  The true class to get related hurricane information from database.

- **SGTDataEntry**:
  It is a class served for get the Julian Date and Genesis Time.

- **Database**:
  It is an abstract concept. It includes all the system provided database operations.

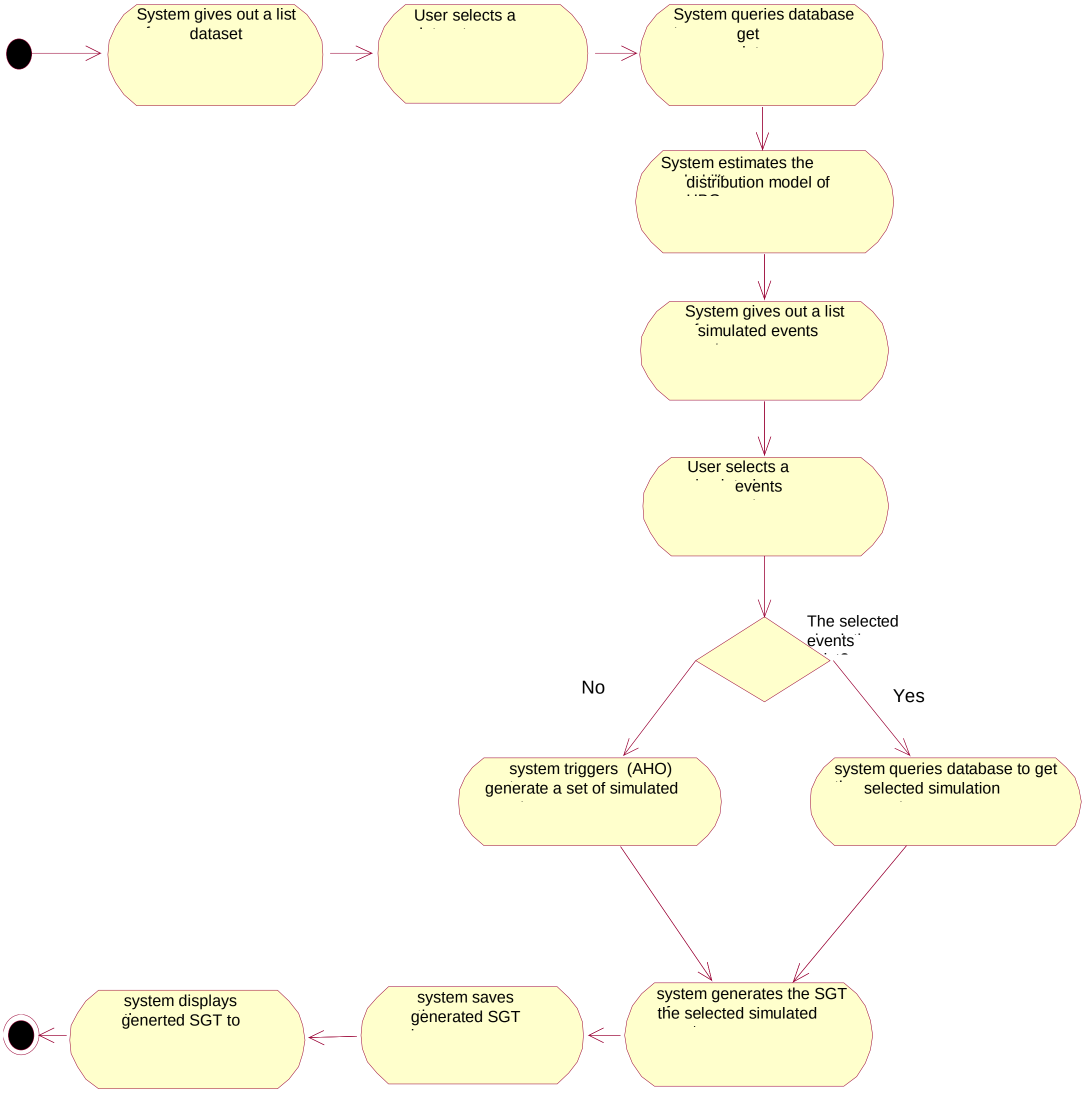
- **SGTBean**:
  The class is used to interface with the true math model.

- **MathModel**:
  The C++ class using IMSL library functions to fit distribution and generating the simulation. It will communicate with the Java main application using JNI interface.

- **IMSL Library**:
  Library functions provided by IMSL.

2.2.3.5 Activity Diagram

Figure 2.2.8 depicts the activity diagrams consisting of the major activities in Use Case Two. This activity diagram offers a clear and direct understanding of the business logic of Use Case Two.



System gives out a list dataset
User selects a
System queries database get
System estimates the distribution model of
System gives out a list simulated events
User selects a events
The selected events
No
Yes
system triggers (AHO) generate a set of simulated
system queries database to get selected simulation
system generates the SGT the selected simulated
system saves generated SGT
system displays generted SGT to

Figure 2.2.8: Activity Diagram for SGT

2.2.3.6 Sequence Diagram

Similar to use case one, we will give out the sequence diagrams for the three major activities in use case two, which are **login**, **simulation** and **generate genesis time**. Because the login process is the same for all use cases, the **login process** is the same as in use case one.

A. Login Process

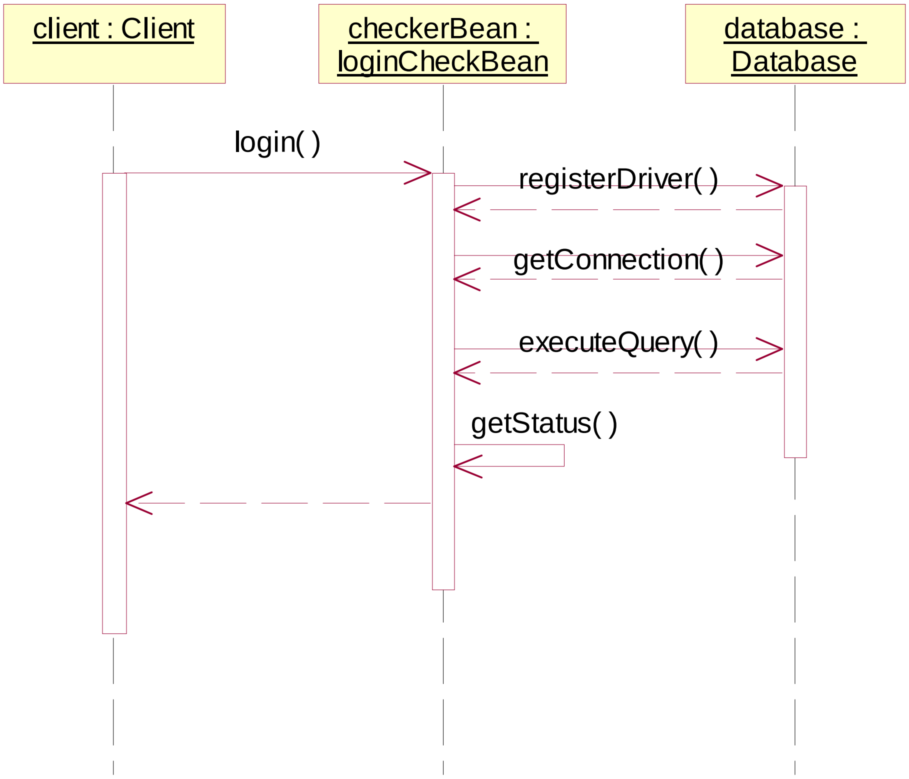


Figure 2.2.9: Sequence diagram for login process

- **Step1**: The user enters its username and password in the web browser and click “login” button

- **Step 2**: This information is passed to the **loginCheckBean** class, which really communicates with the database. If the password or username is not matched with the information stored in the database, the user will get an error message and be asked to login again. If the username and password are matched, user can continue to access all use cases of the system.

B. Simulation Process

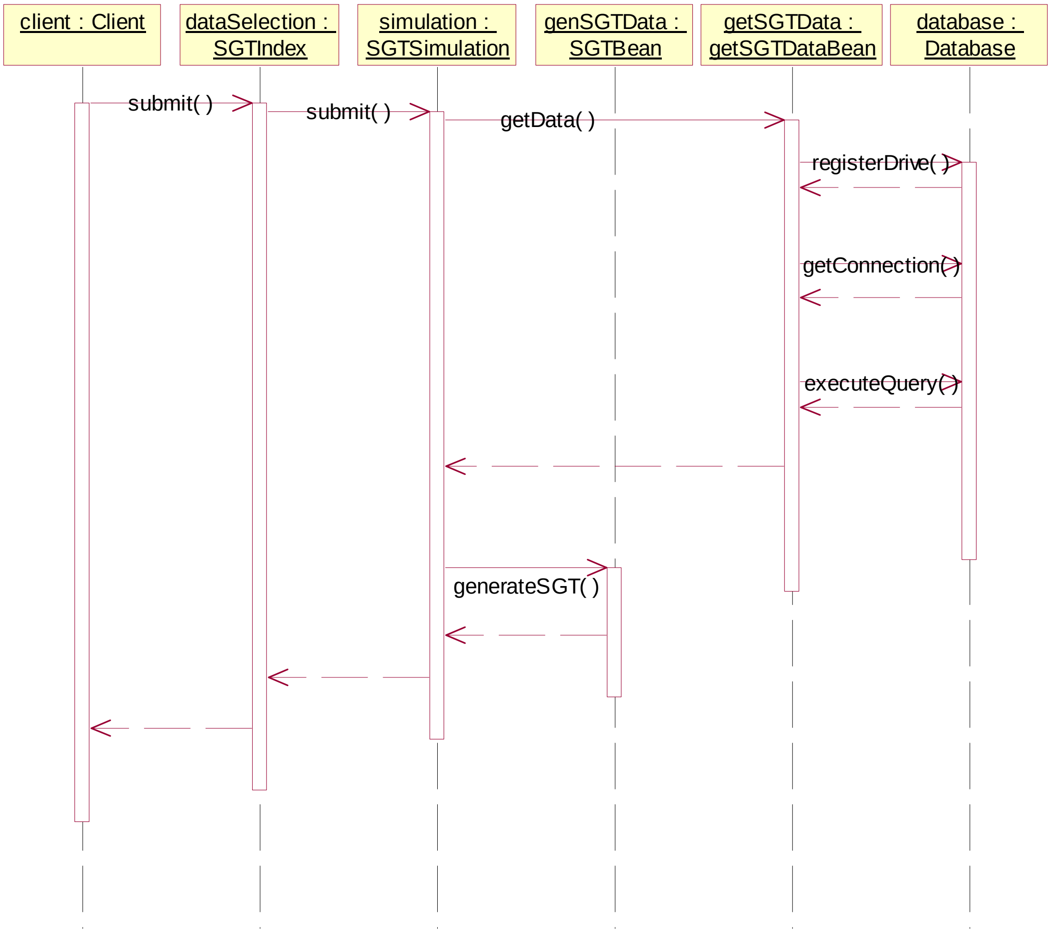


Figure 2.2.10: Sequence diagram for simulation process

- **Step1**: The user selects the data set (e.g. 1851-2001), and then clicks the "Submit" button.

- **Step 2**: The **dataSelection** object verifies the data set the user selected, and then calls the **simulation** object to get the related data from database.

- **Step 3:** The **simulation** object will call the **getSGTData** object, which will connect with the database, create the query, and then get the desired data from database.

- **Step 4:** The **simulation** object will call **genSGTData** object to generate the SGTs.

C. Generate Genesis Time

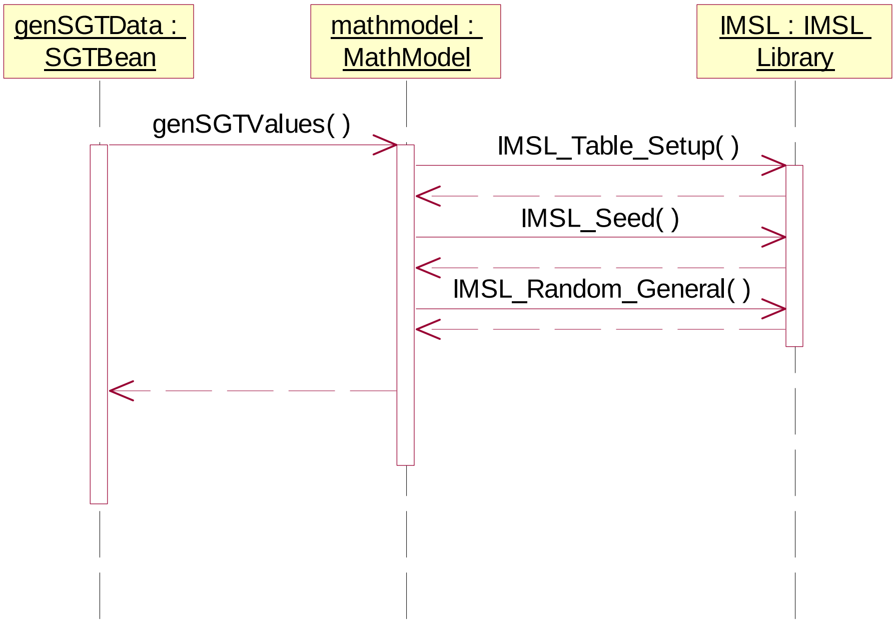


Figure 2.2.11: Sequence diagram for simulation process

- **Step1**: The **genSGTData** object calls the **mathmodel** object to generate the SGTs.

- **Step2:** The **mathmodel** object will do all the pre-process work and then pass the related parameters to the **IMSL** library functions.

- **Step3:** The **IMSL** library functions will create the related data and then pass the data back to the **mathmodel** object. Then an array will be returned to the **genSGTData** object that contains the generated SGTs.

2.2.4 Implementation of SGT

Currently the implementation for Use Case two (SGT) has been finished. The demo is online at http://www.cs.fiu.edu/PHRLM.

2.2.4.1 Login page:

The users need a username and a password to access the *FIU/IHRC Public Hurricane Risk and Loss Model*. Following is the snapshot of the web page for login purpose.

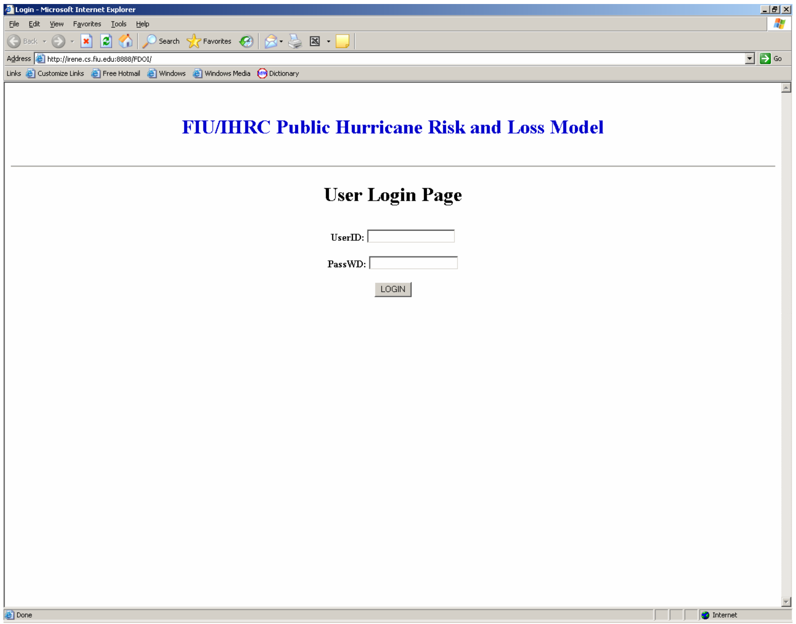
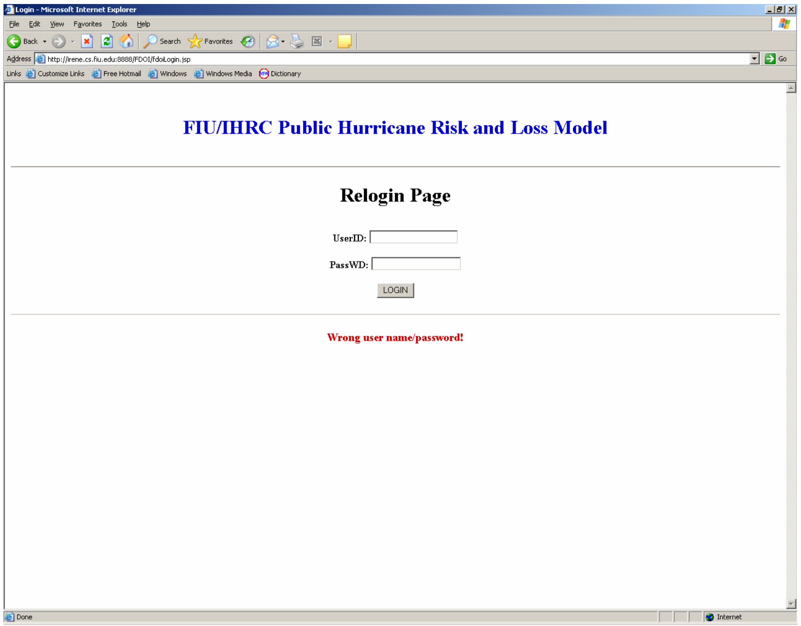

Login - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://irene.cs.fiu.edu:8888/FDOI/
Go
Links Customize Links Free Hotmail Windows Windows Media Dictionary
FIU/IHRC Public Hurricane Risk and Loss Model
User Login Page
UserID:
PassWD:
LOGIN
Done
Internet

Figure 2.2.12. Snapshot of Login page for SGT

If the username/password is wrong, error message will show and the user is required to input the username and password again.


Login - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://irene.cs.fiu.edu:8888/FDOI/fdoiLogin.jsp
Go
Links Customize Links Free Hotmail Windows Windows Media Dictionary
Done
Internet

FIU/IHRC Public Hurricane Risk and Loss Model

Relogin Page

UserID:

PassWD:

LOGIN

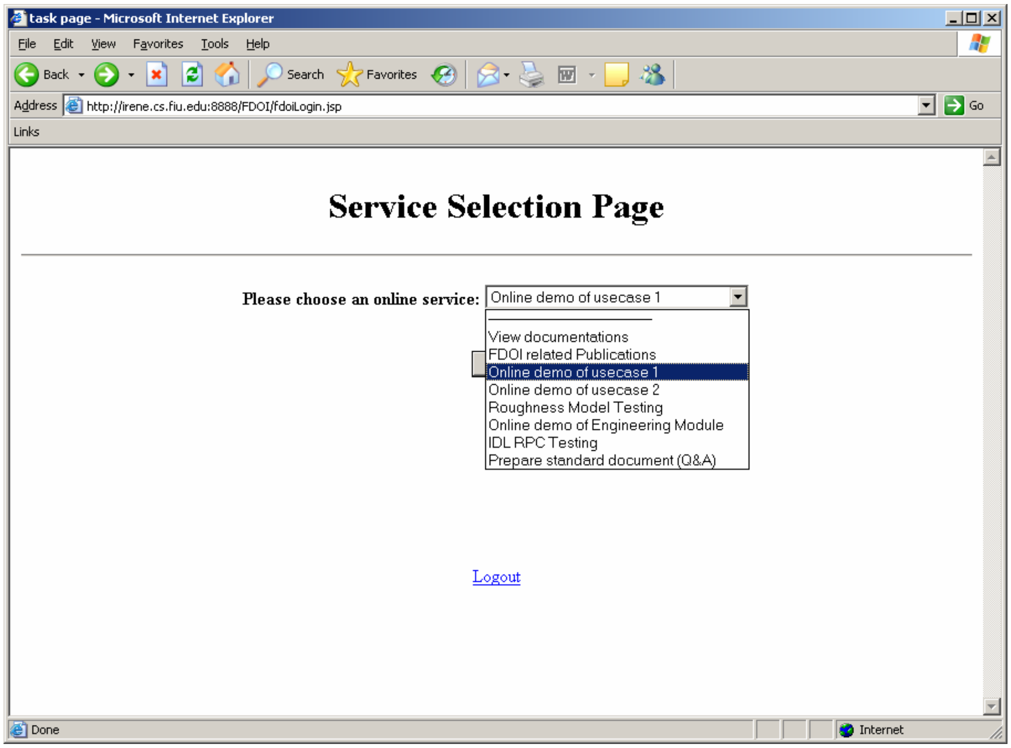
Wrong user name/password!

Figure 2.2.13. Snapshot of the Login error page for SGT

2.2.4.2 SGT page:

If the login is successful, the user can go to access Use Case One: *Annual Hurricane Occurrence* via selecting the option "Online Demo of Use Case 2" in the Service Selection Page as illustrated below.

Use Case Two is used to estimate the storm genesis time and generate corresponding time for simulated hurricanes which are obtained from Use Case One: *Annual Hurricane Occurrence*. Several steps are conducted to achieve that task.


task page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Address
http://irene.cs.fiu.edu:8888/FDOI/fdoiLogin.jsp
Go
Links
Done
Internet

Service Selection Page

Please choose an online service: Online demo of usecase 1

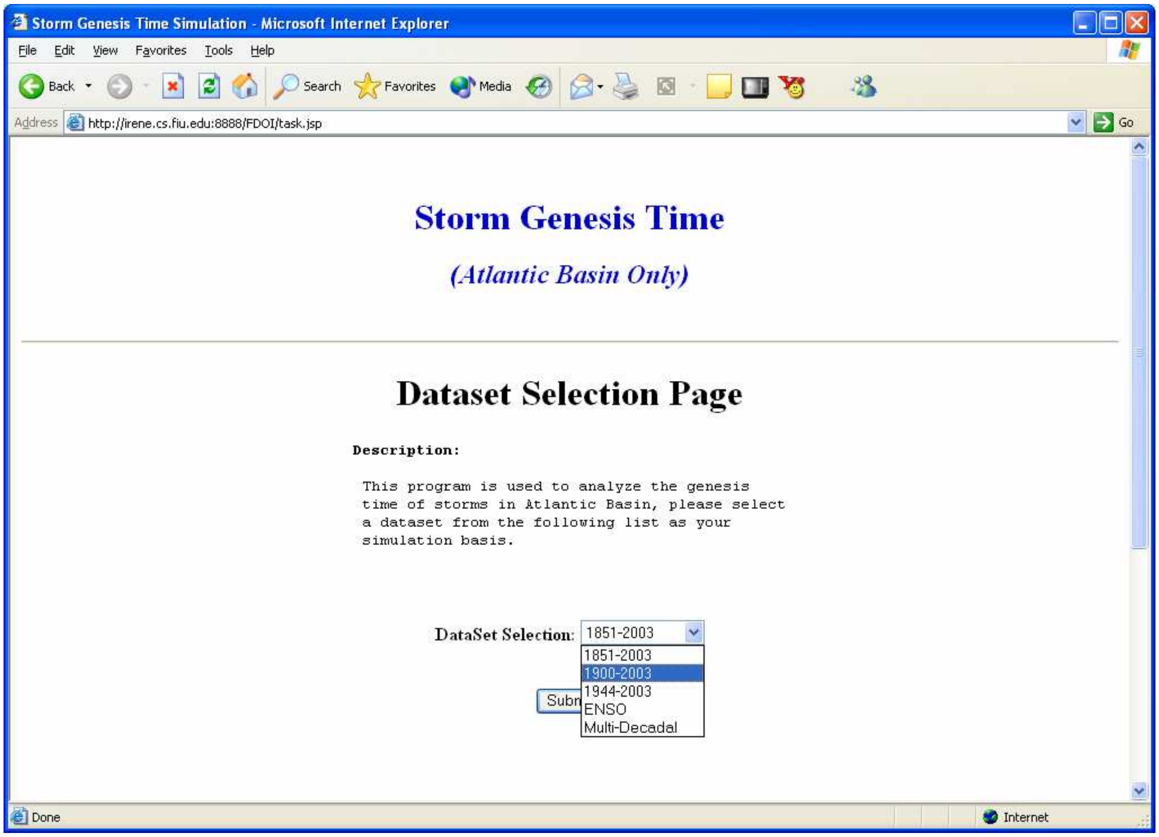
- View documentations
- FDOI related Publications
- Online demo of usecase 1
- Online demo of usecase 2
- Roughness Model Testing
- Online demo of Engineering Module
- IDL RPC Testing
- Prepare standard document (Q&A)

Logout

Figure 2.2.14. Snapshot of the Service Selection Page

Step 1:

To accomplish the above task, first the users need to select a year range. The Dataset Selection Page is designed for that purpose, which is the first page for Use Case Two. Figure 2.2.15 illustrates the snapshot of the Dataset Selection Page.


Storm Genesis Time Simulation - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://irene.cs.fiu.edu:8888/FDOI/task.jsp
Go
Subm
Done
Internet

Storm Genesis Time

(Atlantic Basin Only)

Dataset Selection Page

Description:

This program is used to analyze the genesis time of storms in Atlantic Basin, please select a dataset from the following list as your simulation basis.

DataSet Selection: 1851-2003

- 1851-2003
- 1900-2003
- 1944-2003
- ENSO
- Multi-Decadal

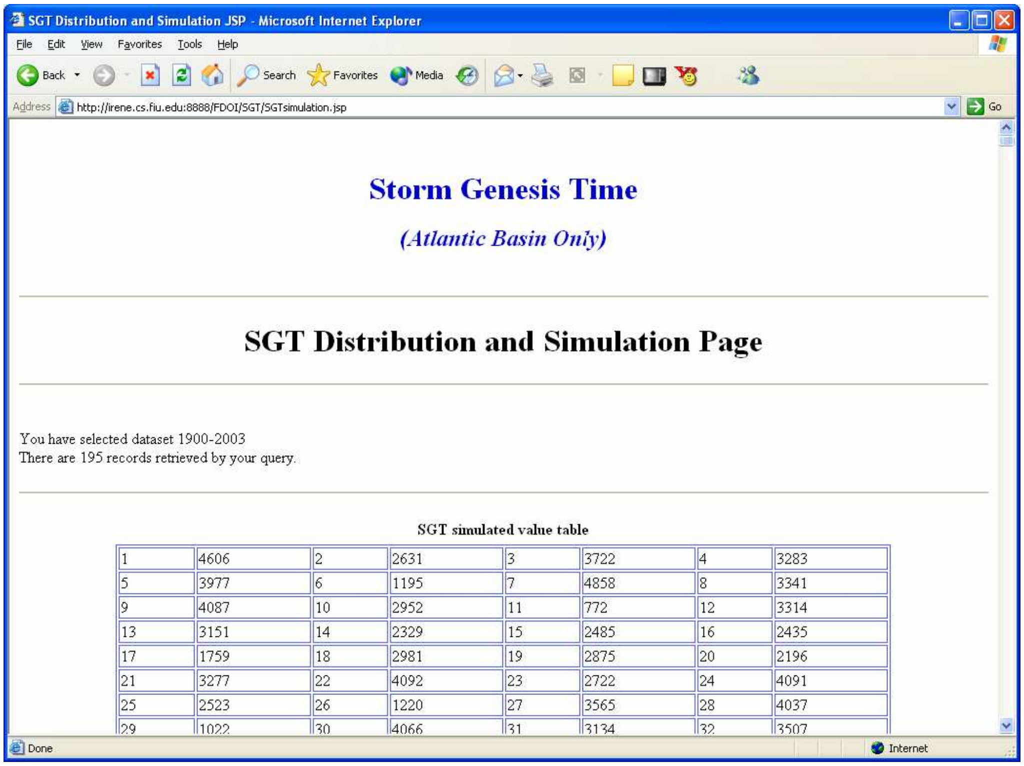
Figure 2.2.15. Snapshot of the first web page for SGT

A dropdown list that is consisted of some possible year ranges is offered to make the selection simpler to the users and, on the other side, to avoid the user type in any wrong year range. There are five possible choices: 1851-2003, 1900-2003, 1944-2003, ENSO, and Multi-Decadal. (Please see to user requirement documentation for detailed explanation of these terms.) The user then selects a year range from the dropdown list and submits his/her selection to the system.

<u>Step 2:</u>

Upon the user's year-range selection, the system constructs a query and questions the underlying Oracle database to get the data set pertaining to the user's year-range selection which contains the hurricanes and their related Julian date, the first fixed time and so on.

The system uses the specific stochastic approaches to fit the storm genesis time based on the historical data retrieved from the Oracle database according to the user's year-range selection. Then the system generates a sequence of genesis time for the simulated hurricanes produced in Use Case One: Annual Hurricane Occurrence. Figure 2.2.16 and 2.2.17 depict the snapshot of some final result after running SGT. The first 100 storm genesis time obtained was displayed in a table for both debugging and demonstrating purpose.


SGT Distribution and Simulation JSP - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://irene.cs.fiu.edu:8888/FDOI/SGT/SGTsimulation.jsp
Go
Done
Internet

Storm Genesis Time

(Atlantic Basin Only)

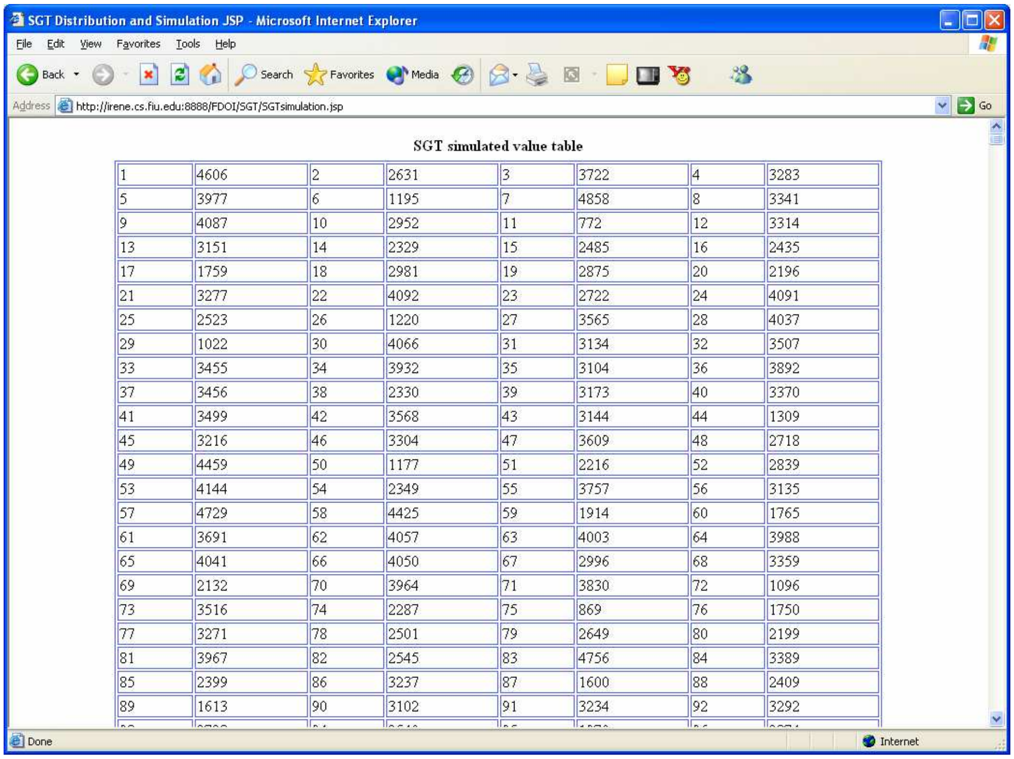
SGT Distribution and Simulation Page

You have selected dataset 1900-2003
There are 195 records retrieved by your query.

SGT simulated value table

1	4606	2	2631	3	3722	4	3283
5	3977	6	1195	7	4858	8	3341
9	4087	10	2952	11	772	12	3314
13	3151	14	2329	15	2485	16	2435
17	1759	18	2981	19	2875	20	2196
21	3277	22	4092	23	2722	24	4091
25	2523	26	1220	27	3565	28	4037
29	1022	30	4066	31	3134	32	3507

Figure 2.2.16: Snapshot of the result page for Use Case Two


SGT Distribution and Simulation JSP - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://irene.cs.fiu.edu:8888/FDOI/SGT/SGTsimulation.jsp
Go
Done
Internet

SGT simulated value table

1	4606	2	2631	3	3722	4	3283
5	3977	6	1195	7	4858	8	3341
9	4087	10	2952	11	772	12	3314
13	3151	14	2329	15	2485	16	2435
17	1759	18	2981	19	2875	20	2196
21	3277	22	4092	23	2722	24	4091
25	2523	26	1220	27	3565	28	4037
29	1022	30	4066	31	3134	32	3507
33	3455	34	3932	35	3104	36	3892
37	3456	38	2330	39	3173	40	3370
41	3499	42	3568	43	3144	44	1309
45	3216	46	3304	47	3609	48	2718
49	4459	50	1177	51	2216	52	2839
53	4144	54	2349	55	3757	56	3135
57	4729	58	4425	59	1914	60	1765
61	3691	62	4057	63	4003	64	3988
65	4041	66	4050	67	2996	68	3359
69	2132	70	3964	71	3830	72	1096
73	3516	74	2287	75	869	76	1750
77	3271	78	2501	79	2649	80	2199
81	3967	82	2545	83	4756	84	3389
85	2399	86	3237	87	1600	88	2409
89	1613	90	3102	91	3234	92	3292

Figure 2.2.17: Snapshot of the result table in the result page for Use Case Two

Section 3

Wind Field Module
Module II

Section 3.1

Storm Track Model
Use Case III

3.1.1 General Description of Storm Track Model

Strom track model is aimed at generating the storm tracks for simulated storms based on data obtained from Use Case II and stochastic algorithms.

The storm track model consists of two main components: the empirical probability distribution generator (GENPDF), and the storm track generator (STORMGEN). Descriptions of these components are given below.

3.1.2 Technical Description of the Storm Track Model

3.1.2.1 The empirical probability distribution generator (GENPDF)

This component derives the probability distribution functions (PDFs) from the historical record (HURDAT) that are subsequently used by the STORMGEN track generator. The PDFs are conditional probabilities, as they depend on location, time of season and other parameters. The PDFs are empirical in that they are obtained by discrete binning. The following PDFs are derived:

- Initial storm speed
- Initial storm direction
- Initial storm intensity (pressure)
- Change in storm speed
- Change in storm angle
- Change in storm intensity (pressure and relative intensity)

The bin size and location of these PDFs are defined in a header file“genpdf.h” which is used by both GENPDF and STORMGEN. The bins may be linearly on nonlinearly spaced. A mapping function is available which allows nonlinear mapping so that higher resolution (of a particular parameter) may be obtained.

Storm genesis is defined to occur when a storm first enters or appears within the threat area and has a minimum wind speed of 64 kt. The threat area is described in Section 2.1. The HURDAT database contains a variety of storm report types:

- “E” - extratropical
- “L” - low
- “D” - depression
- “S” - subtropical
- “W” - wave
- tropical – pressure reports
- tropical – wind reports

All non-tropical storm reports (“E”,”L”,”D”,”W”,”S”) are excluded in the intensity PDFs. Pressure reports are used whenever available. If a pressure report is not available, then an attempt is made to interpolate from reports that are within a 24 hour period including the target report. Otherwise, pressure is obtained using an empirical wind-pressure relation (see Appendix A). Intensity changes are only computed for similar report types – observed pressure or wind-derived pressures. Mixing observed and wind-derived pressures was found to create spurious pressure changes. Pressures over land were excluded.

Due to sparsity of data in some regions or parameter space, the PDFs may be coarsened (bins widened) so that a sufficient number of observations are available to create a robust PDF. This is done in the RESIZE function in GENPDF.

Pressure changes are converted to relative intensity changes. The relative intensity calculation is described in Appendix B. PDFs for pressure and relative intensity are created, though only one is used in STORMGEN. By default, the relative intensity PDF is used by STORMGEN.

♦ Input Data

GENPDF requires the following input files:

- The HURDAT database
- A control file which contains the dates of the historical record to use
- Land Mask file – the land mask is based on USGS land use data.
- Outflow temperature for the relative intensity calculation (see Appendix B)
- Sea surface temperatures for the relative intensity calculation (see Appendix B)

♦ Output Data

- Initial storm location, motion and intensity of all selected storms
- Initial storm location, motion and intensity PDFs
- Storm motion and intensity change PDFs
- Diagnostic output file

3.1.2.2 The storm track generator (STORMGEN)

STORMGEN generates the stochastic tracks based on the PDFs derived by GENPDF. The initial conditions may either be sampled from the initial storm location, motion and intensity PDFs or taken from observed initial conditions. Both these input data are created by GENPDF.

The model uses a 1-hour time step, which requires interpolation of the 6-hour report changes used in the storm motion change and intensity PDFs. Currently, storm motion is persisted during 6-hour intervals, and the pressure is linearly interpolated.

The basic flow of the model is as follows:

1 If using specified initial conditions, read in initial storm location, date, motion and intensity. If using random initial conditions, read in storm genesis time (see Use Case for Hurricane Genesis, SGT) and sample initial storm location, motion and intensity PDFs. Add a uniform random term equal to the width of the location PDF bin size, so that the storm may form anywhere within the bin.
2 Sample storm parameters Rmax and Beta.
3 Update storm position using current motion

4 If at 6-hour interval, sample new motion and intensity change. Pressure tendency is interpolated to one-hour tendency.
5 Determine if landfall or currently over land. If yes, decay the storm using the decay model described Section 5. Otherwise, update pressure.
6 Check if maximum relative intensity is exceeded, cap if necessary. If pressure is greater than 1011 mb, dissipate storm.
7 Calculate new Rmax, Beta.
8 If storm outside threat area, terminate. Otherwise go to step 3.
9 After storm track is generated, it is trimmed based on the distance criteria described in the Use Case for Zip Code Criterion.

♦ Input Data

- Initial storm location, motion and intensity (if using specified initial conditions)
- Initial storm location, motion and intensity PDFs from GENPDF
- Storm motion and intensity change PDFs from GENPDF
- Hurricane genesis time (output from Use Case for Hurricane Genesis, SGT)
- Zip code locations (used for distance criteria described in Use Case for Zip Code Criterion)
- Land Mask file
- Outflow temperature file (see Appendix B)
- Sea surface temperature file (see Appendix B)

♦ Output Data

- Track positions of stochastic storms in original HURDAT format (Note: small changes are needed as the original format is not capable of handling large number of storms)
- Track positions in special format for use in wind model.
- Landfall data for diagnostic purposes
- Diagnostic output file

3.1.2.3 Appendix A – Wind-Pressure Relation

An empirical wind-pressure relation is used to convert HURDAT wind reports to pressure. The relation is dependent on region.

The relation is
If longitude is > 81.5W and latitude > 20N,

$$P = 1013 - (W/10.627)^{1.7730}$$

Else if latitude < 25 N,

$$P = 1013 - (W/12.016)^{1.8737}$$

Else if latitude < 35N,

$$P = 1013 - (W/14.172)^{2.0929}$$

Else,

$$P = 1013 - (W/16.086)^{2.3079}$$

Where P is central pressure in mb and W is wind speed in kt.

3.1.2.4 Appendix B – Relative Intensity Calculation

The relative intensity calculation is based on Darling (1991). The calculation is as follows:

$$rv = 461$$

$$rh = 0.80$$

$$e = (ts - \mathrm{to})/ts$$

$$es = 6.112 * \exp(17.67(ts - 273.)/(ts - 29.5))$$

$$Pda = 1013 - (rh * es)$$

$$Lv = 2.5 * 10^6 - 2320.(ts - 273)$$

$$a = e * Lv * es/((1 - e) * rv * ts * Pda)$$

$$b = rh * (1 + es * \log(rh)/(Pda * a))$$

Then solve for x in

$$x = \exp(-a(1/x - b))$$

and then finally the relative intensity is given by

$$RI = (1013 - Pmsl + (1 - rh) * es)/((1 - x)(1013 - (rh * es)))$$

3.1.2.5 Data Sources

This calculation requires as input the mean sea level pressure (Pmsl), which in our case is the storm central pressure, the outflow (to) and sea surface temperatures (ts). The outflow temperature is taken to be the monthly mean 100 millibar temperature derived by the Climate Diagnostics Center (CDC) using National Center for Environmental Prediction Center (NCEP) Reanalysis II data. This data is available online at http://www.cdc.noaa.gov/ncep_reanalysis. The sea surface temperature data is monthly mean Reynolds Optimal Interpolation Version 2 (OIv2) data (Reynolds et al., 2002).

3.1.3 Computer Model Design & Implementation

3.1.3.1 Use Case View of Storm Track Model

A. Actors:
There is one actor, scientist.

B. Use Case:

Strom track model is aimed at generating the storm tracks for simulated storms based on data obtained from Use Case II and stochastic algorithms.

C. Use Case Diagram:

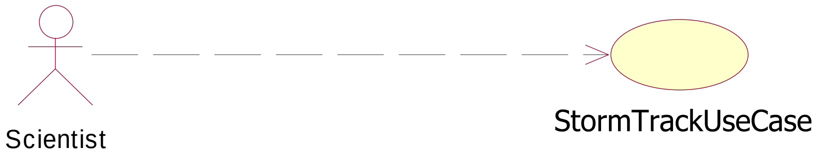
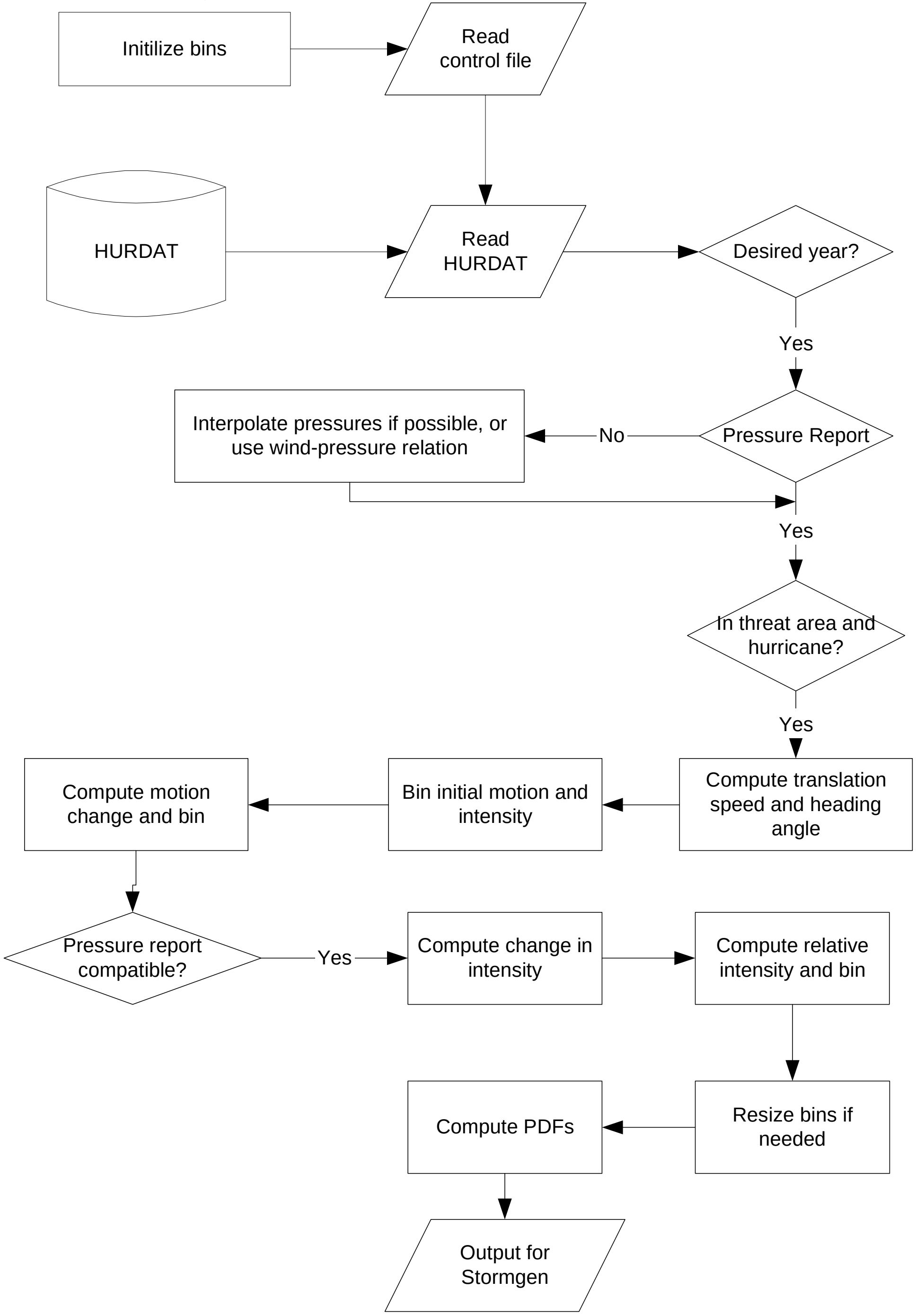


Figure 3.1.1: Use Case Diagram

3.1.3.2 Storm Track Model Implementation

This model is implemented using FORTRAN language in Unix console-based environment. This section includes the overall flow chart of Storm Track Model Implementation.

3.1.3.3 Program Flow Chart of Storm Track


Initilize bins
Read control file
HURDAT
Read HURDAT
Desired year?
Yes
Interpolate pressures if possible, or use wind-pressure relation
No
Pressure Report
Yes
In threat area and hurricane?
Yes
Compute motion change and bin
Bin initial motion and intensity
Compute translation speed and heading angle
Pressure report compatible?
Yes
Compute change in intensity
Compute relative intensity and bin
Compute PDFs
Resize bins if needed
Output for Stormgen

3.1.3.4 Storm Track Output

```
12
storm00004   8/24/   1992  01:00

4  1992  0824  05 00    25.4    79.3   937   19   1.4772400  0
4  1992  0824  06 00    25.4    79.3   937   19   1.4772400  0
4  1992  0824  07 00    25.4    79.6   939   18   1.4727061  0
4  1992  0824  08 00    25.4    80.0   942   18   1.4681721  0
4  1992  0824  09 00    25.5    80.4   945   18   1.4636379  0
4  1992  0824  09 05    25.5    80.3   922   19   1.5048399  1
4  1992  0824  10 00    25.5    80.8   948   18   1.4591039  3
4  1992  0824  11 00    25.6    81.2   951   17   1.4545699  3
4  1992  0824  12 00    25.6    81.2   951   17   1.4545699  2
4  1992  0824  13 00    25.6    81.5   950   18   1.4541880  0
4  1992  0824  14 00    25.6    81.9   949   19   1.4538059  0
4  1992  0824  15 00    25.7    82.3   948   20   1.4534241  0
```

3.1.4 References

Reynolds, R.W., N.A. Rayner, T.M. Smith, D.C. Stokes, and W. Wang, 2002: An Improved In Situ and Satellite SST Analysis for Climate. J. Climate, 15, 1609-1625.

Section 3.2

Wind Field Model
Use Case IV

3.2.1 General Description Of Wind Field Model

Wind Field model aims at estimating the terrain wind speed with respect to the actual terrain (based on land use – land cover). To be precise it calculates wind speed time series for each of the zip code affected by the storm. The time series includes the date, landfall time of the storm. It also includes the zonal wind speed (m/s), surface wind speed (m/s), and the wind direction in degrees at regular time intervals.

3.2.2 General Requirements of Wind Field Model

Name: Wind Speed Model

Description: The user enters *Category, Year, Date, Time, Latitude, Longitude, Centre pressure, Rmax, Holland B and lsflg for each of hourly fixes of the storm*. The system generates the following:

1. Landfall or bypassing location (i.e. longitude/latitude) of storm
2. Maximum open terrain (OT) wind speed/time/direction anywhere in the storm
3. Maximum Marine (MA) Exposure at landfall or bypassing position
4. Maximum wind speed/time/direction at each zip code affected by the storm

1. The end user enters the input file as the following format:

<number of fixes>
<storm Number><m/d/ yyyy > <hh: mm>
<storm category><year><mmdd><hh><minute><latitude><longitude><center pressure><R_{max}><Holand> <lsflg>

Example:

```
13
storm00001   8/24/ 1992  05:00
4  1992  0824  05 00    25.4    79.3   937   19   1.4772400  0
4  1992  0824  06 00    25.4    79.3   937   19   1.4772400  0
```

2. Based on the input data from step 1, the model generates the output as the following:

Given below a partial output file showing the wind field for some of the zip codes affected by the storm Andrew while the original file contains wind fields for all of the zip codes in the threat area, which were affected by this storm.

A. Land falling storms

```
ANDREW  8/24/92  5:00 UTC
landfall: longitude:     -80.3000 deg   latitude:      25.5000 deg
ter      day    hour     min      zonal     meridional    total m/s dir(deg)
 MA       1       9       5     -52.1610     -16.7574      54.7866      72
 OT       1       9       5     -41.1068     -26.5020      48.9094      57

zipcode:          31  longitude:     -80.1000 deg   latitude:      25.5900 deg
ter      day    hour     min      zonal     meridional    total m/s dir(deg)
 OT       1       9       0     -42.9686      13.8472      45.1448     107

zipcode:          32  longitude:     -80.2700 deg   latitude:      25.3400 deg
ter      day    hour     min      zonal     meridional    total m/s dir(deg)
 OT       1       9       0      30.4650      21.1683      37.0973     235
```

B. Bypassing Storms

3. For a storm that does NOT make landfall but passes close enough to a zip code to do damage:

    a. For each affected zip code, find the peak OT wind and the date/time of this peak wind speed

    b. Find the track positions closest to the time of the peak zip code wind speed and choose the fix with the lowest central pressure. Use the fix information to compute the maximum marine (MA) exposure wind speed in the storm.

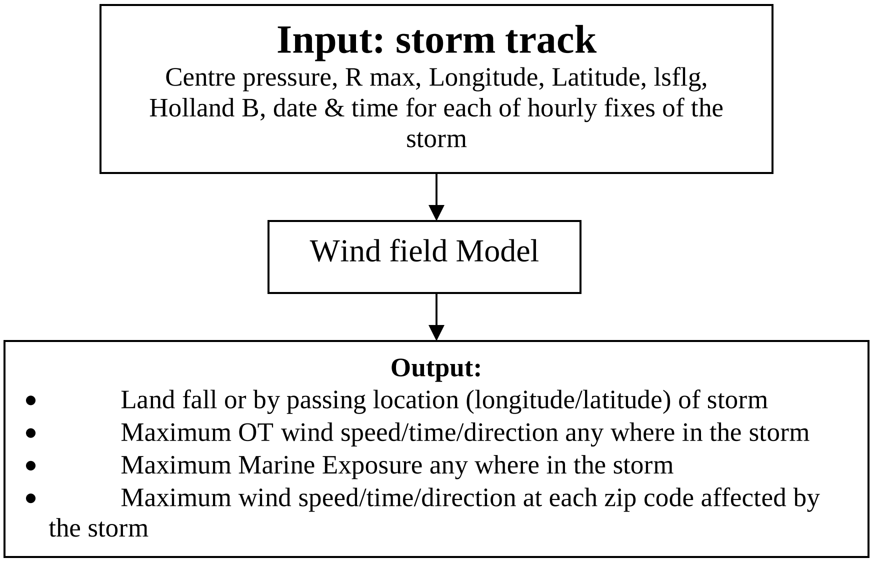
    c. Label this storm as "By-Passing" in the header and include the MA wind speed and date/time.

    d. Include the zip code information identical to that done with the storms that make landfall.

This method will not include storms that bypass and then make landfall further along the track, or landfall and then bypass further along the track.

```
DAVID  9/03/79  06:00 UTC
bypass:  longitude:     -80.5000 deg   latitude:      28.8000 deg
ter      day    hour     min      zonal     meridional    total m/s dir(deg)
 MA       2       4       0     -23.6102      34.6558      41.9340     145
 OT       2       4       0     -26.7334      23.0222      35.2802     130

zipcode:          31  longitude:     -80.1000 deg   latitude:      25.5900 deg
ter      day    hour     min      zonal     meridional    total m/s dir(deg)
 OT       1       8       0      8.13654     -10.2191      13.0626     321

zipcode:          32  longitude:     -80.2700 deg   latitude:      25.3400 deg
ter      day    hour     min      zonal     meridional    total m/s dir(deg)
 OT       1       8       0      8.26174     -4.41157      9.36580     298

zipcode:          33  longitude:     -80.1310 deg   latitude:      25.7100 deg
ter      day    hour     min      zonal     meridional    total m/s dir(deg)
 OT       1       8      15      7.02892     -10.4825      12.6210     326

zipcode:          34  longitude:     -80.3200 deg   latitude:      25.6100 deg
ter      day    hour     min      zonal     meridional    total m/s dir(deg)
 OT       1       8      15      6.55432     -6.86560      9.49187     316
```

3.2.3 Technical Description of Wind Field Model



Input: storm track

Centre pressure, R max, Longitude, Latitude, lsflg, Holland B, date & time for each of hourly fixes of the storm

Wind field Model

Output:

- Land fall or by passing location (longitude/latitude) of storm
- Maximum OT wind speed/time/direction any where in the storm
- Maximum Marine Exposure any where in the storm
- Maximum wind speed/time/direction at each zip code affected by the storm

Figure 3.2.1 Input Output of Wind Field Model

Once a simulated hurricane moves to within a distance threshold of Florida communities, the wind field model is turned on. Gradient balance represents a circular flow caused by the balance of forces on the flow whereby the inward directed pressure gradient force is balanced by outward Coriolis and centripetal accelerations. The coordinate system translates with the hurricane vortex moving at velocity **c**. The vortex translation is assumed to equal the geostrophic flow associated with the large-scale pressure gradient. In cylindrical coordinates that translate with the moving vortex, equations for a slab hurricane boundary layer under a prescribed pressure gradient are:

$$\frac{u\partial u}{\partial r}-\frac{v^2}{r}-fv+\frac{v}{r}\frac{\partial u}{\partial \phi}+\frac{\partial p}{\partial r}-K\left(\nabla^2 u-\frac{u}{r^2}-\frac{2}{r^2}\frac{\partial u}{\partial \phi}\right)+F(\vec{c},u)=0=\frac{\partial u}{\partial t} \qquad (1)$$

$$u\left(\frac{\partial v}{\partial r}+\frac{v}{r}\right)+fu+\frac{v}{r}\frac{\partial v}{\partial \phi}-K\left(\nabla^2 v-\frac{v}{r^2}+\frac{2}{r^2}\frac{\partial u}{\partial \phi}\right)+F(\vec{c},v)=0=\frac{\partial v}{\partial t} \qquad (2)$$

where u and v are the respective radial and tangential wind components relative to the moving storm, p is the sea-level pressure which varies with radius (r), f is the Coriolis parameter which varies with latitude, ϕ is the azimuthal coordinate, K is the eddy diffusion coefficient, and F(c,u), F(c,v) are frictional drag terms. All terms are assumed to be representative of means through the boundary layer. The motion of the vortex is determined by the modeled storm track.

The hurricane windfield model is based on a fully two dimensional, time-independent, scaled version of the tangential and radial momentum equations (1 and 2) for the mean boundary layer wind components. The model makes use of a polar coordinate representation grid (Fig. 1) centered on the moving cyclone. The nested circles are separated from their inscribed and circumscribed neighbors by a radial separation of 0.1 in units of Rmax (Radius of maximum winds); the azimuthal interval is 10 degrees.

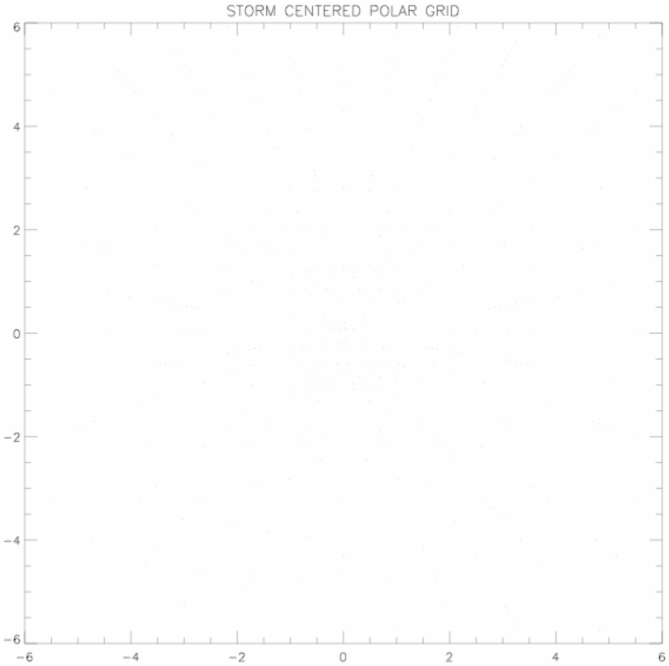


Figure 3.2.2: Polar coordinate system for solving equations of motion.

Implementation proceeds according to the following steps: First, based on the input parameters, namely the radius of maximum winds, the central pressure and the Holland B parameter, radial profiles of the radial and tangential winds are calculated based on a stationary cyclone over open water to provide an "envelope" with which to set the size of the cyclone vortex. The wind field produced by these profiles is radially symmetric.

Azimuthal variation is introduced thru the use of two form factors. The form factors multiply the radial and tangential profiles described above and provide a "factorized" ansatz for both the radial and tangential storm–relative wind components. Each form factor contains three constant coefficients which are variationally determined in such a way that the ansatz constructed satisfies (as far as its numerical degrees of freedom permit) the scaled momentum equations for the storm-relative polar wind components. The azimuthal variable (ϕ) has its usual mathematical meaning such that ϕ increases from left to right with the rectangular X axis aligned (ϕ =180, 0) and the Y axis aligned (ϕ =270, 90) with Y increasing in the direction of storm translation.

The translational motion of the storm is vectorially added to the storm-relative wind components in order to obtain the earth-relative wind field. The translational motion of the storm is incorporated in the surface friction terms in the momentum equations which depend on the ϕ and are specific for the direction of storm translation which is aligned with the Y axis. The wind field grid is then rotated so that the computational y axis coincides with the actual direction of motion of the cyclone center. The wind field thus far constructed (Fig. 2) usually shows the location of peak winds to be to the right or forward edge of the right-rear quadrant of the cyclone.

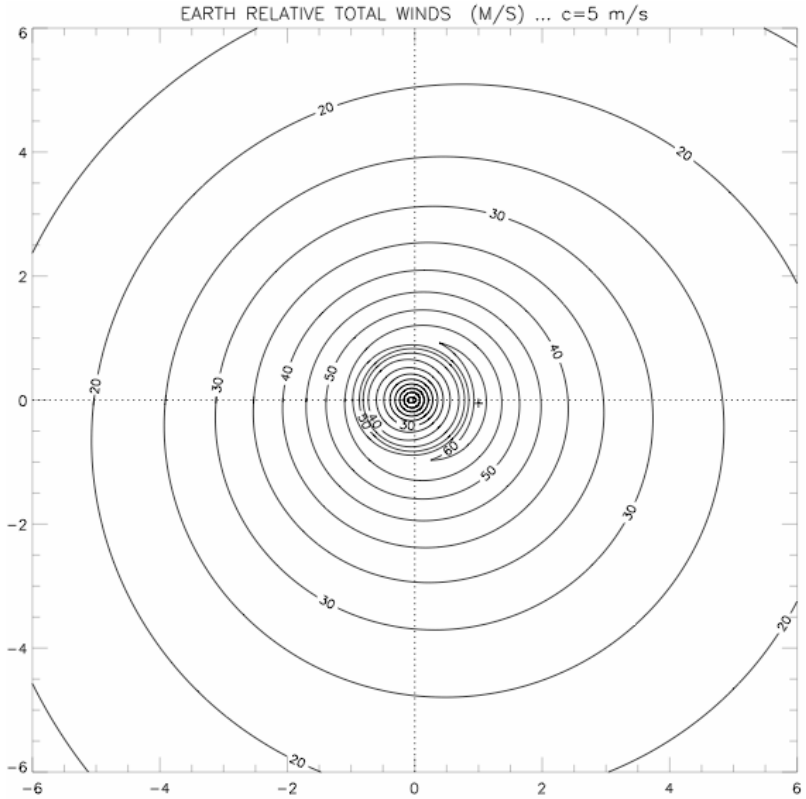


Figure 3.2.3. Horizontal distribution of mean boundary layer wind speed (m s^{-1}) relative to the earth for a Hurricane moving northward (top of page) at 5 m s^{-1}. Horizontal coordinates are scaled by the radius of maximum wind.

3.2.3.1 Wind Model Parameters

Following are the input parameters to the wind field model

3.2.3.1.1 Delta P: Intensity parameter

This is the difference between the central minimum sea level pressure and an outer peripheral pressure. (assumed to be 1012 hPa). Intensity change is modeled by using the observed geographic probability distribution of six-hour changes of central pressure as related to the relative intensity. Potential intensity takes into account the concept of the hurricane as a heat engine constrained by the input (sea surface) and outflow (upper troposphere) temperatures. Intensity change is limited so as to not exceed the maximum observed change for a particular geographic region. When a storm center crosses the coastline (landfall) the intensity change follows a pressure decay model (discussed below). If the storm moves back over the sea, the former intensity change model is reinstated.

3.2.3.1.2 R max : Radius of Maximum Wind

The radius of maximum wind is determined from a distribution of values as a function of po and latitude, where po is the central minimum sea level pressure. A log normal distribution is assumed for R with a mean value determined as a function of Δp and Latitude. The relationship between R and Δp and latitude shows much scatter but a generalized linear model for the natural log of R (r2 = 0.212) provides a useful estimation:

$$\Delta p = 1012 - p_o$$

$$\ln R_{\max} = 2.0633 + 0.0182\Delta p - 0.00019008\Delta p^2 + 0.0007336 Lat^2 + \varepsilon \qquad (3)$$

Where ε is a normal random variable with a mean of zero and a variance of 0.169. Equation (3) describes the mean of the log normal distribution of R in nautical miles. When a simulated storm is close enough to land to become a threat, an R-value is randomly chosen given the Δp and Latitude. R is computed at each time step but the random error term is computed only once for each landfall.

Rmax in nautical miles is calculated as follows:

$$R_{\max} = e^{\ln(R_{\max})}$$

3.2.3.1.3 Pressure Profile & Holland B

The symmetric pressure field p(r) is specified as:

$$p(r) = p_o + \Delta p e^{\left(\frac{R_{\max}}{r}\right)^B} \qquad (4)$$

where po is the central minimum sea level pressure, B is the Holland pressure profile shape parameter, R is the radius of maximum wind speed (in nautical miles), and Δp is the pressure deficit defined earlier. The central pressure is modeled according to the intensity modeling in concert with the storm track.

The resulting expression for B is:

$$B = 1.38 + 0.00184\Delta p - 0.00309 R_{\max} + \varepsilon \qquad (5)$$

Where ε is a random term from a zero mean normal distribution with a standard deviation of 0.05.

3.2.3.1.4 Land See Flag: lsfg

Gives the position of the wind at the storm fix.

0 – Over Ocean
1 – Land Fall
2 – Sea Fall
3 – Over Land
4 – Closest Approach of bypassing Storm

3.2.3.2 Definitions and Equations of the wind model

R = Radius of maximum surface wind speed, specified

ct = storm translation speed, specified

cdir = storm translation direction compass heading , specified

Δp = Central pressure deficit, specified

$p(r) = p_o + \Delta p e^{\left(\frac{R_{\max}}{r}\right)^B}$ = sea level pressure

$B = 1.38 + 0.00184\Delta p - 0.00309R$ = Holland profile parameter

ϕ = Azimuthal coordinate, measured counterclockwise from east

$s = \frac{r}{R} =$ normalized radial coordinate

$v_g(s)$ = Gradient wind: $\frac{{v_g}^2}{s} + Rfv_g = \frac{1}{\rho}\frac{\partial p}{\partial s}$

$f = 2\Omega \sin\vartheta$ =Coriolis parameter

ϑ = latitude of storm center

$v_0(s)$ = normalized gradient wind (symmetric) = $\frac{v_g(s)}{V_{g\max}}$ where $V_{g\max}$ is the maximum gradient wind in the radial profile

$\bar{f} = \frac{Rf}{V_{g\max}}$ = Normalized Coriolis parameter

$v(s,\phi) = \frac{v}{v_g} =$ Normalized storm-relative tangential wind component

$u(s,\phi) = \frac{u}{v_g} =$ Normalized storm – relative radial wind component

$\alpha =$ Friction coefficient $= \frac{RC_d}{h}$

h= mean boundary layer height

C_d = Drag Coefficient

$c = \dfrac{c_t}{V_{g\max}}$ = normalized translation speed

$$g(s) = 2v_o(s)s^{-1} + \bar{f} \tag{A1}$$

$$d(s) = \dot{v}_0 + v_0 s^{-1} + \bar{f} \tag{A2}$$

where a “dot” represents a derivative with respect to s, $g(s)$ and $d(s)$ depend only on V_0 and $\bar{f}$

$\sigma(s,\phi) = v(s,\phi) - v_o(s) =$ Normalized departure from gradient balance

♦ Scaling of the governing equations prior to implementation.

Substituting the terms from the above definitions and changing the radial coordinate from r to s, the steady-state form of the governing equations (1) and (2) become:

$$u\partial_s u + s^{-1}(v_o + \sigma)\partial_\phi u - \sigma(g + s^{-1}\sigma) + \alpha(u + c\sin\phi)(w - c) = 0 \tag{A3}$$

$$u\partial_s \sigma + s^{-1}(v_o + \sigma)\partial_\phi \sigma + u(d + s^{-1}\sigma) + \alpha(v_o + \sigma + c\cos\phi)(w - c) = 0 \tag{A4}$$

$$w = \sqrt{(u + c\sin\phi)^2 + (v_o + \sigma + c\cos\phi)^2} \tag{A5}$$

where w is the total normalized earth-relative wind.

In the event that c vanishes, so that the cyclone is stationary, these equations reduce to the ordinary differential equations:

$$u\dot{u} - \sigma(g + s^{-1}\sigma) + \alpha u w = 0 \tag{A6}$$

$$u(\dot{\sigma} + s^{-1}\sigma + d) + \alpha(v_0 + \sigma)w = 0 \tag{A7}$$

$$v = v_0 + \sigma$$

$$w^* = 0.8\sqrt{u^2 + v^2}$$

$$w = w^*(1 + \beta w^*) \qquad \beta = 0.132653 \tag{A8}$$

for the radial profiles $u(s)$ and $\sigma(s)$, Here, “.” indicates differentiation with respect to s.

Equations A3 and A4 supplemented by A5, constitute two, coupled, time independent partial differential equations for the storm relative radial velocity u and the storm relative departure from gradient balance σ. The storm relative tangential wind is then given by $v = v_g + \sigma$.

Unfortunately, the direct numerical solution of A3 and A4 is time consuming even though the equations are time-independent because the non-linear coupling of the terms necessitates an iterative numerical approach.

However, equations A6 and A7, can readily be numerically integrated to furnish a completely symmetric windfield fully described by the radial profiles u(s) and $v(s) = v_g(s) + \sigma(s)$.

The functions u(s) and $\sigma(s)$ so obtained can serve as radial profiles for the construction of basis functions for a more realistic attack on A3 and A4.

Namely, we put forth the ansatz:

$$u(s,\phi) = ffu(\phi)u(s) \qquad \text{(A9)}$$

$$\sigma(s,\phi) = ff\sigma(\phi)\sigma(s) \qquad \text{(A10)}$$

where the azimuthal dependence is introduced through the form factors:

$$ffu(\phi) = a_0 + a_1 \cos\phi + a_2 \sin\phi \qquad \text{(A11)}$$

$$ff\sigma(\phi) = b_0 + b_1 \cos\phi + b_2 \sin\phi \qquad \text{(A12)}$$

Now the six coefficients a0, a1, a2 and b0, b1, b2 can be variationally determined by substituting A9 and A10 into the left hand sides of A3 and A4, supplemented by A5 to form the "residuals" RA3 and RA4. We then form the functional:

$$|RA3| = \sum |(A3)| = \sum | u\partial_s u + s^{-1}(v_o + \sigma)\partial_\phi u - \sigma(g + s^{-1}\sigma) + \alpha(u + c\sin\phi)(w - c) |$$

$$|RA4| = \sum |(A4)|$$

$$J(a,b) = \frac{\sum |RA3| + |RA4|}{NGRID} \qquad \text{(A13)}$$

Where the sum is taken over every spatial point for which the profiles and trigonometric functions are known (polar grid) and NGRID is the total number of such grid points.

J then depends solely on the unknown coefficients a0,a1,a2 and b0,b1,b2. These coefficients are chosen to minimize J and so furnish us with an approximate solution for u(s,ϕ) and σ(s,ϕ), from which we form the storm relative radial and tangential wind components ur and vt, namely:

ur(s,ϕ)= u(s,ϕ) and vt(s,ϕ)= vg(s)+ σ(s,ϕ) (A14)

By adding the translational velocity c (in polar coordinates) to ur and vt we obtain the earth-relative components of the windfield uer and ver :

$$uer(s,\phi)=ur(s,\phi)+c\sin\phi \qquad (A15)$$

$$ver(s,\phi)=vt(s,\phi)+c\cos\phi \qquad (A16)$$

where c is the normalized translation speed c=c_t/Vgmax.

Finally, since A3, A4 and A5 refer to a cyclone moving along the y-axis, the entire generated windfield grid must be rotated so that the y-axis of the calculation coincides with the actual compass direction of motion of the translating cyclone.

3.2.4 Computer Model Design

3.2.4.1 Use Case View of Wind Speed Model

A. Actors:

There is one actor, scientist.

C. Use Case:

Wind Speed model is used to estimate terrain wind speed.

C. Use Case Diagram:

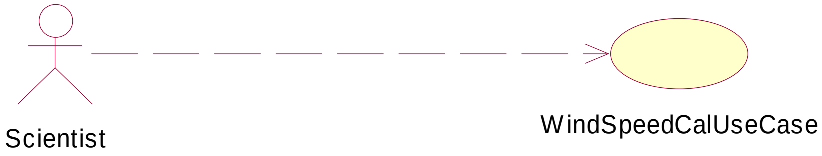


Figure 3.2.4: Use Case Diagram

3.2.5 Implementation of Wind Field Model

This model is implemented using Interactive Data Language (IDL) language in Unix console-based environment. This section includes appropriate diagrams and the overall flow chart of Wind Speed Model Implementation.

3.2.5.1 Program Flow Chart of Wind Speed Model

Wind field model has been implemented using Interactive Data Language (IDL). To be precise it calculates wind speed time series for each of the zip code affected by the storm. The time series includes the date, landfall time of the storm. It also includes the zonal wind speed (m/s), surface wind speed (m/s), and the wind direction in degrees at regular time intervals.

General structure of the main IDL modules is given below.

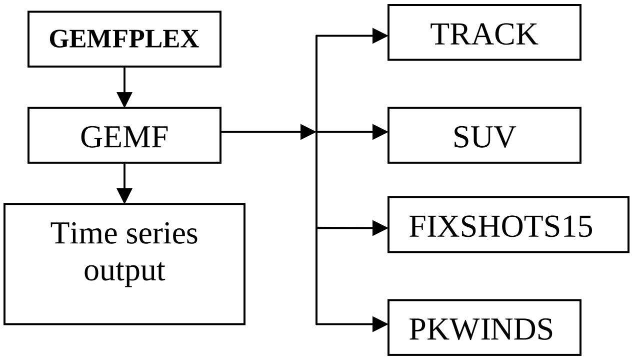


Figure 3.2.5. General structure or flow of the IDL

- GEMFPLEX is analogous to a main or the entrant procedure in C/C++. It reads g_trackfile and separates it into individual track files for processing.
- GEMF processes each single track.
- Each of the procedures TRACK, SUV, FIXSHOTS15 and PKWINDS call other procedures.
- TRACK reads the necessary input parameters from the storm track and thins out the fixes based on the storm category and saves track related quantities for future use.
- SUV generates radial profiles from stationary cyclone equations.
- FIXSHOTS15 generate field snapshots with azimuthal variation for each fix.
- PKWINDS is responsible for picking the maximum wind for each zip code. If the storm happens to encompass or run through the entire state of the FL then this step would end up consuming a lot of resources.

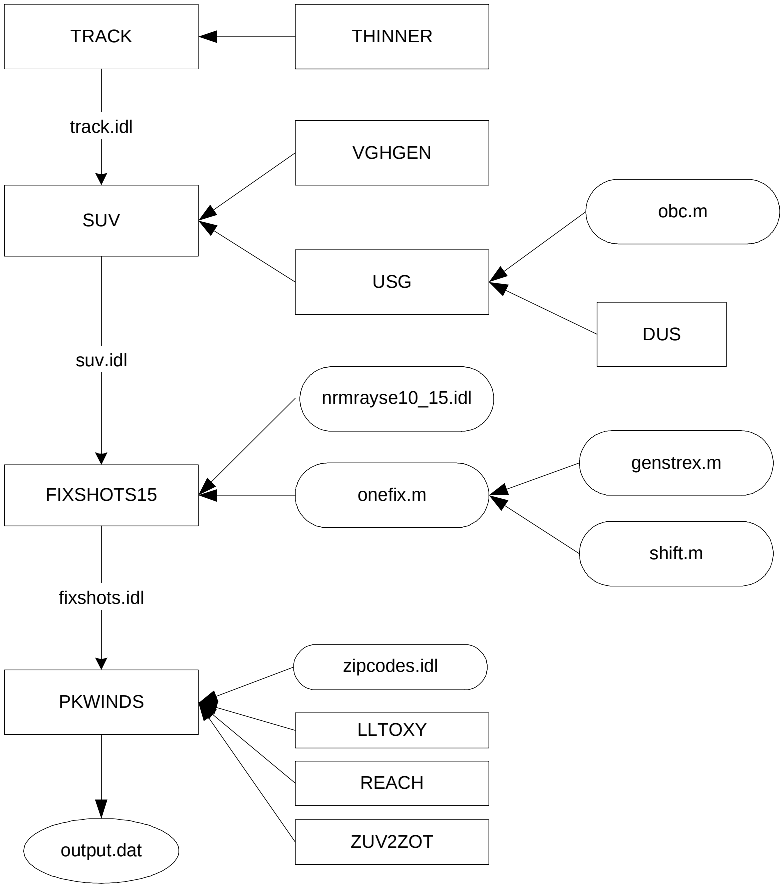
Note: All the equations referenced in the following are from Wind field Model Technical description. Please see the document for the detailed information.

Note

The files names without any extension in them are idl procedure files (.pro files)

Wind model code (IDL code) flowchart

Coded by: Dr George A. Soukup | Frozen Model



i. TRACK.PRO

1. Reads in the trackfile to arrays
   ctg = storm category, zhour = fix hour, zmin=fix min, nlat=latitude, elon=longitude, cpr=centre pressure, rmx=Rmax, hdb=Holland B, lsflg=land sea flag.
2. Mark the fix of lowest central pressure unless it coincides with landfall. (lsflg is set to 4)
3. Thins out the storm fixes based on the adjusted fix frequency. THINNER.PRO is used to accomplish this task.
4. Calculate the time in minutes for each fix from the start of the storm rack. (ktime)

5. Samples the data at regular (1 hrs) intervals prior to the smoothing using cubic spline interpolation.
6. Calculate fbarx=Rmax . f   where f= 0.14544* sin(nlat)
7. Sub-samples the smoothed input data to recover the original resolution (unequal intervals based on the storm category).
8. For the landfall fix get the landfall location and time.
9. Calculates the storm translation speed in m/s (spdmsx) and bearing (bearx) based on the fix data.
10. Smoothens translation speed and bearing (clock wise angle from north) on hourly grid.
11. Evaluates smoothed translation speed (spdms) and bearing (bear) at fix times using Cubic spline interpolation.
12. Evaluates smoothed track positions (elonk,nlatk) and Rmax (rmwk) minute by minute.
13. Finally, saves track related quantities for use by other procedures as trackc.idl.

    bear=Bearing at each fix,
    cpr=center pressure at each fix,
    day=day of each fix,
    elon=longitude of each fix,
    elonk=longitude of storm at each minute,
    fbr=f bar at each fix,
    hdb=Holland B at each fix,
    ktime=array from 0 to last minute of storm track (step=1),
    lsflg=land sea flag of each fix,
    minz=min of each fix,
    nlat=latitude of each fix,
    nlatk=latitude of storm at each minute,
    pdf=delta p of each fix,
    rmwk=R max at each minute,
    rmx=Rmax of each fix,
    spdms=translation speed of the storm at each fix

ii. THINNER.PRO

This module reduces the number of fixes depending upon the storm intensity.

1. Locate landfalls, sea falls and minimum pressure point.
2. Classify fixes by category
3. Thin out the fixes as below;
    i. Category 0:Select fixes in 8 hour separations
    ii. Category 1:Select fixes in 6 hour separations
    iii. Category 2:Select fixes in 4 hour separations
    iv. Category 3:Select fixes in 3 hour separations
    v. Category 4:Select fixes in 2 hour separations
    vi. Category 5:Select fixes in 1 hour separations
4. Merge retained fixes and additional landfall and seafall fixes.

iii. SUV.PRO

This module computes the radial and tangential wind profiles u and v, as well as the gradient wind profile and the functions g and d and their second derivatives.

1. Restores variables saved in track.idl
2. If Holland b is zero calculate it using $B = 1.38 + 0.00184\Delta p - 0.00309R$ (this step is rarely done)
3. Calculates radial(ur) and tangential(vt) wind profiles for each storm fix. Wind profiles are calculated at 201 points starting from 0(the storm center) to 20 in steps of 0.1 (in units of RMW)

4. Calculates the gradient wind profile for each fix using VGHGEN.PRO
5. Calculate g and d using equation (A1) and (A2).
6. Calculate initial estimate for alpha (alfi) neglecting first derivatives of u and σ.
7. Alpha is iteratively estimated until correct alpha is obtained (USG.PRO is used). To check the correctness of the estimate the boundary condition following boundary condition is used.

   Peak wind should be at s=1. (i.e. if iw is 10 answer of alpha is correct)
8. Momentum equations are used to furnish tangential and radial profiles. USG.PRO
9. Collects radial and tangential profiles into a structure.
10. Saves the variables for use by the other procedures as suv.idl.

    uvstr= holds the wind profiles calculated for each fix.

iv. VGHGEN.PRO

This module calculates the gradient wind profile and its second derivative.

v. USG.PRO

This module computes the radial and tangential wind profiles for a stationary storm with surface friction for exactly one fix. Wind profiles are calculated from two directions, inward and outward from center. Then the results are combined to get the complete profile.

1. Form the inward boundary value at s=20 using obc.m
2. Numerically integrate momentum equations for stationary storm profiles LSODE.
3. Match solutions across the shock and obtain uz and sgz.

   For 0≤ s≤1; uz = uout
   sgz = sgout
   for 1<s≤20; uz = uinw
   sgz = sginw
4. Sub-grid smoothing process simulates turbulent diffusion.
5. Sub sample to recover original resolution.

vi. USNOADV.M

Ignore radial advection and algebraically solve for u and sigma (equation (A6) (A7)) starting the numerical integration outwards from the origin.

vii. USADV.M

This module iteratively improves the results of USNOADV.M by including radial advection terms (first derivatives) evaluated from the previous iteration.

viii. OBC.M

This module computes the outer boundary values for u and sg to start the inward numerical integration of u and sg using LSODE. (Refer to IDL manual for LSODE). Procedure DUS is used to calculate the derivatives of u and σ from (A6, A7, A8).

ix. DUS.PRO

Calculates radial derivatives from momentum equations. (A6)(A7)(A8)

$vz = v_0 + sg$ $dz=d+sg/s$ $gz=g+sg/s$

du =first derivative of u

dsg=first derivative of sigma

x. FIXSHOTS15.PRO

This module calculates the field snapshots and their second time derivatives at each retained fix time on a polar grid extending outward from the storm center to 15 RMW in steps of 0.1RMW and 10^0 angle. (This would give a matrix of 151 x 36 points. But three extra lines are added for the convenience of future calculations making the matrix dimension 151 x 39)

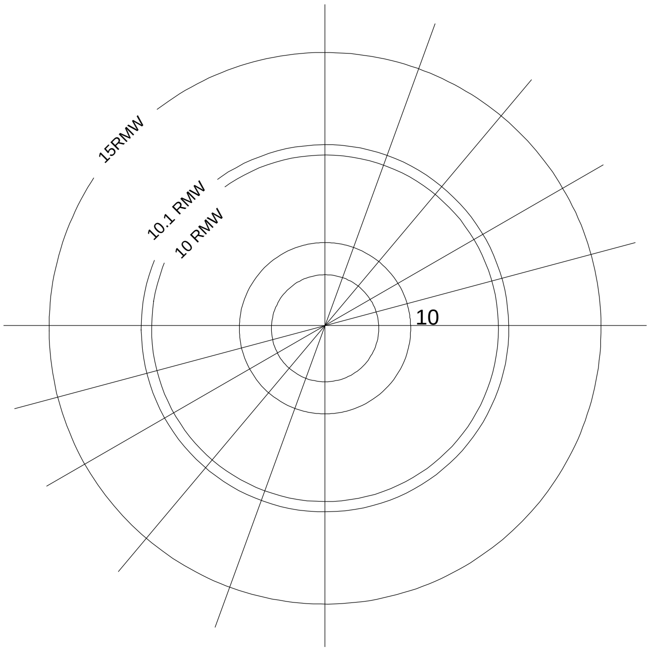


Figure 3.2.6: Polar grid

1. Restore suv.idl
2. Restore nrmrayse10_15.idl, which contain some trigonometric values corresponding to each of the grid point.
3. For each retained fix, construct the polar grid of earth relative marine surface winds. (onefix.m)
4. onefix.m gives the polar grid of earth relative marine surface winds for exactly one fix.
5. ‘reform’ converts this 151x39 matrix in to a raw matrix of 1x 5889.
6. zsnapi is a complex matrix which contains the snap shots of the retained fixes. [#retained fixes X 5889]
7. usnap contains the earth relative zonal winds and vsnap contains the earth relative meridional winds.
8. Compute second time derivative of fields for time interpolation. Time interpolation is done in order to find the details of the storm every minute.

xi. ONEFIX.M

This module constructs zonal and meridional windfield components for exactly one fix.

1. Load single fix profiles and corresponding data.
2. Calculate purely radial (no azimuthal dependence) functions on a polar grid. (GENSTREX.M)
3. Then introduce azimuthal dependencies and calculate storm relative-wind field. Equation (A11) through (A13)
4. Calculate the form factors (a_0, a_1, a_2, b_0, b_1, b_2 of equation A11 & A12)
   cfu=coefficients of u (a_0, a_1, a_2) cfsg=Coefficients of Sigma (b_0, b_1, b_2)
   initial estimate=1,0,0 initial estimate=1,0,0
5. Keep changing the estimate to minimize J (equation A13) using AMOEBA, MNRDU and MNRDSG. (AMOEBA is a built in function in IDL)
   MNRDU= Calculate a’s keeping b’s fixed.
   MNRDSG=Calculate b’s keeping a’s fixed.
6. Form the earth relative wind field assuming that the storm moves northwards.
7. Calculate u, σ , uer and ver using equations (A14) through (A16)
8. Storm rotates counter clockwise. Once the northward storm translation speed is induced storm center tend to move towards west. Shift.m takes this into account and shifts the polar origin to the storm center.
9. Advance phase
10. Orient the wind field to track direction. - Initially we assumed that the storm is moving northwards. In this step some interpolation is required since the actual direction of the storm unlikely to lie exactly on a radial of the grid.
11. Convert the radial and tangential wind fields to zonal and meridional components. vystre=holds the meridional component of the wind at each grid point.
    uxstre=holds the zonal component of the wind at each grid point.
    zxystre=Complex array containing the zonal and meridional wind components at each grid point.

xiii. GENSTREX.M

This module places the profile functions and the auxiliary functions g and d (which we calculated earlier) on the polar grid yielding fields with no azimuthal dependence for exactly one fix. sstre=array containing radial distance to each of the grid point from the centre.

xiv. SHIFT.M

This module simply shifts the polar coordinate system, so that the origin of the coordinate system lies on the center of the storm. (Center of the storm is the point where wind speed is zero.)

xv. PKWINDS.PRO

This module produces an output file, which lists the peak marine and open terrain wind components experienced at each zip code for the current storm. If the storm makes landfall, then the peak marine and open terrain winds are listed at the tie and the site of the landfall. If the storm only bypasses the state then the peak marine and open terrain winds are listed for the fix exhibiting the lowest central pressure.

1. restore zipcodes.idl. This contains the longitude and latitude of all zip codes.
2. restore fixshots.idl. (We generated this in previous step)
3. Initialization of other variables.
   elonk=east longitude of the track every minute
   nlatk=north latitude of the track every minute
   kmax=maximum time(life time) of the storm in Minutes. Since the storm is moving it will affect one zip code for a variable time. But we initialize zuvzip for the worst case.
   nzip=Number of zip codes
   werzipx=holds the maximum wind per each zip
4. Calculate all time series. (time k is incremented in steps of ‘kinc’ from zero to kmax)
   elc=longitude of the storm center at each time step.
   nlc=latitude of the storm center at each time step.
   rmw=radius of maximum wind at each time step.
5. Determine which zip codes will be affected by the storm.
   At time k the storm can affect several zip codes in its vicinity and the affected area depends on Rmax. MAP_2POINTS is used calculate the distance from the center of the storm to each of the zip codes. (This is done at each time step). Then REACH is used to calculate the reach of the storm at that particular time step. Storm ‘reach’ is calculated in terms of RMW. If the calculated ‘reach’ is less than 12.5 that calculated value is taken as the reach. Other wise 12.5 is considered as the storm reach.
6. If at least one of the zip codes is affected by the storm; generate relevant portion of gridded field for current time k.
   unow=value of u at this time at each grid point.
   vnow=value of v at this time at each grid point.
7. Evaluate marine windfield components at admissible zip code centroids. First use LLTOXY, latitude & longitude information of the storm center and zip code centroid to calculate the (x,y) distance between storm center and the zip code centroid. Then using interpolation calculates the marine wind speed at the zip code centroid.
8. Use ZMAR2ZOT to convert above calculated marine windspeeds into Open Terrain windspeeds.
9. After the construction of the time series record maximum total OT windspeed at each zip code.
10. Obtain marine and OT peak winds at landfall or lowest pressure for bypassing storms. At the same time record the time and location of landfall or lowest pressure fix.
11. Write the output file if at least one zip code is affected by the storm.

xvi. REACH.M

This function determines the influence radius.

Influence radius = 12.3246 – 0.162*rmw

If the calculated value is less than 4, then set it to 4

xvii. LLTOXY. PRO

This module converts east longitude and north latitude into zonal distance (xmerc) and meridional distance (ymerc) in meters from the cyclone center (elo,gglo)

ymerc= mercator y coordinates from latitudes

xmerc= mercator x coordinates from longitudes

xviii. ZMAR2ZOT.PRO

This module converts marine wind speeds (m/s) into Open Terrain windspeeds (m/s).

xix. GEMF.M

This module is used to set the time step and call the executable.

1. set the time step for storm series calculations and load gemplex.exe

kinc=15 : time step is set to 15 minutes

flcnt= 0: start the output file numbering from 1. (output1.dat, output2.dat…)

References

1. Vickery, P. J., and L. A. Twisdale, 1995: Wind field and filling models for hurricane wind speed predictions, Journal of Structural Engineering, 121, 1700-1709.

2. Ho, F. P., J. C. Su, K. L. Hanevich, R. J. Smith, and F. P. Richards, 1987: Hurricane climatology for the Atlantic and Gulf coasts of the United States. NOAA Tech Memo NWS 38, NWS Silver Spring, MD.

3. Kaplan, J. and M. DeMaria, 1995: A simple empirical model for predicting the decay of tropical cyclone winds after landfall. J. App. Meteor., 34,

4. Ooyama, K. V., 1969: Numerical simulation of the life cycle of tropical cyclones. J. Atmos. Sci., 26, 3-40.

5. Shapiro, L. 1983: The asymmetric boundary layer flow under a translating hurricane. J. Atmos. Sci., 40, 1984-1998.

6. Thompson, E. F., and V. J. Cardone, 1996: Practical modeling of hurricane surface wind fields, Journal of Waterways, Port, Coastal, and Ocean Engineering Division, ASCE, 122, 195-205.

7. Vickery, P. J., P. F. Skerjl, A. C. Steckley, and L. A. Twisdale, 2000a: A hurricane wind field model for use in simulations. Journal of Structural Engineering, 126, 1203-1222.

8. Vickery, P. J., P. F. Skerjl, , and L. A. Twisdale, 2000b: Simulation of hurricane risk in the United States using an empirical storm track modeling technique, Journal of Structural Engineering., 126, 1222-1237.

9. Kurihara, Y. M., M. A. Bender, R. E. Tuleya, and R. J. Ross, 1995: Improvements in the GFDL hurricane prediction system. Mon. Wea. Rev., 123, 2791-2801.

10. Holland, G. J., 1980: An analytic model of the wind and pressure profiles in hurricanes, Mon. Wea. Rev., 108, 1212-1218.

11. Dunion, J. P. , C. W. Landsea, and S. H. Houston, 2003: A re-analysis of the surface winds for Hurricane Donna of 1960. Mon. Wea. Rev., 131, 1992-2011.

12. Willoughby, H. E. and E. Rahn, 2002: A new parametric model of hurricane wind profiles. 25th AMS Conference on Hurricanes and Tropical Meteorology, San Diego, 29 April - 3 May 2002.

13. Powell, M. D., P. J. Vickery, and T. Reinhold, 2003: Reduced drag coefficient for high wind speeds in tropical cyclones. Nature, 422, 279-283.

14. Large, W. G. and S. Pond, 1981: Open ocean momentum flux measurements in moderate to strong winds. J. Phys. Oceanography, 11, 324-336.

15. Moss, M. S. and S. L. Rosenthal, 1975: On the estimation of planetary boundary layer variables in mature hurricanes. Mon. Wea. Rev., 106, 841-849.

16. Powell, M.D., 1980: Evaluations of diagnostic marine boundary layer models applied to hurricanes. Mon.Wea. Rev., 108, 757-766.

17. ASTM 1996: Standard practice for characterizing surface wind using a wind vane and rotating anemometer. D 5741-96, Annual Book of ASTM Standards, Vol. 11.03.

18. Anctil, F. and M. Donelan, 1996: Air-water momentum flux observations over shoaling waves. J. Phys. Oceanogr., 26, 1344-1353.

19. Reinhold, T. and K. Gurley, 2003: Florida Coastal Monitoring Program. http://www.ce.ufl.edu/~fcmp.

Section 3.3

Wind Speed Correction (WSC)
Use Case V

3.3.1 General Description Of WSC

WSC, short for Wind Speed Correction, is the fifth use case of the FIU/IHRC Public Hurricane Risk and Loss model. It aims at refining open terrain wind speed produced by the hurricane wind model with respect to the actual terrain (based on land use – land cover).

3.3.2 WSC General Requirements

Name: Wind Speed Correction

Description: The inputs are zip code, surface wind speed for open terrain produced by the wind model, surface wind direction, and roughness length for open terrain. The system generates the following:

(1) Surface wind speed for actual terrain (m/s).
(2) One-hour sustained wind speed for actual terrain (mph).
(3) 3-Second gust wind speed for actual terrain (mph).

1. Following are the input data:
   Zip: Zip Code
   Vo: Surface Wind Speed for open terrain produced by the wind model (m/s)
   WD: Surface Wind Direction (Deg from North)
   Zoo: Roughness Length (m) for open terrain = 0.03m
   Zoa: Roughness length based on upstream terrain
   Lat: Latitude

2. Based on the input data from step 1, the system queries the database and returns Zoa parameter, which corresponds to the actual roughness length based on FEMA HAZUS conversion table relating land use-land cover (LULC) to aerodynamic roughness (m). Roughness represents a weighted average of all roughness pixels within a 45-degree sector with origin at the population-weighted centroid of the zip code and extending outward to 20 km from the centroid. The weighting function for averaging the roughness values is a Gaussian filter with a half power point at 3 km. The format of the lookup table is as following:

Zip	**Lon**	**Lat**	**Zo1**	**Zo2**	**Zo3**	**Zo4**	**Zo5**	**Zo6**	**Zo7**	**Zo8**

Where:

Zo1 = Actual Roughness for wind directions inclusive of 46-90
Zo2 = Actual Roughness for wind directions inclusive of 1-45
Zo3 = Actual Roughness for wind directions inclusive of 316-0, 360
Z04 = Actual Roughness for wind directions inclusive of 271-315
Z05 = Actual Roughness for wind directions inclusive of 226-270
Zo6 = Actual Roughness for wind directions inclusive of 181-225
Zo7 = Actual Roughness for wind directions inclusive of 136-180
Zo8 = Actual Roughness for wind directions inclusive of 91-135

Table 3.3.1 shows a sample record from the lookup table.

Zip	33172
Lon	80.24401855
Lat	25.73268509
Zo1	0.2399817854
Zo2	0.3124250770
Zo3	0.3429141343
Zo4	0.3098731637
Zo5	0.3196663558
Zo6	0.2674820721
Zo7	0.5406716093E-01
Zo8	0.7273393869E-01

Table 3.3.1: A sample record for the lookup table

3. Given the wind direction for each zip code centroid, the appropriate value for actual terrain roughness is extracted from the lookup table. The system then computes the output values as below:

   3.1.Compute open terrain friction velocity *U*o* (Unit: m/s):
   Uo = Vo * 0.4 / [Ln (10.0 / 0.03)]

   3.2.Compute actual terrain friction velocity *U*a* (Unit: m/s, using equation 3 of Powell et al., 1996)
   Ua = Uo / ([Zoo / Zoa] ^ 0.0706)

   3.3.Compute actual terrain wind speed at 10 m *Va*:
   Va = (Ua / 0.4) (Ln (10 / Zoa))

   3.4. Convert wind speed to the unit of MPH:
   Vamph = Va * 2.24

Compute gust factors for peak 1 min wind over the hour G1h,60 and peak 3s wind over the hour G1h,3 based on the actual roughness. [See gust factor calculations below]

   3.5. Compute max 1 min wind (m/s) occurring within 1-hour period
   V1 = Va * G1h,60

   3.6.Compute max 1 min sustained wind speed in mph
   V1mph = V1 * 2.24

   3.7.Compute peak 3s gust in mph
   V3 = Va * G1h,3 * 2.24

4. Gust factor calculations

4.1. Compute friction velocity (u)

$$u = \frac{0.4Va}{Ln\left(\frac{10}{Zoa}\right)}$$

4.2. Compute Height scaling parameter based on a height of 10 m

$$\eta = 1 - 6f\left(\frac{10}{u}\right)$$

where $f = 2\left(7.292 * 10^{-5}\sin(Lat)\right)$ is the Coriolis parameter

4.3. Compute the standard deviation of the wind speed

$$\sigma_u(z) = \frac{7.5\eta u\left(0.09Ln\left(\frac{10}{Zoa}\right) + 0.538\right)^{\eta^{16}}}{\left(1 + 0.156Ln\left(\frac{u}{f.Zoa}\right)\right)}$$

4.4. Compute the standard deviation of the low-pass filtered wind speed considering a filter with a cut-off frequency of 1 cycle per 3 seconds (for the peak 3s gust) and 1 cycle per 60 seconds (for the maximum 1 min sustained wind speed calculation:

$$\sigma_u(z,60) = \sigma_u(z)\left(1 - 0.193\left(\frac{I_t}{60} + 0.1\right)^{-0.68}\right)$$

$$\sigma_u(z,60) = 0.386762\ \sigma_u(z)$$

Where 60 represents 1 min or 60 seconds and the integral scale time parameter I_t is,

$$I_t = 3.13Z^{0.2}$$

$$I_t = 4.96$$

In which Z = 10 meters is used.

$$\sigma_u(z,3) = \sigma_u(z)\left(1 - 0.193\left(\frac{I_t}{3} + 0.1\right)^{-0.68}\right)$$

$$\sigma_u(z,3) = 0.868256421\sigma_u(z)$$

Where 3 represents 3 seconds

4.5. Compute the wind fluctuation cycling rates:

$$C_r(60) = \frac{\left(0.007 + 0.213\left(\frac{I_t}{60}\right)^{0.654}\right)}{I_t}$$

$$C_r(60) = 0.00982$$

$$C_r(3) = \frac{\left(0.007 + 0.213\left(\frac{I_t}{3}\right)^{0.654}\right)}{I_t}$$

$$C_r(3) = 0.061$$

4.6. Compute the Peak factors for the max 1 min (60 sec) and max 3 second winds

$$P_f(3) = \left[\sqrt{2Ln(600C_r)} + \frac{0.557}{\sqrt{2Ln(600C_r)}}\right]\frac{\sigma_u(z,3)}{\sigma_u(z)}$$

$$P_f(60) = \left[\sqrt{2Ln(600C_r)} + \frac{0.557}{\sqrt{2Ln(600C_r)}}\right]\frac{\sigma_u(z,60)}{\sigma_u(z)}$$

4.7. Compute the longitudinal turbulent intensity

$$T_{il} = \frac{\sigma_u(z)}{U_h}$$

4.8. Compute the gust factors:

$$G_{10\ \mathrm{min},60} = 1 + T_{il}P_f(60)$$
$$G_{10\ \mathrm{min},3} = 1 + T_{il}P_f(3)$$

A sample calculation is as follows:

Input	Lookup value	Output

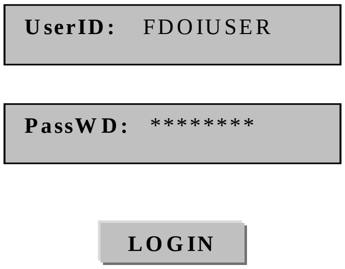
Zip = 33133
Vo = 50 m/s
WD = 60
Zoo = 0.03
Lat = 25.73
	Zoa = 0.219 m	U*o = 3.44 m/s U*a = 3.96 m/s Va = 37.845 m/s V1 = 101.504 mph V3 = 134.605 mph

3.3.3 WSC Interface Design Requirements

This section presents the Graphic User Interface design for the Wind Speed Correction (WSC).

1. The first step: the user logs in the system

Figure 3.3.1 shows the Login Interface. User needs to enter the user id and password to enter the system. The system verifies the user's information with the login data extracted from the database. If there is a match, the user logs into the system successfully. Otherwise, system displays the "wrong user name/password" error and requests the user to login again.



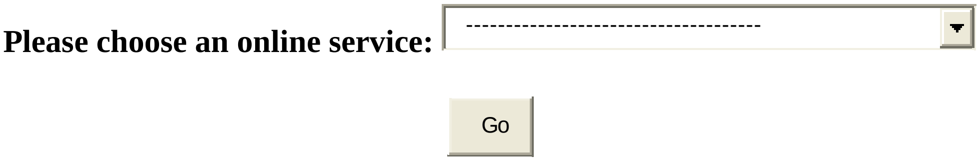
UserID: FDOIUSER

PassWD: ********

Figure 3.3.1: Login Interface

2. The second step: select the use case from the service selection page

Figure 3.3.2 is the service selection page interface. System presents a list of available use cases to the user. User selects "Roughness Model" use case and clicks "Go" to submit.



Please choose an online service: -------------------------------------

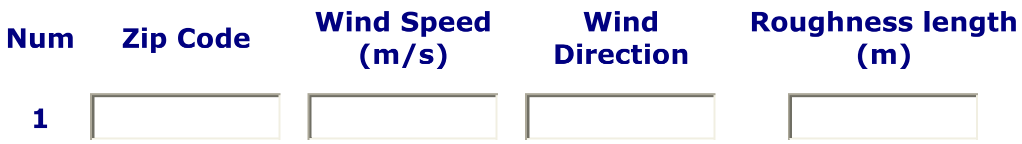
Figure 3.3.2: Service Selection Interface

3. The third step: The user provides the input from the wind model

In this step, system provides the interface for the user to input data generated by the wind model. The following inputs are required and are illustrated by Figure 3.3.3:

- *Zip code*:

- *Wind Speed*: Surface wind speed (m/s) for open terrain produced by the wind model.
- *Wind Direction*: Surface wind direction (Degree(s) from the North).
- *Roughness Length*: Roughness length (m) for open terrain = 0.03 m.



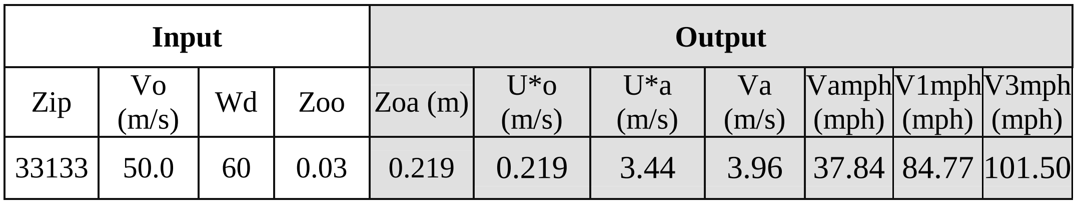
Num	Zip Code	Wind Speed (m/s)	Wind Direction	Roughness length (m)
1				

Figure 3.3.3: Input from Wind Model Interface

4. The forth step: The system displays the result in the interface

In this step, system calculates the result and displays both the input and the output to the user as shown in Figure 3.3.4. The output includes:

- *Zoa*: Actual roughness length based on FEMA HAZUS conversion table relating land use land cover (LULC) to aerodynamic roughness (m).
- *U*o*: Open terrain friction velocity (m/s).
- *U*a*: Actual terrain friction velocity (m/s).
- *Va*: Surface wind speed for actual terrain (m/s).
- *Vamph*: Above with English units of statute miles per hour.



Input				Output						
Zip	Vo (m/s)	Wd	Zoo	Zoa (m)	U*o (m/s)	U*a (m/s)	Va (m/s)	Vamph (mph)	V1mph (mph)	V3mph (mph)
33133	50.0	60	0.03	0.219	0.219	3.44	3.96	37.84	84.77	101.50

Figure 3.3.4: Wind Field Roughness Calculation Result Interface

3.3.4 Computer Model Design

3.3.4.1 Use Case View of WSC

A. Actors:

There is one actor, scientist in WSC.

D. Use Case:

WSC is used to determine a more accurate model of terrain winds produced by the hurricane wind model.

C. Use Case Diagram:

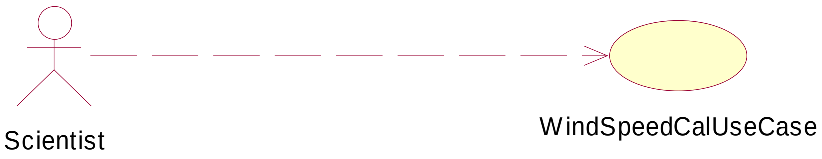


Figure 3.3.5: Use Case Diagram for WSC

3.3.4.2 System Design

This section includes the appropriate diagrams to describe the system classes, components, activities and the overall flow chart of WSC.

3.3.4.2.1 Program Flow Chart of WSC

The flow chart of WSC is depicted in Figure 3.3.6.

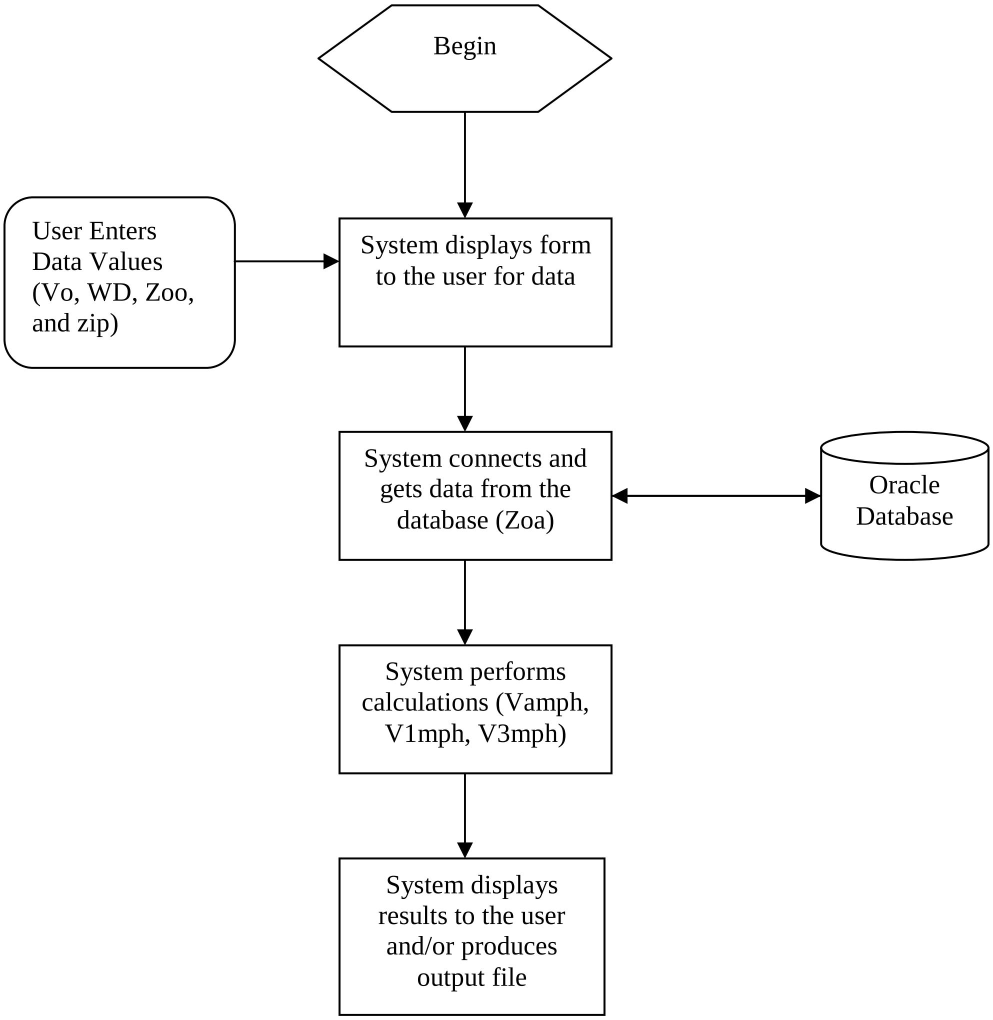

Begin
User Enters Data Values (Vo, WD, Zoo, and zip)
System displays form to the user for data
System connects and gets data from the database (Zoa)
Oracle Database
System performs calculations (Vamph, V1mph, V3mph)
System displays results to the user and/or produces output file

Figure 3.3.6: Flow chart of WSC

3.3.4.3 Class Diagram and Description

A. Class Diagram

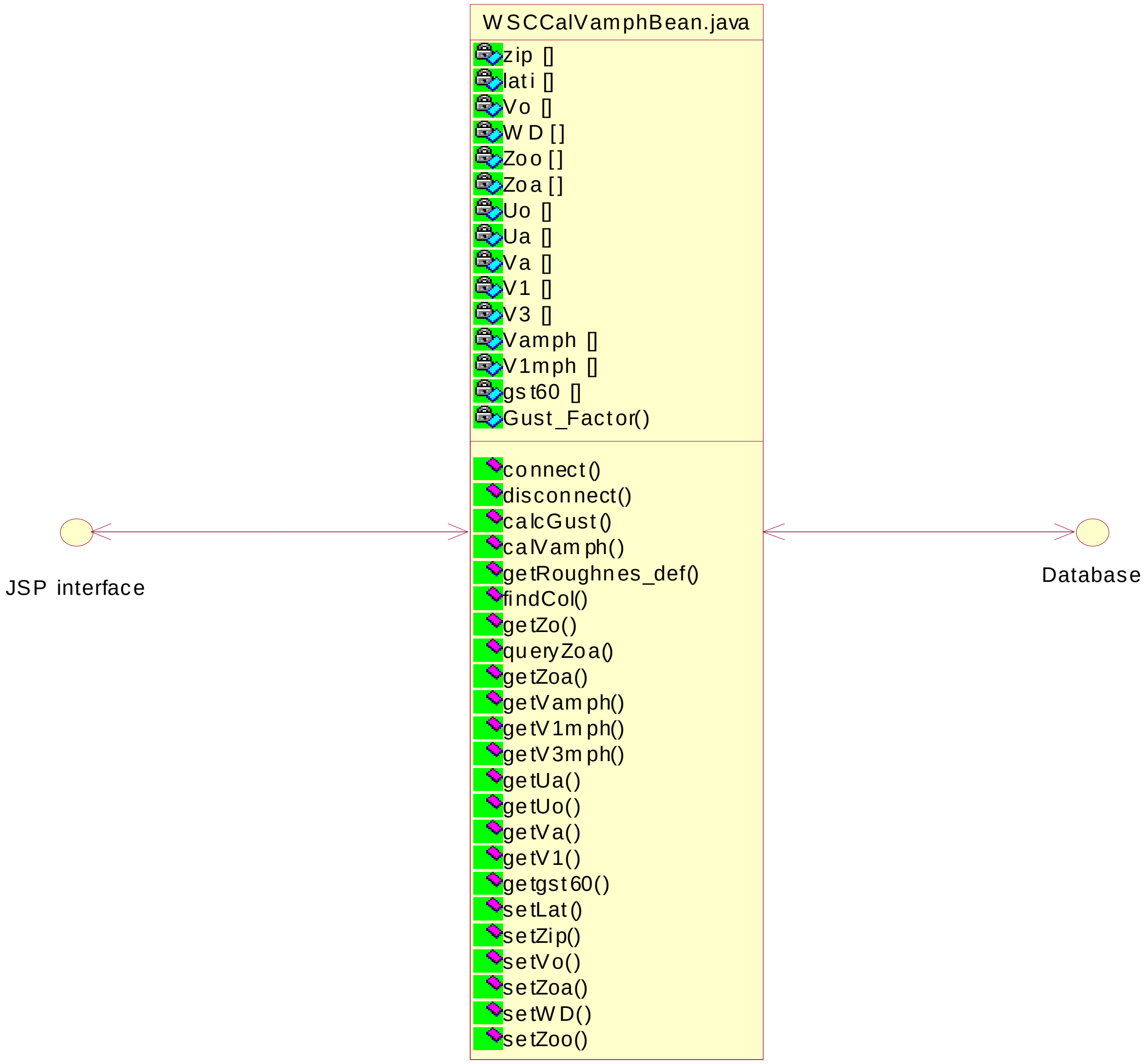


Figure 3.3.7: Class Diagram for WSC

B. Classes Descriptions

Here is a brief introduction of the functions in the class we used.

- WCSCalVamphBean
  This class performance all the functionalities needed for the wind speed correction calculation. It includes the following main methods:
  - connect()
    Method is used to establish a connection to the database

- setLat (double [] l)

Method takes an array of doubles and sets the latitude array to the passed array

- setZip(int [] z)

Method takes an array of integers and sets the zip array to the passed array

- setVo(double [] z)

Method takes an array of doubles and sets the Vo array to the passed array

- setWD(int [] w)

Method takes an array of integers and sets the WD array to the passed array

- setZoo(double [] z)

Method takes an array of doubles and sets the Zoo array to the passed array

- queryZoa()

Uses the connection to the database to send a query and retrieve the value for Zoa based on the zip and the returned string value from a call to the findCol method.

- calVamph()

Method is used to calculate the Vamph, Va, Uo and Ua using the following input values from the user Vo, Zoo and Zoa

- calcGust()

Method is used to calculate the gust factors $G_{1h,60}$ and $G_{1h,3}$ using the following input values Lat, WD, Zoa and Va

- GetRoughness_def()

Method uses the established connection to the database to find and return the column names and starting and ending degrees corresponding to each column of the roughness_def table.

- findCol(int wd)

Method takes a wind direction (WD) as a parameter. It uses the established connection to the database to find and return the correct string, using the WD, which represents the column in the lookup table for Zoa.

- get Zoa()

Return the value of Zoa

- getVamph()

Return the value of Vamph

- getV1mph()

Return the value of V1mph

- getV3mph()

Return the value of V3mph

- getUa()

Return the value of Ua

- getUo()

Return the value of Uo

- getVa()

Return the value of Va

- disconnect()

Method is used to disconnect from the database

3.3.4.4 State Chart Diagram

Figure 3.3.8 depicts the state chart diagram for Use Case Five. This diagram illustrates states that the use case goes through from beginning to end.

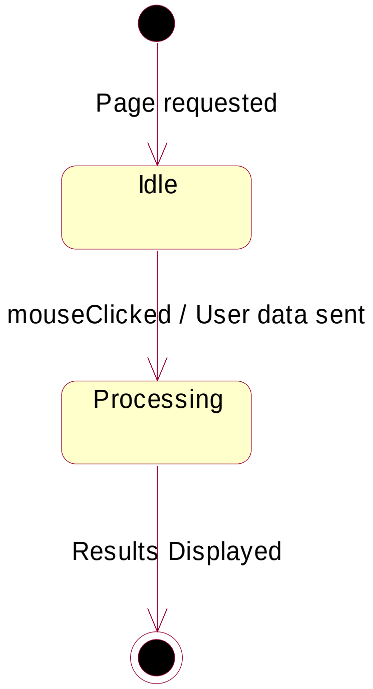


Figure 3.3.8: State Chart for WSC

3.3.4.5 Sequence Diagram

A. Sequence Diagram for the Vamph Calculation Process

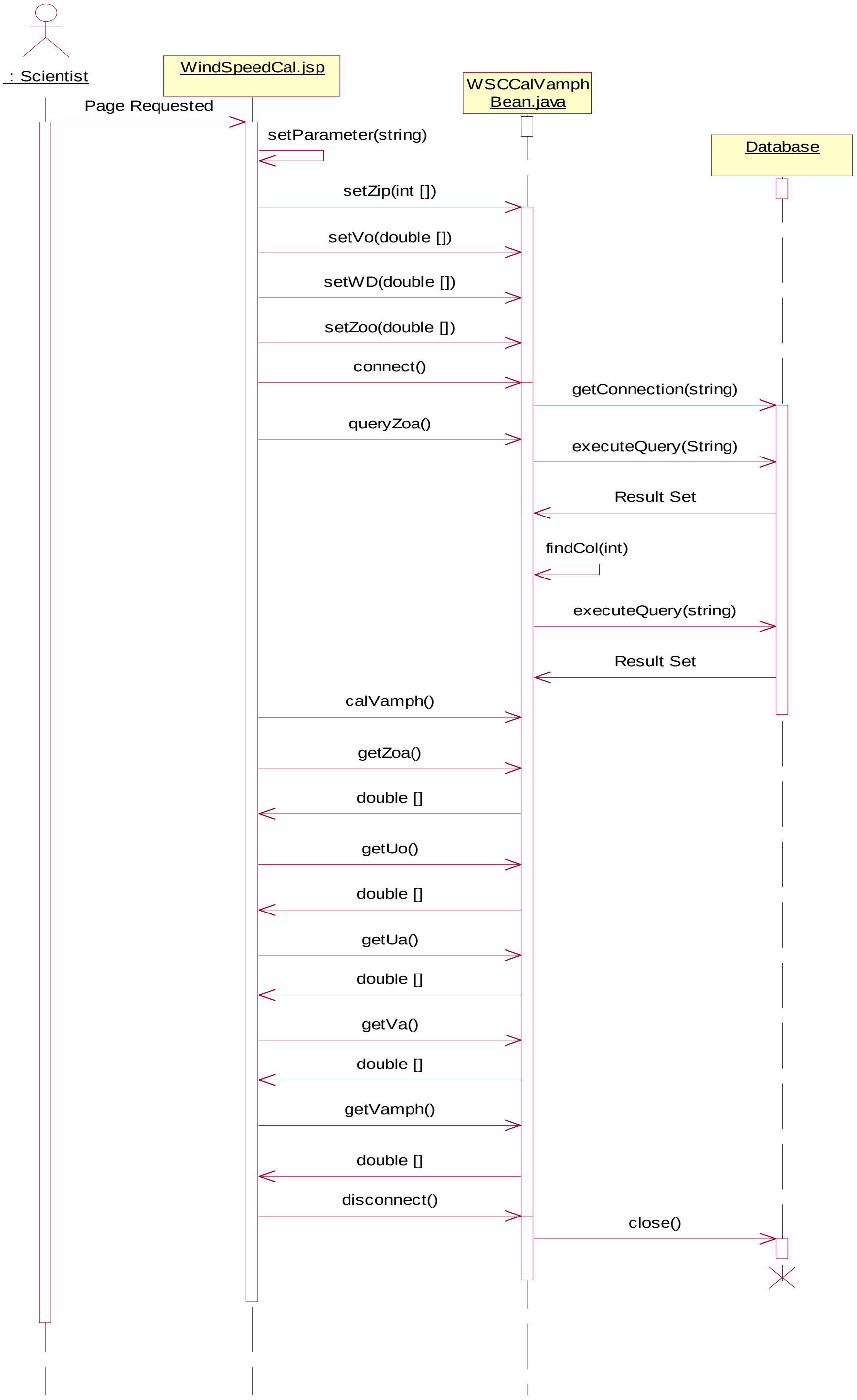


Figure 3.3.9: Sequence diagram for Vamph calculation process

B. Sequence Diagram Steps for the Vamph Calculation Process

- **Step 1**: The user requests the html page

- **Step 2**: The user enters the number of data sets to be calculated

- **Step 3**: The user’s input data for zip code, Vo, WD and Zoo are passed to WSCCalVamphBean object

- **Step 4**: WindSpeedCalc.jsp requests WSCCalVamphBean object to establish a connection to the database

- **Step 5**: WSCCalVamphBean establishes a connection with the database

- **Step 6**: WindSpeedCalc.jsp requests WSCCalVamphBean to query the database based on data passed from JSP

- **Step 7**: WSCCalVamphBean queries the database which returns a ResultSet to the BEAN

- **Step 8**: WSCCalVamphBean calls its findCol method.

- **Step 9**: WSCCalVamphBean queries the database which returns a ResultSet to the BEAN

- **Step 10**: WindSpeedCalc.jsp requests to perform Vamph calculations.

- **Step 11**: WindSpeedCalc.jsp requests the results for Zoo, Uo, Ua, Va and Vamph and WSCCalVamphBean returns the required data.

- **Step 12**: WindSpeedCalc.jsp notifies that it is okay to close the database connection

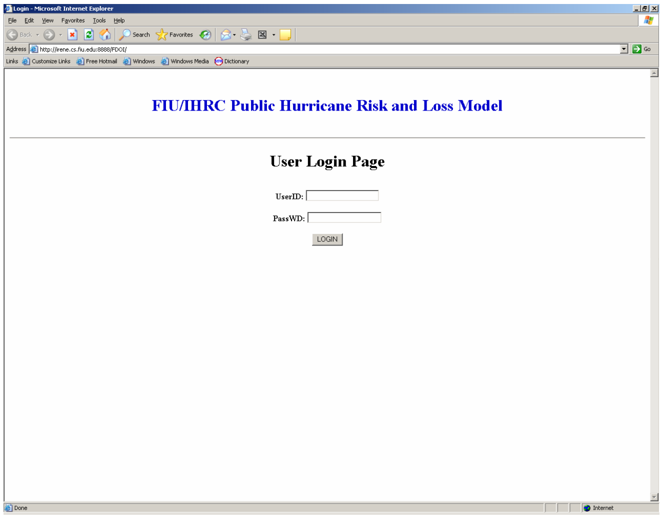
- **Step 13**: WSCCalVamphBean closes the database connection

3.3.5 Implementation of WSC

Currently, the implementation for Use Case five (WSC) has been finished. The demo is online at http://www.cs.fiu.edu/PHRLM.

3.3.5.1 Login page:

Users need a username and a password to access the *FIU/IHRC Public Hurricane Risk and Loss Model*. Following is a snapshot of the login web page.


Login - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://irene.cs.fiu.edu:8888/FDOI/
Go
Links Customize Links Free Hotmail Windows Windows Media Dictionary
Done
Internet

FIU/IHRC Public Hurricane Risk and Loss Model

User Login Page

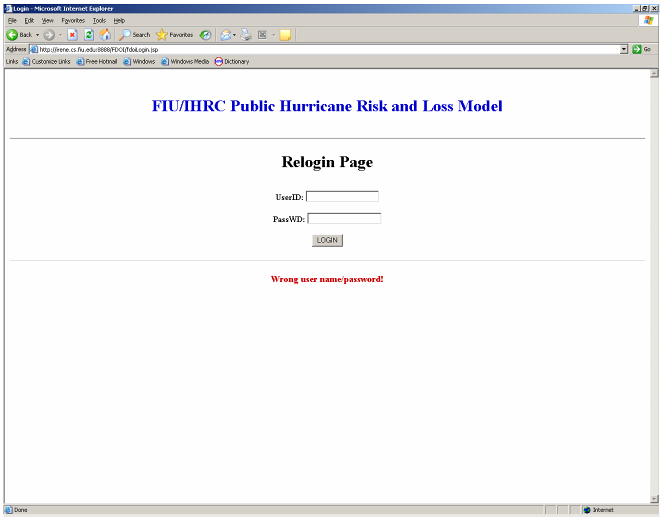
UserID:

PassWD:

LOGIN

Figure 3.3.10. Login webpage for FIU/IHRC PHRLM

If the username/password is wrong, an error message will be displayed. User will be required to input the username and password again to enter.


Login - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://irene.cs.fiu.edu:8888/FDOI/fdoiLogin.jsp
Go
Links Customize Links Free Hotmail Windows Windows Media Dictionary
Done
Internet

FIU/IHRC Public Hurricane Risk and Loss Model

Relogin Page

UserID:

PassWD:

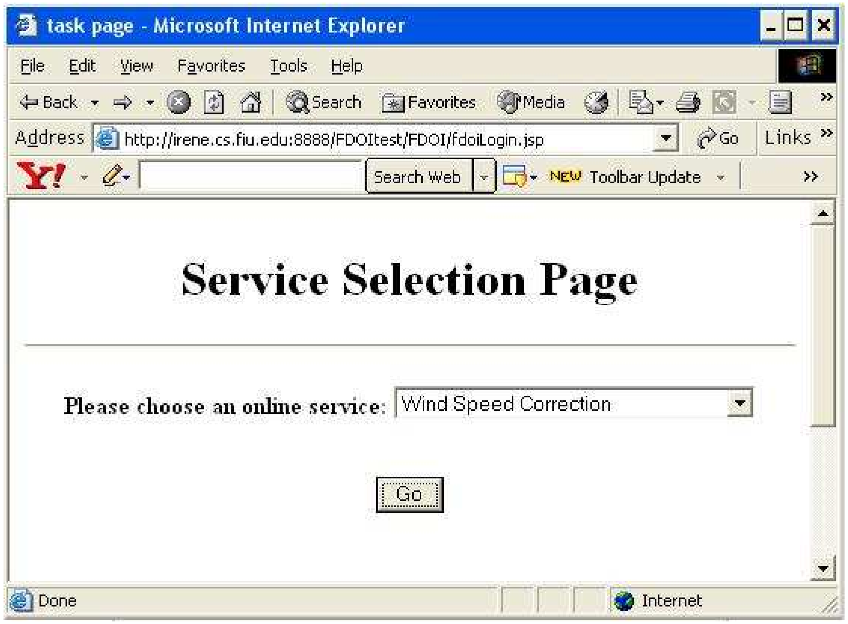
LOGIN

Wrong user name/password!

Figure 3.3.11. Login webpage shows the inputted user ID or password is wrong

3.3.5.2 WSC Page:

If the login is successful, the user can see the web page named “Service Selection Page” (as shown in Figure 3.3.12). To view the WSC use case page, from the drop-down list, select “Wind Speed Correction” and click “Go” button.


task page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://irene.cs.fiu.edu:8888/FDOItest/FDOI/fdoiLogin.jsp
Go Links
Search Web Toolbar Update
Done
Internet

Service Selection Page

Please choose an online service: Wind Speed Correction

Go

Figure 3.3.12. Service selection page for WSC

Several steps need to be followed to accomplish the task of Wind Speed Correction,
User can select two input methods.

a. Input from file (this is the use case four output)
b. Manual Input

If user want to take the input from file

1. Click on ‘From File’ radio button
2. Select the input data set from an available data set
3. Click on ‘submit’

If the user wants to enter the input manually, first click on ‘Manual Input’ radio button.

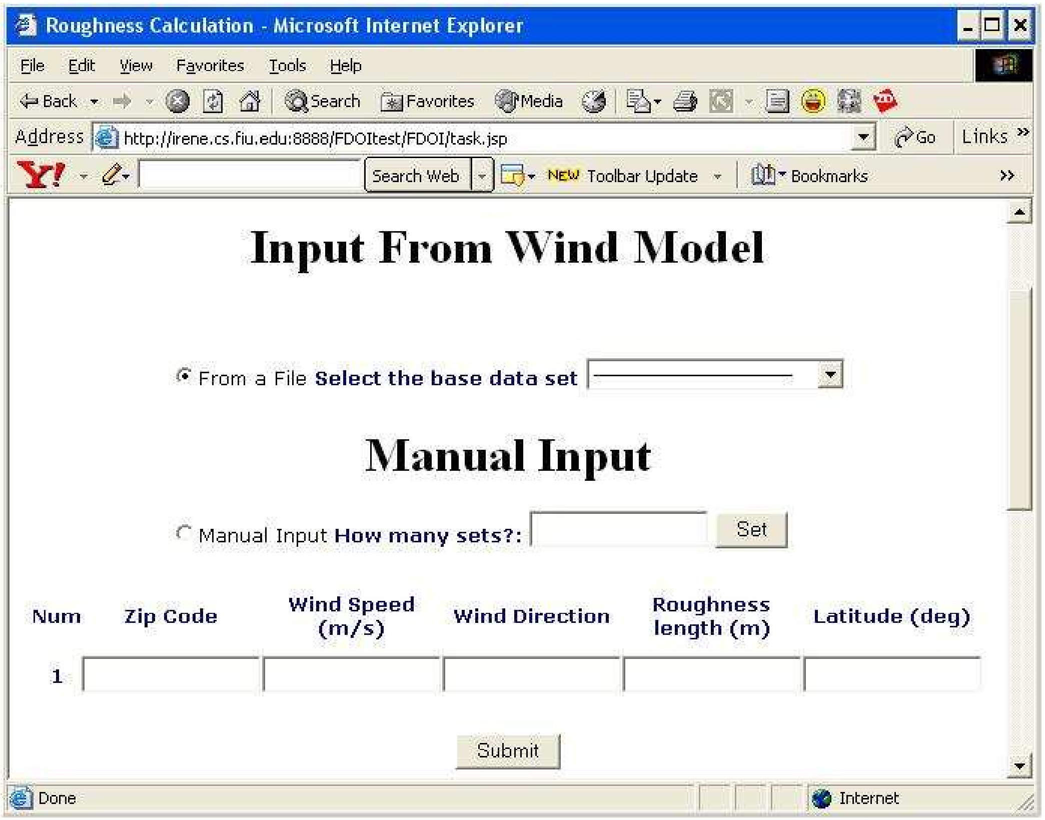
<u>Step 1:</u>

Then, the users need to input the wind field data fields. Input data sets from wind model include the following fields:

- ***Zip***: Zip code
- ***Vo***: Surface wind speed for open terrain produced by the wind model (m/s)
- ***Wd***: Surface wind direction (Deg from North). The data range for this field is from 0 to 360.
- ***Zoo***: Roughness length (m) for open terrain = 0.03 m
- ***Latitude:*** Latitude of the corresponding zip code. (a value between 20 and 40)

A data input page is provided to facilitate the users to input the corresponding data easily. Users are allowed to input a variety of collections of input data and submit this data for wind speed correction calculation.

By default, the number of input data sets is one. So initially, there is only one set of blanks for the user to input his/her data. Figure 3.3.13 illustrates the snapshot of the dataset input page for wind speed calculation.


Roughness Calculation - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://irene.cs.fiu.edu:8888/FDOItest/FDOI/task.jsp
Go Links
Search Web
NEW Toolbar Update
Bookmarks
Done
Internet

Input From Wind Model

● From a File **Select the base data set**

Manual Input

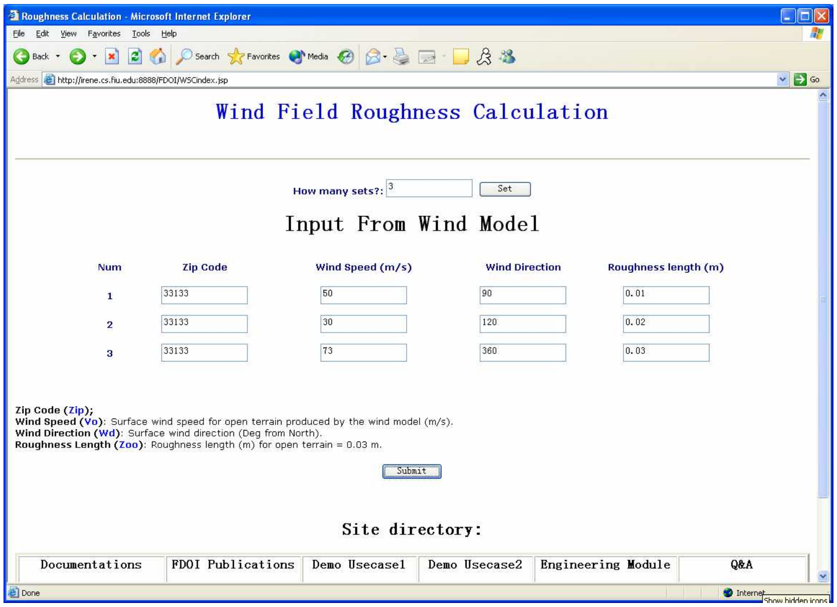
○ Manual Input **How many sets?:** [Set]

Num	Zip Code	Wind Speed (m/s)	Wind Direction	Roughness length (m)	Latitude (deg)
1					

[Submit]

Figure 3.3.13. Snapshot of the first web page for WSC

To input more than one set of input data, the user can change the number of input data sets by using the “number of sets change“ option provided at the top of the dataset input page. The user inputs the desired number of data sets in the blank after “How many sets?”, and then clicks “Set” button. Figure 3.3.14 shows an example dataset input web page after the user requests three sets of input data.


Roughness Calculation - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://irene.cs.fiu.edu:8888/FDOI/WSCindex.jsp
Go

Wind Field Roughness Calculation

How many sets?: 3 Set

Input From Wind Model

Num	Zip Code	Wind Speed (m/s)	Wind Direction	Roughness length (m)
1	33133	50	90	0.01
2	33133	30	120	0.02
3	33133	73	360	0.03

Zip Code (Zip);
Wind Speed (Vo): Surface wind speed for open terrain produced by the wind model (m/s).
Wind Direction (Wd): Surface wind direction (Deg from North).
Roughness Length (Zoo): Roughness length (m) for open terrain = 0.03 m.

Submit

Site directory:

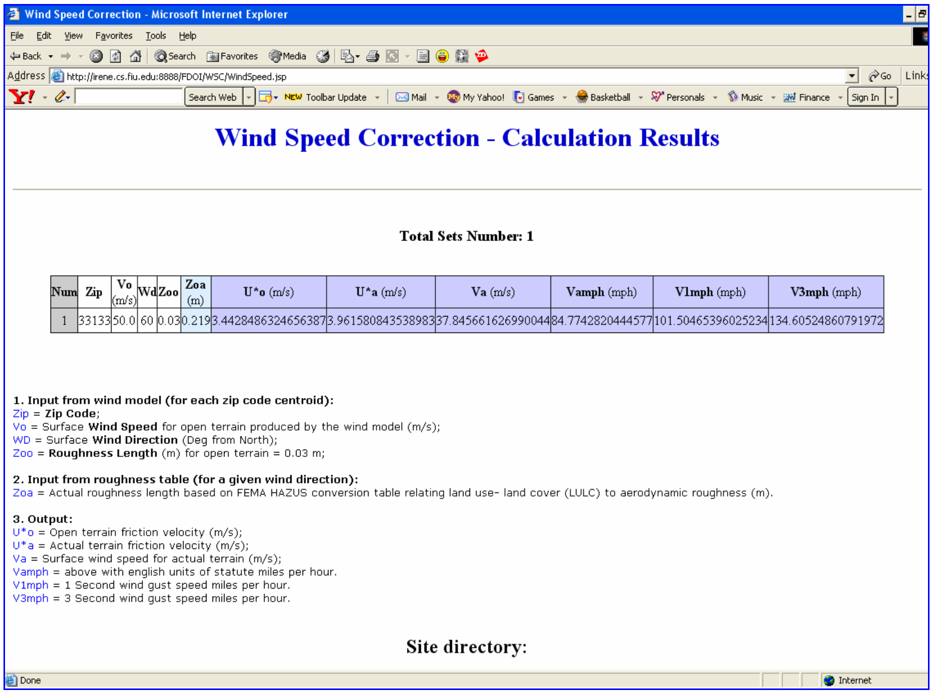
Documentations	FDOI Publications	Demo Usecase1	Demo Usecase2	Engineering Module	Q&A

Figure 3.3.14. Snapshot of the web page after user specify the sets number to 3

Step 2:

Secondly, the system constructs a query using the user’s data to obtain desired data from the underlying Oracle database. This desired data along with the user data is used to carry out the correction computation.

Once the correction computation is done, the system displays the whole data set: the input data set, retrieved data, and the computation results, back to the user. Figure 3.3.15 shows a result page as an example.


Wind Speed Correction - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://irene.cs.fiu.edu:8888/FDOI/WSC/WindSpeed.jsp
Done
Internet

Wind Speed Correction - Calculation Results

Total Sets Number: 1

Num	Zip	Vo (m/s)	Wd	Zoo	Zoa (m)	U*o (m/s)	U*a (m/s)	Va (m/s)	Vamph (mph)	V1mph (mph)	V3mph (mph)
1	33133	50.0	60	0.03	0.219	3.4428486324656387	3.961580843538983	37.84566162699004	84.7742820444577	101.50465396025234	134.60524860791972

1. Input from wind model (for each zip code centroid):
Zip = **Zip Code**;
Vo = Surface **Wind Speed** for open terrain produced by the wind model (m/s);
WD = Surface **Wind Direction** (Deg from North);
Zoo = **Roughness Length** (m) for open terrain = 0.03 m;

2. Input from roughness table (for a given wind direction):
Zoa = Actual roughness length based on FEMA HAZUS conversion table relating land use- land cover (LULC) to aerodynamic roughness (m).

3. Output:
U*o = Open terrain friction velocity (m/s);
U*a = Actual terrain friction velocity (m/s);
Va = Surface wind speed for actual terrain (m/s);
Vamph = above with english units of statute miles per hour.
V1mph = 1 Second wind gust speed miles per hour.
V3mph = 3 Second wind gust speed miles per hour.

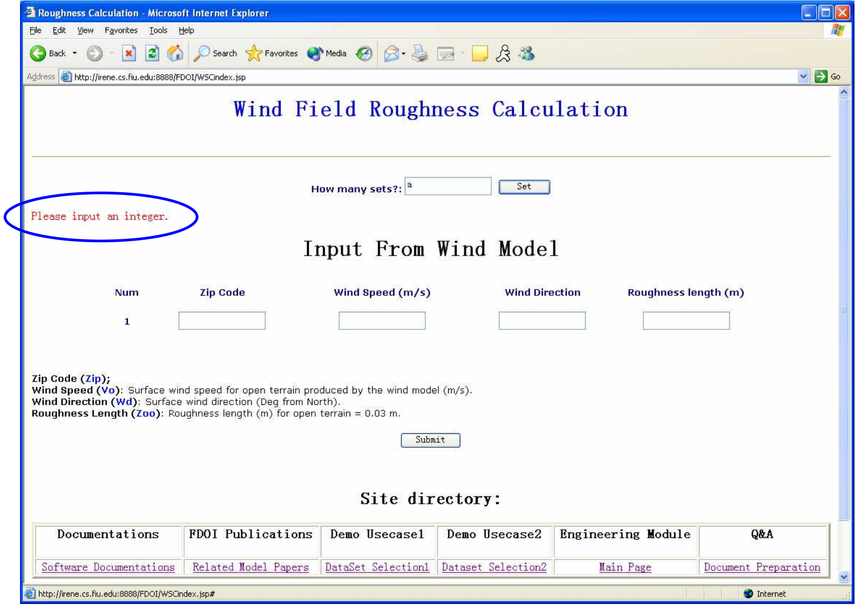
Site directory:

Figure 3.3.15: Snapshot of the result web page for WSC

3.3.5.3 Exception Handling:

Users may make some error inputs. The JSP webpage can catch the exceptions and show the error messages. Here we show some examples:

After user fills in the set number blank and clicks the “Set” button, the system will check if the inputted value is an integer. If no, the corresponding error message will be generated and shown under this blank (as shown in Figure 3.3.16).


Roughness Calculation - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://irene.cs.fiu.edu:8888/FDOI/WSCindex.jsp
Set
Submit
http://irene.cs.fiu.edu:8888/FDOI/WSCindex.jsp#
Internet

Wind Field Roughness Calculation

How many sets?: a

Please input an integer.

Input From Wind Model

Num	Zip Code	Wind Speed (m/s)	Wind Direction	Roughness length (m)
1				

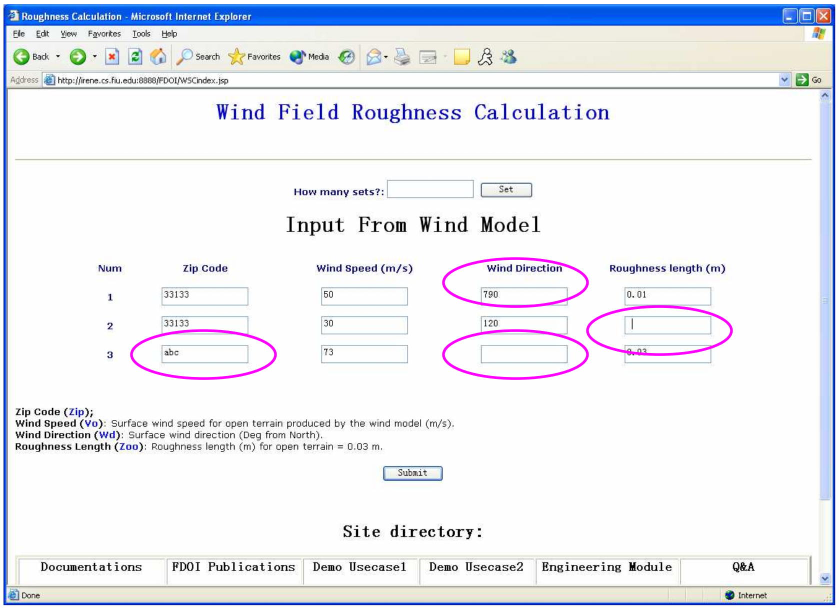
Zip Code (Zip);
Wind Speed (Vo): Surface wind speed for open terrain produced by the wind model (m/s).
Wind Direction (Wd): Surface wind direction (Deg from North).
Roughness Length (Zoo): Roughness length (m) for open terrain = 0.03 m.

Site directory:

Documentations	FDOI Publications	Demo Usecase1	Demo Usecase2	Engineering Module	Q&A
Software Documentations	Related Model Papers	DataSet Selection1	Dataset Selection2	Main Page	Document Preparation

Figure 3.3.16. The input webpage shows the exception that the inputted set number is not an integer.

User need to follow the instructions, fill in all the blanks and input the data sets in the correct format. Figure 3.3.17 shows some error inputs. For example: some blanks are not filled; the zip code field is filled by words; the wind direction is not in the interval [0, 360]; etc. Figure 3.3.18 displays the webpage which caught these exceptions and displays several error messages.


Roughness Calculation - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://irene.cs.fiu.edu:8888/FDOI/WSCindex.jsp
Go
Done
Internet

Wind Field Roughness Calculation

How many sets?: Set

Input From Wind Model

Num	Zip Code	Wind Speed (m/s)	Wind Direction	Roughness length (m)
1	33133	50	790	0.01
2	33133	30	120	
3	abc	73		0.03

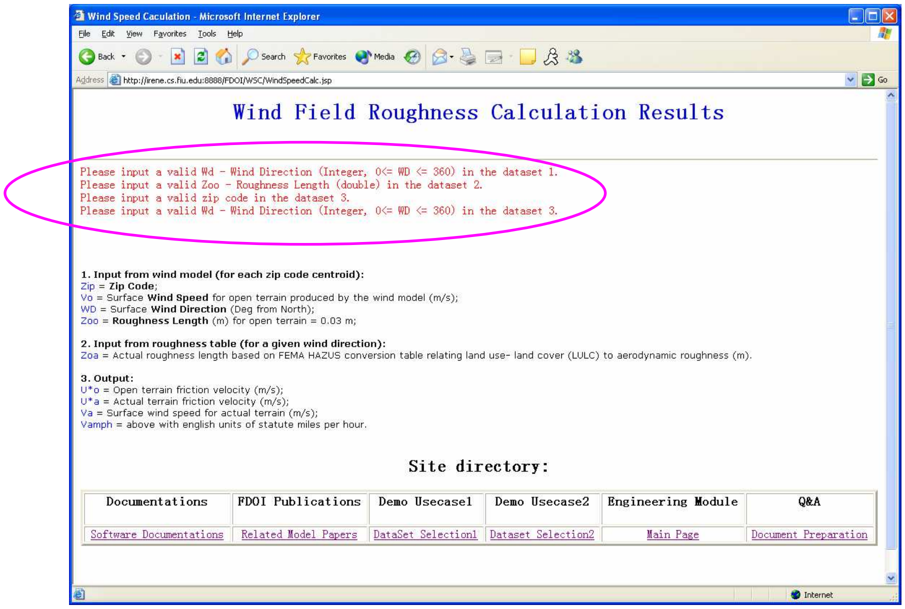
Zip Code (Zip);
Wind Speed (Vo): Surface wind speed for open terrain produced by the wind model (m/s).
Wind Direction (Wd): Surface wind direction (Deg from North).
Roughness Length (Zoo): Roughness length (m) for open terrain = 0.03 m.

Submit

Site directory:

Documentations	FDOI Publications	Demo Usecase1	Demo Usecase2	Engineering Module	Q&A

Figure 3.3.17: The input webpage contains a set of error inputs


Wind Speed Caculation - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://irene.cs.fiu.edu:8888/FDOI/WSC/WindSpeedCalc.jsp
Go
Internet

Wind Field Roughness Calculation Results

Please input a valid Wd - Wind Direction (Integer, 0<= WD <= 360) in the dataset 1.
Please input a valid Zoo - Roughness Length (double) in the dataset 2.
Please input a valid zip code in the dataset 3.
Please input a valid Wd - Wind Direction (Integer, 0<= WD <= 360) in the dataset 3.

1. Input from wind model (for each zip code centroid):
Zip = **Zip Code**;
Vo = Surface **Wind Speed** for open terrain produced by the wind model (m/s);
WD = Surface **Wind Direction** (Deg from North);
Zoo = **Roughness Length** (m) for open terrain = 0.03 m;

2. Input from roughness table (for a given wind direction):
Zoa = Actual roughness length based on FEMA HAZUS conversion table relating land use- land cover (LULC) to aerodynamic roughness (m).

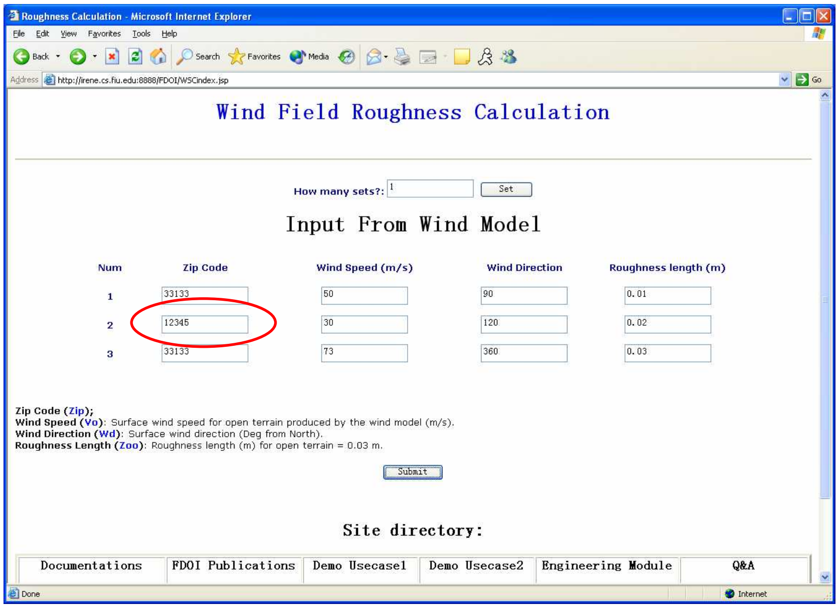
3. Output:
U*o = Open terrain friction velocity (m/s);
U*a = Actual terrain friction velocity (m/s);
Va = Surface wind speed for actual terrain (m/s);
Vamph = above with english units of statute miles per hour.

Site directory:

Documentations	FDOI Publications	Demo Usecase1	Demo Usecase2	Engineering Module	Q&A
Software Documentations	Related Model Papers	DataSet Selection1	Dataset Selection2	Main Page	Document Preparation

Figure 3.3.18: The result webpage catches all the available exceptions

Another possible exception is caused by the invalid zip code. Sometimes the user inputs the zip code within the correct format, but this zip code cannot be found in the specific lookup table. As shown in the Figure 3.3.19, user inputted a zip code as “12345”, which is not in the lookup table. In Figure 3.3.20, the webpage shows this message so that the user can identify this error.


Roughness Calculation - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://irene.cs.fiu.edu:8888/FDOI/WSCindex.jsp
Go
Done
Internet

Wind Field Roughness Calculation

How many sets?: 1 Set

Input From Wind Model

Num	Zip Code	Wind Speed (m/s)	Wind Direction	Roughness length (m)
1	33133	50	90	0.01
2	12345	30	120	0.02
3	33133	73	360	0.03

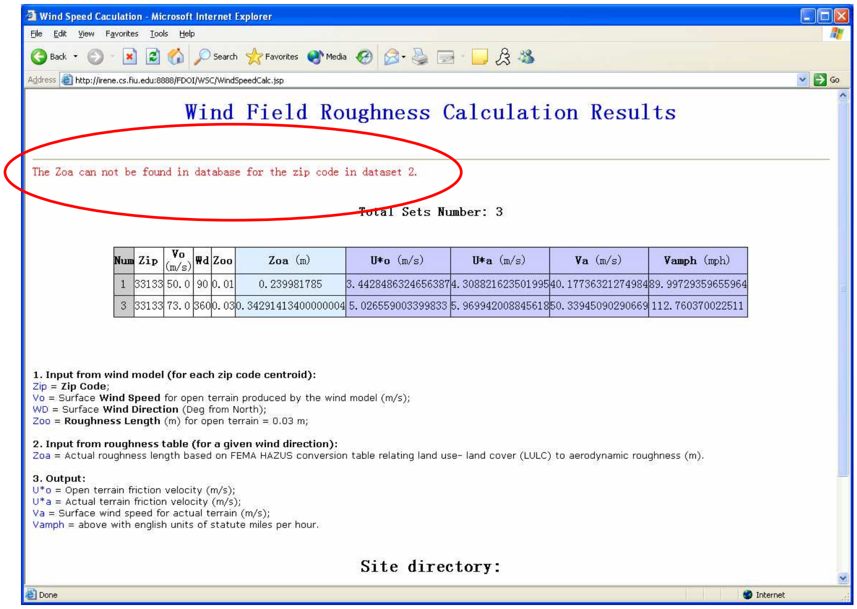
Zip Code (Zip);
Wind Speed (Vo): Surface wind speed for open terrain produced by the wind model (m/s).
Wind Direction (Wd): Surface wind direction (Deg from North).
Roughness Length (Zoo): Roughness length (m) for open terrain = 0.03 m.

Submit

Site directory:

Documentations	FDOI Publications	Demo Usecase1	Demo Usecase2	Engineering Module	Q&A

Figure 3.3.19: The input webpage contains a Zip code value which is not included in the lookup table


Wind Speed Caculation - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://irene.cs.fiu.edu:8888/FDOI/WSC/WindSpeedCalc.jsp
Go
Done
Internet

Wind Field Roughness Calculation Results

The Zoa can not be found in database for the zip code in dataset 2.

Total Sets Number: 3

Num	Zip	Vo (m/s)	Wd	Zoo	Zoa (m)	U*o (m/s)	U*a (m/s)	Va (m/s)	Vamph (mph)
1	33133	50.0	90	0.01	0.239981785	3.4428486324656387	4.308821623501995	40.17736321274984	89.99729359655964
3	33133	73.0	360	0.03	0.34291413400000004	5.026559003399833	5.969942008845618	50.33945090290669	112.760370022511

1. Input from wind model (for each zip code centroid):
Zip = **Zip Code**;
Vo = Surface **Wind Speed** for open terrain produced by the wind model (m/s);
WD = Surface **Wind Direction** (Deg from North);
Zoo = **Roughness Length** (m) for open terrain = 0.03 m;

2. Input from roughness table (for a given wind direction):
Zoa = Actual roughness length based on FEMA HAZUS conversion table relating land use- land cover (LULC) to aerodynamic roughness (m).

3. Output:
U*o = Open terrain friction velocity (m/s);
U*a = Actual terrain friction velocity (m/s);
Va = Surface wind speed for actual terrain (m/s);
Vamph = above with english units of statute miles per hour.

Site directory:

Figure 3.3.20: The result webpage which caught the exception when the Zoa value can not be fetched for some specific Zip code

Section 3.4

Wind Speed Probability (WSP)
Use Case VI

3.4.1 General Description Of WSP

WSP, short for Wind Speed Probability, is the sixth use case of the FIU/IHRC Public Hurricane Risk and Loss model. It aims at calculating the probabilities of the 3s gust wind speeds affecting each of the zip codes in the threat area.

3.4.2 WSP General Requirements

Name: Wind Speed Probability

Description: The user provides the system with the surface corrected wind speed time series for each of the storm. The wind speeds are in units of miles per hour (mph).

The system computes the following:

(1) For each zip code the annual probability of the maximum wind speeds being within the 5 mph interval for all storms starting at 22.5 mph and ending at 302.5 mph.

(2) For each zip code the annual probability of the wind speeds exceeding the mid-point m for each of the 5 mph interval for all storms starting at 22.5 mph and ending at 302.5 mph.

1. The end user enters the following input data:

   Y: Number of years in simulation
   T: Wind speed time series for each of the storms (one file for each storm)
   Z: All the zip code in threat area

   Each of the *T* consists of following pertinent information

   Storm_name: Storm name
   Storm_date_time: Storm landfall date and time
   Zip: Zip Code
   V3: Surface corrected 3Sec gust wind speed in mph obtained from the WSC use case

   A sample input is as follows:

```
storm0000002   8/19/    1  15:00
Num  Zip       Vo      Wd     Zoo     Zoa     U*o     U*a      Va      V1mph        V3mph
1    32008  21.6816   150    0.03    0.664   1.492   1.857   12.596    38.203       55.113
2    32009  14.6592   163    0.03    0.818   1.009   1.274    7.978    24.786       36.494
3    32011  14.2175   162    0.03    0.699   0.978   1.222    8.132    24.949       36.345
…    …         …       …       …       …       …       …       …       …       …       …
```

2. For each of the zip code:
   a. From each storm wind speed time series we select the maximum 3S gust wind speed (V3).
   b. Wind speeds (V3) affecting each of the zip code for N number of storms in the simulation (affecting each zip code) are then sorted in an ascending order.
   c. The sorted vector of wind speeds is then distributed into bands of 5 mph starting at 22.5 mph and ending at 302.5 mph. Wind speeds over the midpoint of each band are assigned to the corresponding bin.

d. Count the number of instances of maximum wind speeds within each of the wind speed bands and below the midpoint of each band.

e. Compute the annual probability of the maximum wind speed being within each band defined by wind speed x (on low side) and z (on high side) as:

$$\text{PVxz} = \frac{(\text{Number of Instances of } V3 >= x \quad \text{and} < z)}{N}$$

f. Compute the probability of the wind speed exceeding the mid point v of the band as:

$$\text{Pv}< v = \frac{(\text{Number of Instances of } V3 < v)}{N}$$

N= number of storms affecting that zip code

g. Estimate

λ = (total #of storms affecting that zip code)/ (# simulation Years)

h. Estimate the probabilities:

$$P(V>v) \qquad = 1 - e^{-\lambda}\, e^{\lambda Pv}$$

Also $P(V < v) = 1- e^{-\lambda}\, e^{\lambda(1-Pv)}$

$$P(z < V < x) = \sum_{n=1}^{\infty}\left(\sum_{j=1}^{n}\frac{n!}{j!(n-j)!}P_{xz}^{j}(1-P_{xz})^{n-j}\right)\frac{(\lambda)^{n}e^{-\lambda}}{n!}$$

3. Plot histogram and cumulative frequency diagram of maximum wind speeds for each zip code, use x axis with wind speeds at 5 mph resolution starting at 22.5 mph and ending at 302.5 mph.

A sample calculation result is as follows:

```
PVxz = annual probability of a wind(Gust) speed within the band
indicated at left, greater than the first number and less than or equal
to the second number
PV>y = annual probability of the wind(Gust) speed exceeding the mid
point of the band

ZipCode = 41
Gust Band(mph)   Mid-Point   PVxz   Pv
> - =<
22.5- 27.5        25.0        0.017 0.262
27.5- 32.5        30.0        0.022 0.247
32.5- 37.5        35.0        0.022 0.229
37.5- 42.5        40.0        0.023 0.211
42.5- 47.5        45.0        0.022 0.192
47.5- 52.5        50.0        0.019 0.176
52.5- 57.5        55.0        0.017 0.160
57.5- 62.5        60.0        0.017 0.146
62.5- 67.5        65.0        0.017 0.131
```

```
67.5- 72.5        70.0        0.017 0.116
72.5- 77.5        75.0        0.017 0.102
77.5- 82.5        80.0        0.014 0.088
82.5- 87.5        85.0        0.014 0.075
87.5- 92.5        90.0        0.011 0.064
92.5- 97.5        95.0        0.010 0.054
97.5- 102.5      100.0        0.008 0.045
102.5- 107.5     105.0        0.008 0.037
107.5- 112.5     110.0        0.006 0.031
112.5- 117.5     115.0        0.005 0.025
117.5- 122.5     120.0        0.004 0.020
122.5- 127.5     125.0        0.004 0.016
127.5- 132.5     130.0        0.003 0.013
132.5- 137.5     135.0        0.002 0.010
137.5- 142.5     140.0        0.002 0.008
142.5- 147.5     145.0        0.002 0.006
147.5- 152.5     150.0        0.001 0.005
152.5- 157.5     155.0        0.001 0.003
157.5- 162.5     160.0        0.001 0.002
162.5- 167.5     165.0        0.000 0.002
167.5- 172.5     170.0        0.000 0.001
172.5- 177.5     175.0        0.000 0.001
177.5- 182.5     180.0        0.000 0.001
182.5- 187.5     185.0        0.000 0.000
187.5- 192.5     190.0        0.000 0.000
192.5- 197.5     195.0        0.000 0.000
197.5- 202.5     200.0        0.000 0.000
202.5- 207.5     205.0        0.000 0.000
207.5- 212.5     210.0        0.000 0.000
212.5- 217.5     215.0        0.000 0.000
217.5- 222.5     220.0        0.000 0.000
222.5- 227.5     225.0        0.000 0.000
227.5- 232.5     230.0        0.000 0.000
232.5- 237.5     235.0        0.000 0.000
237.5- 242.5     240.0        0.000 0.000
242.5- 247.5     245.0        0.000 0.000
247.5- 252.5     250.0        0.000 0.000
252.5- 257.5     255.0        0.000 0.000
257.5- 262.5     260.0        0.000 0.000
262.5- 267.5     265.0        0.000 0.000
267.5- 272.5     270.0        0.000 0.000
272.5- 277.5     275.0        0.000 0.000
277.5- 282.5     280.0        0.000 0.000
282.5- 287.5     285.0        0.000 0.000
287.5- 292.5     290.0        0.000 0.000
292.5- 297.5     295.0        0.000 0.000
297.5- 302.5     300.0        0.000 0.000
```

3.4.3 Computer Model Design

3.4.3.1 Use Case View of WSP

A. Actors:

There is one actor, scientist in WSP.

B. Use Case:

WSP is used to calculate the annual probabilities of the wind speeds lying in a wind speed interval and lying over the mid point the interval.

C. Use Case Diagram:

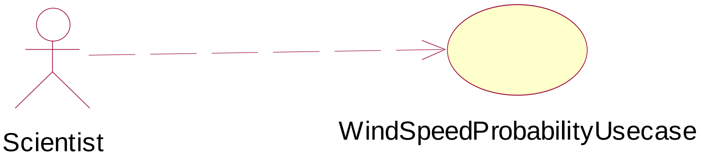


Figure 3.4.1: Use Case Diagram for WSP

3.4.3.2 System Design

This section includes the appropriate diagrams to describe the system classes, components, activities and the overall flow chart of WSP.

General Flow Chart of WSP

The flow chart of WSP is depicted in Figure 3.4.2.

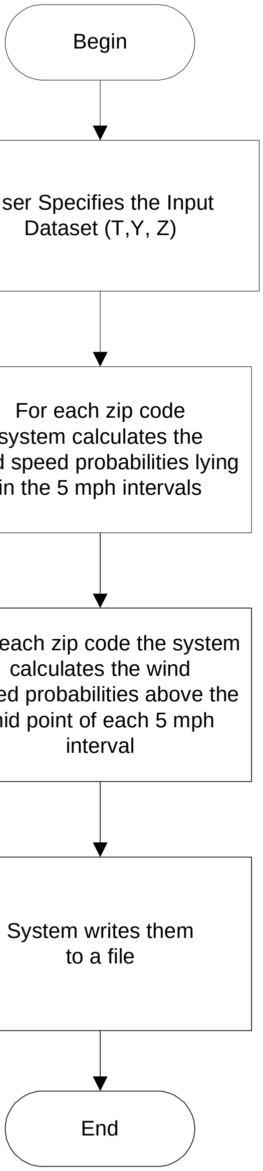

Begin
User Specifies the Input Dataset (T,Y, Z)
For each zip code system calculates the wind speed probabilities lying in the 5 mph intervals
For each zip code the system calculates the wind speed probabilities above the mid point of each 5 mph interval
System writes them to a file
End

Figure 3.4.2: Flow chart of WSC

3.4.3.3 Calculation of WSP

The implementation for Use Case six (WSP) has been completed and meets the requirements specification. The back end for the implementation of WSP use case has been coded in C++.

Input:

1. Zip codes in threat area
2. For each storm passing within 3 Rmax of the threat area:
   - Peak max sustained wind speed and associated peak gust wind speed already corrected for the upstream terrain of the zip code
   - Storm name, date, and time for each peak wind

Implementation:

(1) For each zip code:

(2) For each storm, select the maximum 3s Gust wind speed (V3)

Note: These wind speeds are in units of miles per hour and should already have been corrected for roughness.

(3) Order according to the peak maximum sustained wind speeds of all storms affecting the zip code from low to high

(4) Count the number of instances of maximum wind speeds within each wind speed band and above the mid point of each band.

(5) Estimate P_v, probability of the maximum wind speed being less than mid point v as:

P_v =(Count V< v) /N

Where N = Number of Storms affecting that Zip code

Estimate:

$P(z < V < x) = P_{xz} = \text{Count } (z < V < x)/N$

(6) Estimate:

λ = (total #of storms affecting that zip code)/(total # of years in simulation)

λ = N/ (total # of years in simulation)

(7) Compute the probabilities:

$P(V>v) = 1 - e^{-\lambda} e^{\lambda P v}$

Also:

$P(V<v) = 1- e^{-\lambda} e^{\lambda(1-Pv)}$

(8) $$P(z < V < x) = \sum_{n=1}^{\infty}\left(\sum_{j=1}^{n}\frac{n!}{j!(n-j)!}P_{xz}^{j}(1-P_{xz})^{n-j}\right)\frac{(\lambda)^{n}e^{-\lambda}}{n!}$$

This above equation is going to require a program – however, as n gets large, the Poisson probabilities should start to go to 0 and therefore, the sum might not be too formidable. So the maximum limit of n is taken as 13.

3.4.3.4 Class Diagram

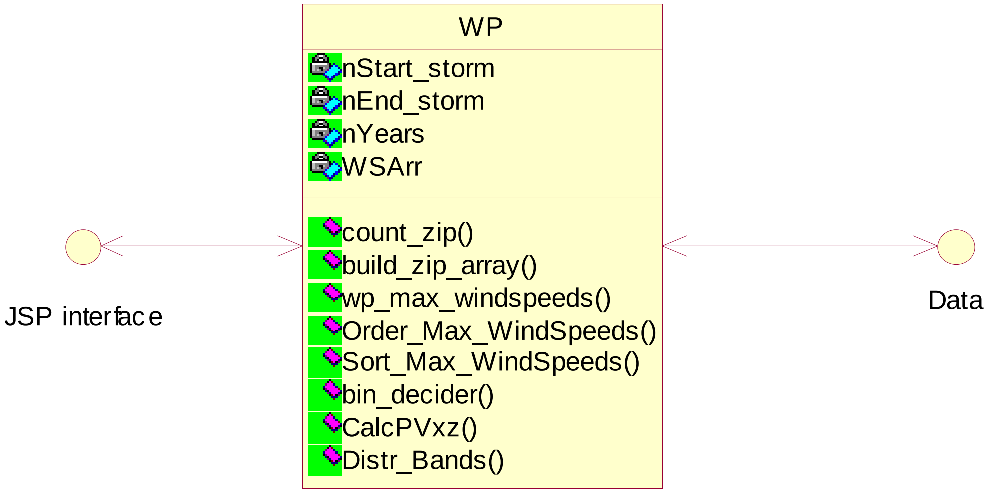


Figure 3.4.3: Class Diagram for WSP

3.4.3.5 Class Description

WP class performs all the calculations needed for WSP use case. It includes following methods.

- **`count_zip()`**
  Read the zipcodes.txt file to count and return the number of lines in the file.
- **`build_zip_array()`**
  Read the zipcodes.txt file and store the zip codes in to an array
- **`wp_max_windspeeds()`**
  Search through one storm to find the maximum 3s Gust wind speeds at each zip code and record it as intermediate output. Iterate this process through all storms.
- **`Order_Max_WindSpeeds()`**
  Search through all the storms and record (in an array) the maximum wind speed for specified zip for each storm.
- **`Sort_Max_WindSpeeds()`**
  Sort the wind speed array in descending order
- **`bin_decider()`**
  maps the specified wind speed to a bin
- **`Distr_Bands()`**
  Estimates the wind speed probabilities of the wind speeds lying in the band (PVxz) and above the midpoint of the band (Pv>y)
- **`CalcPVxz()`**
  Calculate the wind probabilities Pv and PVxz and produces the CalcP__.txt output file.

3.4.3.6 State Chart Diagram

Figure 3.4.4 depicts the state chart diagram for WSP use case. This diagram illustrates states that the use case goes through from beginning to end.

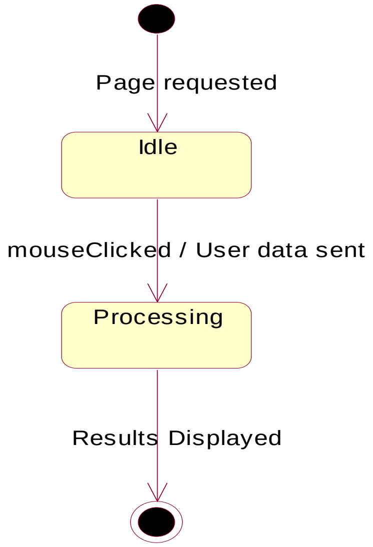


Figure 3.4.4: State Chart for WSP

3.4.3.7 Sequence Diagram

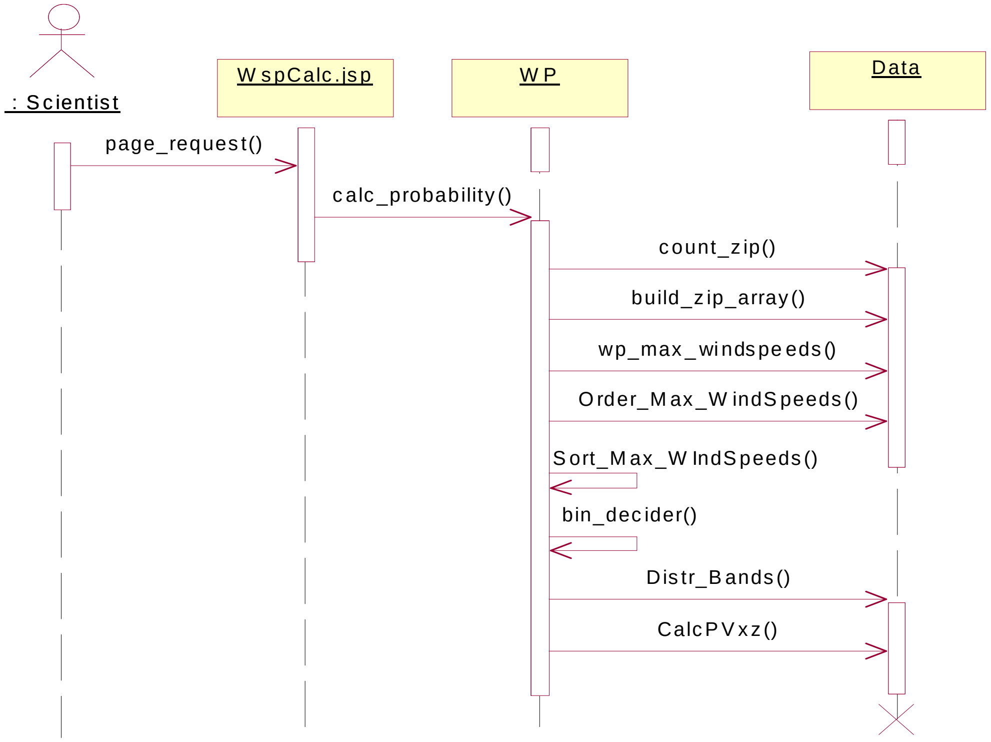


Figure 3.4.5: State Chart for Wind Probability Calculation Process

3.4.3.8 Program Flow Chart

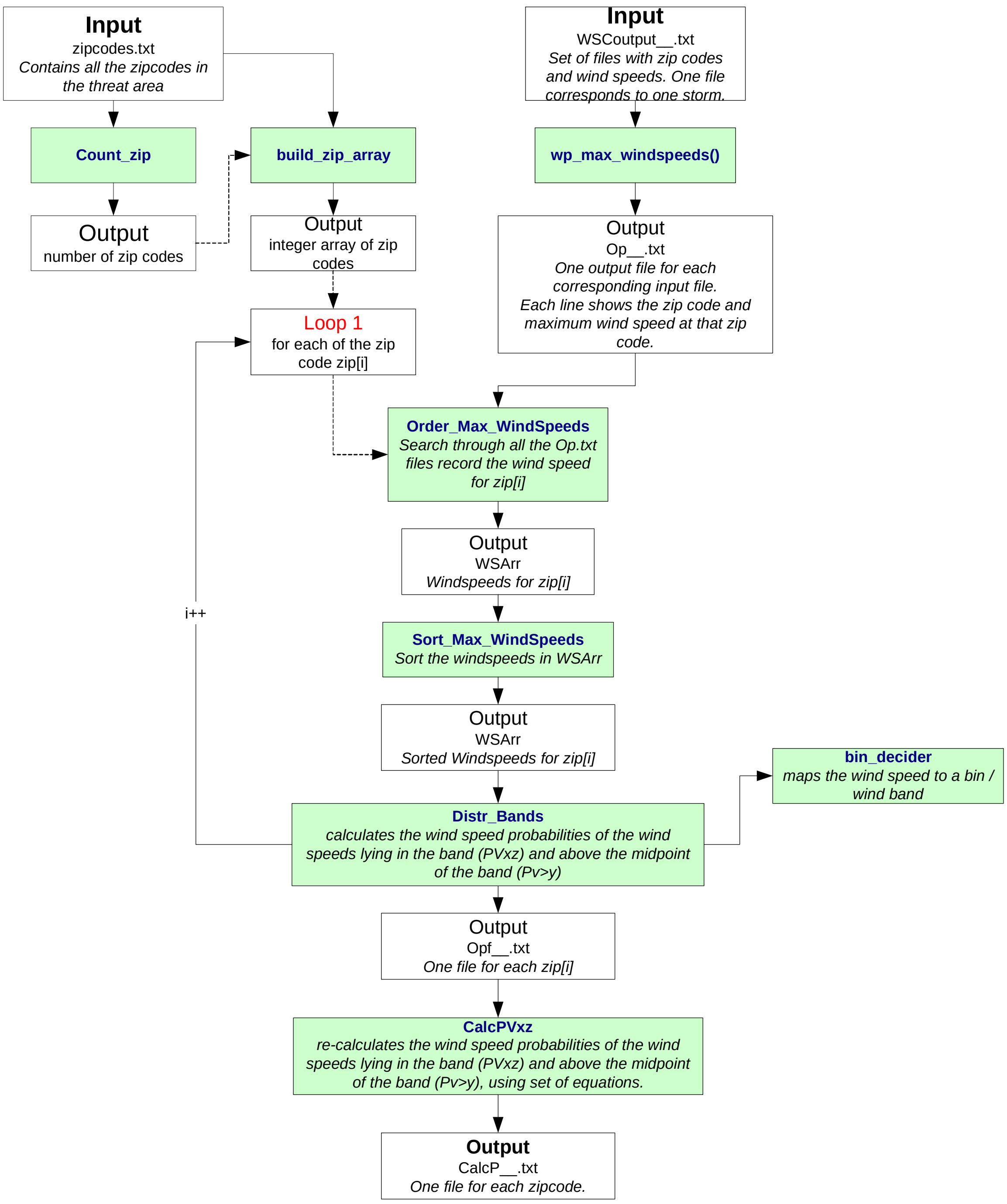

Input
zipcodes.txt
Contains all the zipcodes in the threat area
Count_zip
Output
number of zip codes
build_zip_array
Output
integer array of zip codes
Loop 1
for each of the zip code zip[i]
i++
Input
WSCoutput__.txt
Set of files with zip codes and wind speeds. One file corresponds to one storm.
wp_max_windspeeds()
Output
Op__.txt
One output file for each corresponding input file.
Each line shows the zip code and maximum wind speed at that zip code.
Order_Max_WindSpeeds
Search through all the Op.txt files record the wind speed for zip[i]
Output
WSArr
Windspeeds for zip[i]
Sort_Max_WindSpeeds
Sort the windspeeds in WSArr
Output
WSArr
Sorted Windspeeds for zip[i]
bin_decider
maps the wind speed to a bin / wind band
Distr_Bands
calculates the wind speed probabilities of the wind speeds lying in the band (PVxz) and above the midpoint of the band (Pv>y)
Output
Opf__.txt
One file for each zip[i]
CalcPVxz
re-calculates the wind speed probabilities of the wind speeds lying in the band (PVxz) and above the midpoint of the band (Pv>y), using set of equations.
Output
CalcP__.txt
One file for each zipcode.

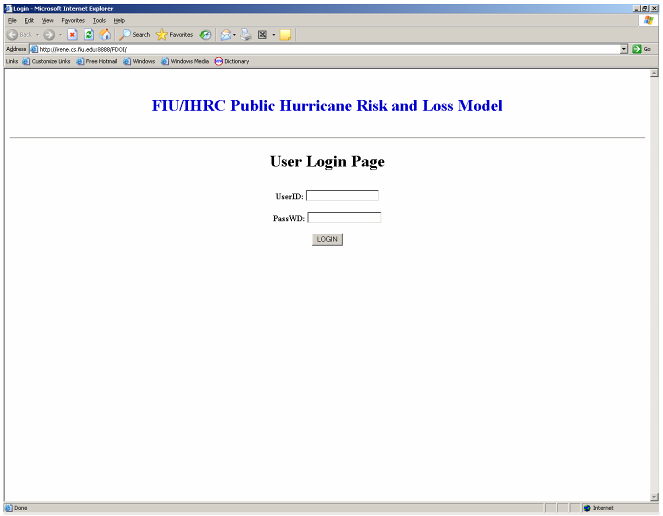
Figure 3.4.6: Unit flow diagram

3.4.4 Implementation of WSP

WSP is online at http://www.cs.fiu.edu/PHRLM.

3.4.4.1 Login page:

Users need a username and a password to access the *FIU/IHRC Public Hurricane Risk and Loss Model*. Following is a snapshot of the login web page.



Login - Microsoft Internet Explorer

Address http://irene.cs.fiu.edu:8888/FDOI/

FIU/IHRC Public Hurricane Risk and Loss Model

User Login Page

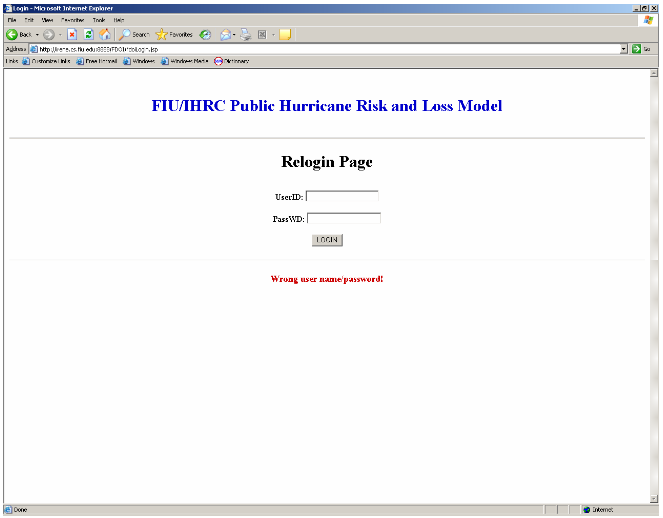
UserID:

PassWD:

File Edit View Favorites Tools Help
Back
Search
Favorites
Go
Links Customize Links Free Hotmail Windows Windows Media Dictionary
LOGIN
Done
Internet

Figure 3.4.7. Login webpage for FIU/IHRC PHRLM

If the username/password is wrong, an error message will be displayed. User will be required to input the username and password again to enter.



Login - Microsoft Internet Explorer

Address http://irene.cs.fiu.edu:8888/FDOI/fdoiLogin.jsp

FIU/IHRC Public Hurricane Risk and Loss Model

Relogin Page

UserID:

PassWD:

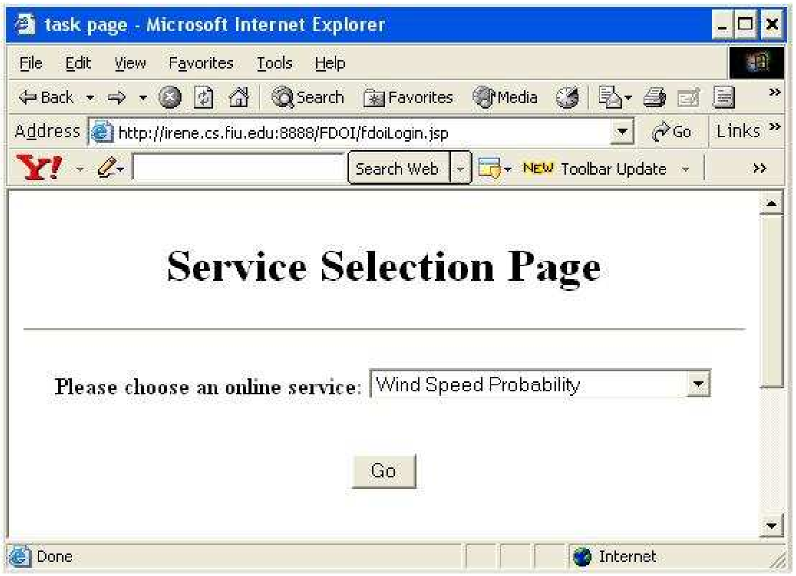
Wrong user name/password!

File Edit View Favorites Tools Help
Back
Search
Favorites
Go
Links Customize Links Free Hotmail Windows Windows Media Dictionary
LOGIN
Done
Internet

Figure 3.4.8. Login webpage shows the inputted user ID or password is wrong

3.4.4.2 WSP page:

If the login is successful, the user can see the web page named “Service Selection Page” (as shown in Figure 3.4.9). To view the WSP use case page, from the drop-down list, select “WSP” and click “Go” button.



task page - Microsoft Internet Explorer

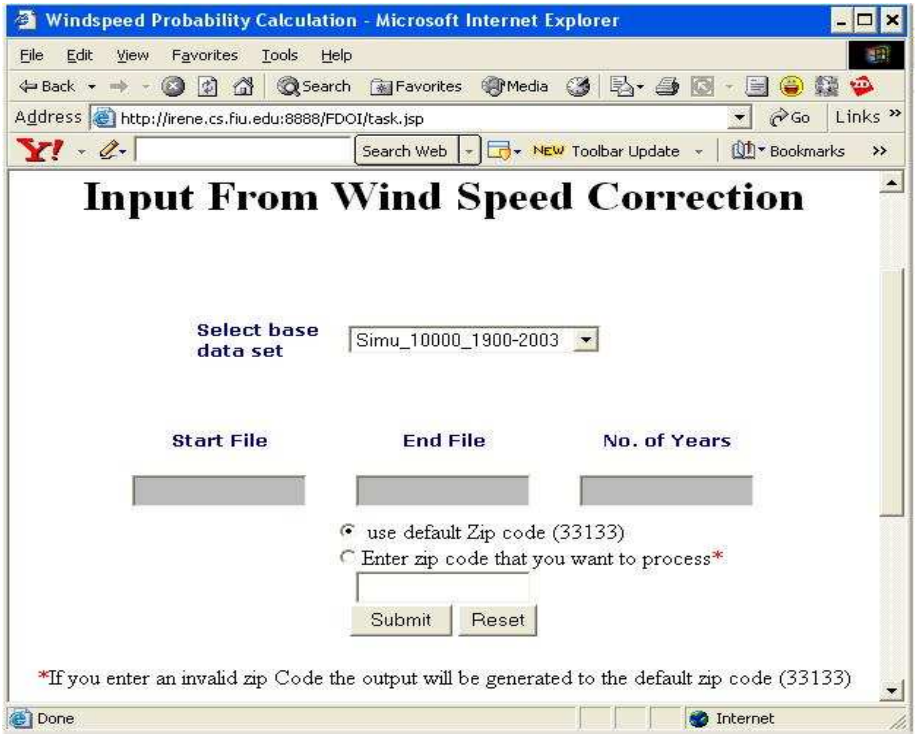
Address: http://irene.cs.fiu.edu:8888/FDOI/fdoiLogin.jsp

Service Selection Page

Please choose an online service: Wind Speed Probability

Figure 3.4.9: Service Selection Page

3.4.4.3 WSP input selection



Windspeed Probability Calculation - Microsoft Internet Explorer

Address: http://irene.cs.fiu.edu:8888/FDOI/task.jsp

Input From Wind Speed Correction

Select base data set: Simu_10000_1900-2003

Start File	End File	No. of Years

● use default Zip code (33133)

○ Enter zip code that you want to process*

*If you enter an invalid zip Code the output will be generated to the default zip code (33133)

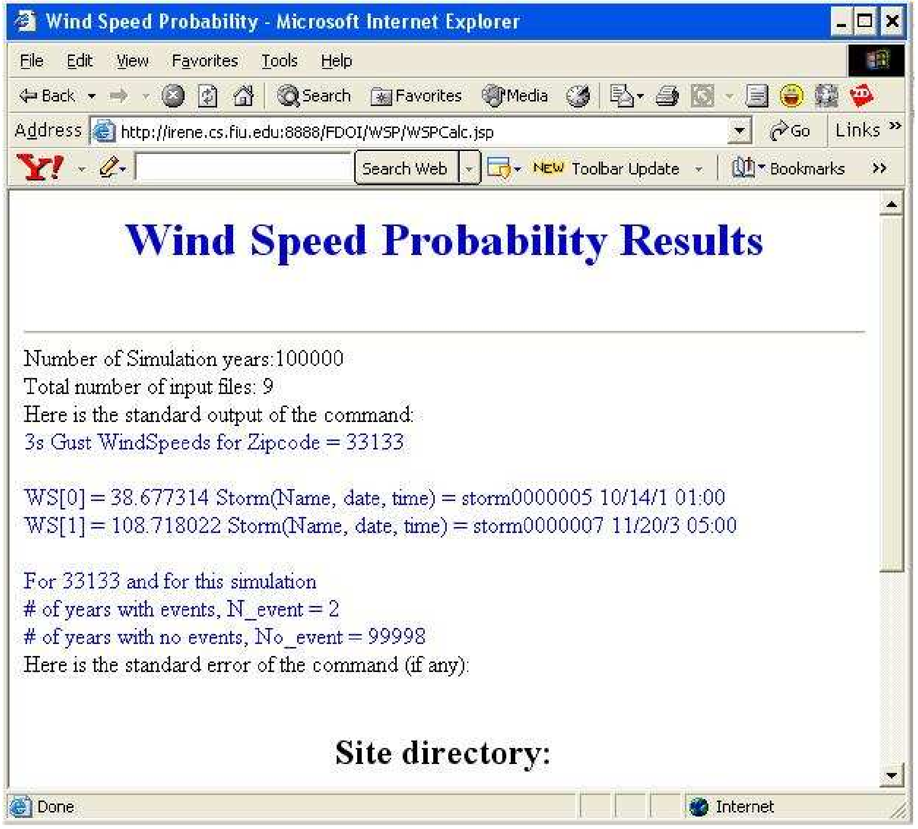
Figure 3.4.10: Input Selection page

User should select the input from the available list of simulations. Then the user has the option of entering a zip code in order to view a summery of results for that zip code. In order to do this, user has to click on the “Enter zip code that you want to process” radio button and then type the required zip code in the input box provided below. If the user doesn’t want to enter any specific zip code just press “Submit” button.

If the user selects the “Simu_other” option from the ‘select base data set’ drop down list then he/she has to enter the “start file” number, “end File” number and the number of years in the simulation. This option is provided for advanced users who may require to do test runs on arbitrary data sets.

3.4.4.4 WSP results page

Note: The results listed below are just placeholders. Actual values will be different.


Wind Speed Probability - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://irene.cs.fiu.edu:8888/FDOI/WSP/WSPCalc.jsp
Go Links
Search Web Toolbar Update Bookmarks
Done
Internet

Wind Speed Probability Results

```
Number of Simulation years:100000
Total number of input files: 9
Here is the standard output of the command:
3s Gust WindSpeeds for Zipcode = 33133

WS[0] = 38.677314 Storm(Name, date, time) = storm0000005 10/14/1 01:00
WS[1] = 108.718022 Storm(Name, date, time) = storm0000007 11/20/3 05:00

For 33133 and for this simulation
# of years with events, N_event = 2
# of years with no events, No_event = 99998
Here is the standard error of the command (if any):
```

Site directory:

Figure 3.4.11: WSP results page

3.4.4.5 Results of WSP

Results for zip code: 41

```
ZipCode = 41
Gust Band(mph)   Mid-Point   PVxz   Pv
> - =<
22.5- 27.5        25.0        0.017 0.262
27.5- 32.5        30.0        0.022 0.247
32.5- 37.5        35.0        0.022 0.229
37.5- 42.5        40.0        0.023 0.211
42.5- 47.5        45.0        0.022 0.192
47.5- 52.5        50.0        0.019 0.176
52.5- 57.5        55.0        0.017 0.160
57.5- 62.5        60.0        0.017 0.146
62.5- 67.5        65.0        0.017 0.131
67.5- 72.5        70.0        0.017 0.116
72.5- 77.5        75.0        0.017 0.102
77.5- 82.5        80.0        0.014 0.088
82.5- 87.5        85.0        0.014 0.075
87.5- 92.5        90.0        0.011 0.064
92.5- 97.5        95.0        0.010 0.054
97.5- 102.5      100.0        0.008 0.045
102.5- 107.5     105.0        0.008 0.037
107.5- 112.5     110.0        0.006 0.031
112.5- 117.5     115.0        0.005 0.025
117.5- 122.5     120.0        0.004 0.020
122.5- 127.5     125.0        0.004 0.016
127.5- 132.5     130.0        0.003 0.013
132.5- 137.5     135.0        0.002 0.010
137.5- 142.5     140.0        0.002 0.008
142.5- 147.5     145.0        0.002 0.006
147.5- 152.5     150.0        0.001 0.005
152.5- 157.5     155.0        0.001 0.003
157.5- 162.5     160.0        0.001 0.002
162.5- 167.5     165.0        0.000 0.002
167.5- 172.5     170.0        0.000 0.001
172.5- 177.5     175.0        0.000 0.001
177.5- 182.5     180.0        0.000 0.001
182.5- 187.5     185.0        0.000 0.000
187.5- 192.5     190.0        0.000 0.000
192.5- 197.5     195.0        0.000 0.000
197.5- 202.5     200.0        0.000 0.000
202.5- 207.5     205.0        0.000 0.000
207.5- 212.5     210.0        0.000 0.000
212.5- 217.5     215.0        0.000 0.000
217.5- 222.5     220.0        0.000 0.000
222.5- 227.5     225.0        0.000 0.000
227.5- 232.5     230.0        0.000 0.000
232.5- 237.5     235.0        0.000 0.000
237.5- 242.5     240.0        0.000 0.000
242.5- 247.5     245.0        0.000 0.000
247.5- 252.5     250.0        0.000 0.000
252.5- 257.5     255.0        0.000 0.000
257.5- 262.5     260.0        0.000 0.000
262.5- 267.5     265.0        0.000 0.000
```

```
267.5- 272.5     270.0        0.000 0.000
272.5- 277.5     275.0        0.000 0.000
277.5- 282.5     280.0        0.000 0.000
282.5- 287.5     285.0        0.000 0.000
287.5- 292.5     290.0        0.000 0.000
292.5- 297.5     295.0        0.000 0.000
297.5- 302.5     300.0        0.000 0.000
```

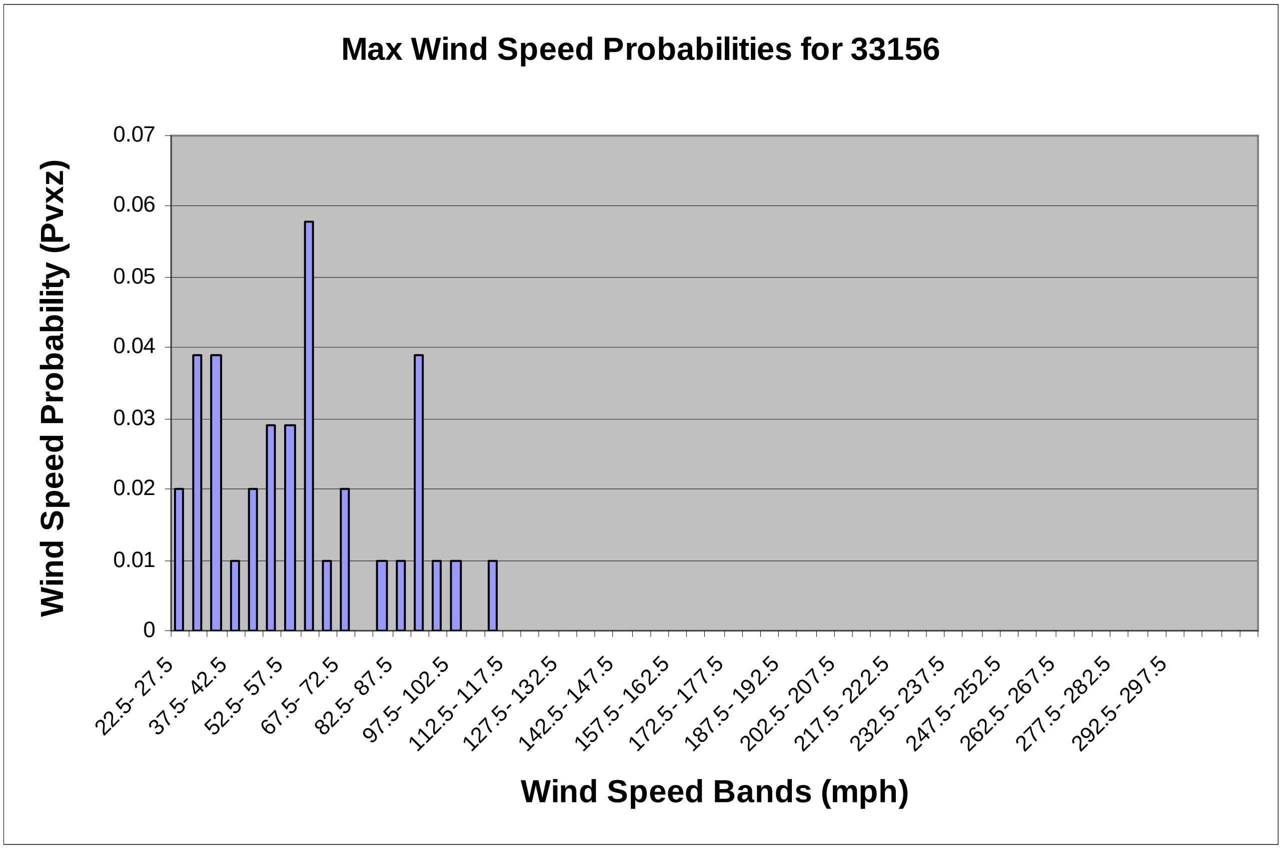


Figure 3.4.12: Histogram for PVxz

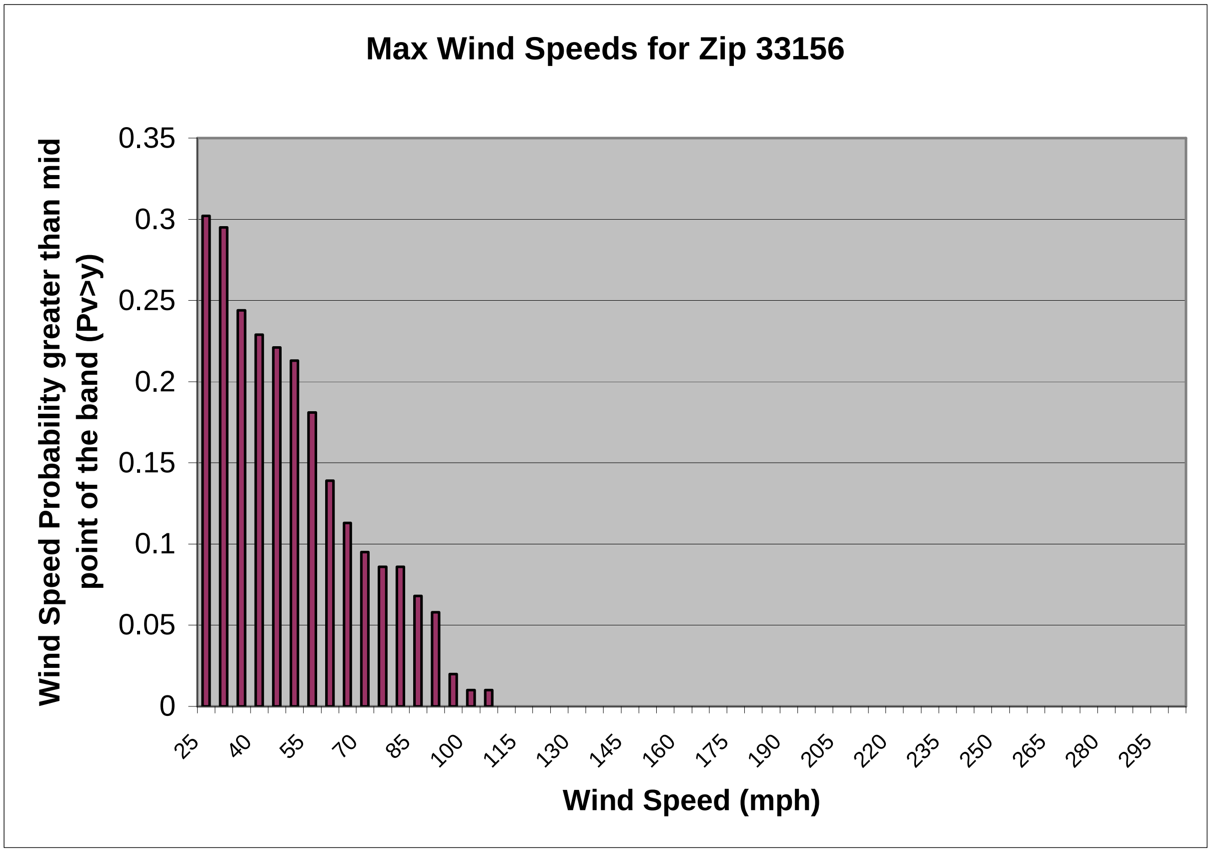


Figure 3.4.13: Histogram for PV>y

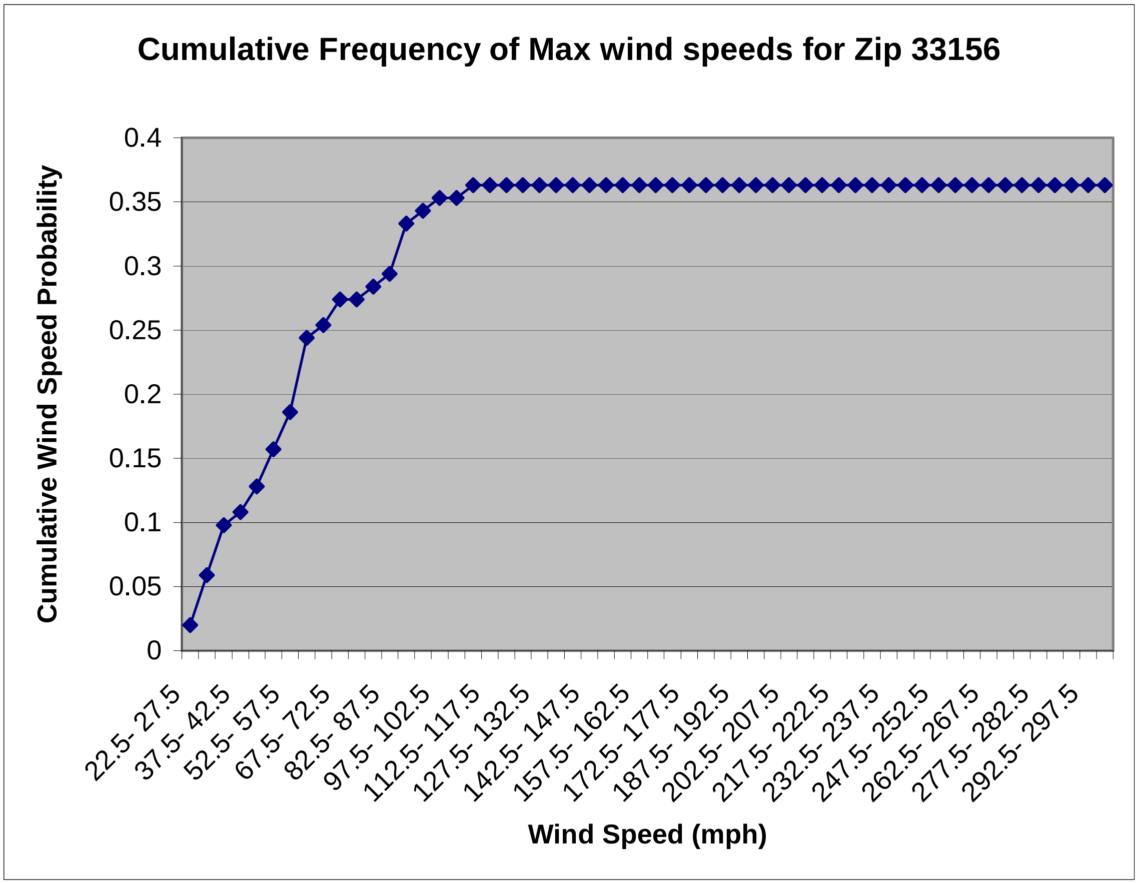


Figure 3.4.14: Cumulative frequency diagram for maximum wind speed probabilities for zip code 33156.

Section 4

Insurance Loss Model (ILM)
Use Case VII

Section 4.1

General Description of ILM

Insurance Loss Model (ILM) calculates the expected losses during storms. There are two variations of ILM; Scenario ILM and Probabilistic ILM.

Scenario ILM takes actual (observed) wind speed or modeled wind speed per each zip code and calculates the expected losses, using Vulnerability Matrices provided by the engineering team, per loss type given the input exposure/insurance policy data.

Probabilistic ILM on the other hand uses all possible wind speeds (from 50 to 250 mph) together with their associated wind probabilities per each zip code, using Vulnerability Matrices to calculate the expected losses.

4.1.1 Design Requirements

Name: Insurance Loss Model Use Case

Description: The traditional actuarial method in loss estimation is typically parametric and involves fitting some distribution to the number of claims (typically Poisson) and the amount of the losses. A variety of distributions are available for fitting losses (e.g. Lognormal, Weibull, Pareto, gamma, Burr, mixture of distributions), most with the preferred two parameters but a few with three parameters to be estimated. There are several techniques to estimate the parameters (percentile matching, method of moments, minimum distance, minimum chi-square, and maximum likelihood). The models are validated or accepted using both statistical and non-statistical criteria. If more than one model are acceptable then a ranking of the models is in order. Models can be ranked and selected by using, e.g., large maximum likelihood function value; small chi-square goodness of fit test statistics; small Kolmogorov-Smirnov test statistics; large p-value from chi-square goodness of fit test; minimum cdf, MSE or LAS etc[1]. Though not necessary, ranking is often done by using the same method employed for estimating parameters.

Once the loss distribution has been selected and its parameters estimated and validated it is rather easy to use and a variety of hypotheses can be tested. For example, for purposes of prediction, hundreds or thousands of sample outcomes of losses can be generated for a given representative insured property. Next, policy modifications such as deductibles and limits can be applied to each outcome to generate a set of net of deductible losses which are then averaged to generate expected loss. The losses can be aggregated. Exceedance probabilities curves can be generated to estimate the likelihood the portfolio of policies will suffer losses in excess of a given level. Alternatively, the Value at Risk (PML) can be estimated at a given exceedance probability.

In the above traditional practice, only the insurance policy files and claims data are used. Typically, the losses are modeled directly and are not derivative of other variables. In our project, however, the catastrophe modeling process requires that in addition to the

[1] [See (Hogg and Klugman, *Loss Distribution*, 1984, particularly Ch. 4 and 5 and the appendix) and (Klugman, Panjer and Willmot, *Loss Models*, 1998)] .

insurance data, output data from the meteorology and engineering components must be utilized. The distributions of losses are driven by both the distribution for damage ratios generated by the engineering component and by the distribution of wind speeds generated by the meteorology component. The wind speeds and damage ratios are estimated through extensive simulations. The engineering group has produced non-parametric damage matrices rather than the traditional continuous vulnerability functions. Damage ratios are grouped and intervals (or classes) of various length are used. No statistical distributions are fitted or tested.

The engineering component produces non-parametric estimates of damage probabilities for various intervals or classes of damage ratios for structures and contents. They do not fit any statistical distributions to the damage ratios. Thus, for the insured loss model a choice must be made to either fit a parametric statistical distribution for the damage intervals using some of the same standard techniques mentioned above but applied to grouped data, or to use a non-parametric technique presented as a broad algorithm.

The advantages and disadvantages of using parametric vs non-parametric techniques are well known. For our purpose the main advantage of the parametric technique would be computational efficiency. Once a statistical distribution has been fitted and its parameters estimated, it is relatively easy to estimate expected losses and formulas are available for estimating the mean and dispersion etc for many distributions in the presence of deductibles or limits (truncated data). Predictions can be made relatively easily if the distributions are stable. Computationally there are fewer steps involved. It is also easier to test hypothesis or to investigate the effect of, e.g., changes in deductibles and limits on expected losses. The major disadvantage is that with limited data, we may never be sure if the right distribution has been fitted and the errors in the estimated parameters can be significant.

In non-parametric estimation empirical functions are fitted to the data. There is no worry whether a correct distribution has been fitted and uncertainties are likely to be lower. The disadvantage is that a complex algorithm may be required that involves many steps and long computational time. Hypothesis testing is also more complicated and stability may be a concern.

For various reasons we have decided to pursue the non-parametric option initially. Given the large computing power available, computational time is not a major concern. Thus, it may be prudent to develop a logical non-parametric and deterministic algorithm that should produce low uncertainties. The broad algorithm utilized to estimate insured losses is discussed below.

Section 4.2

Detailed design and Implementation of Insurance Loss Model

Input:

Both the meteorology and engineering components provide outputs that constitute critical inputs into the insured loss model. The meteorology component provides, for each zip code, the associated probabilities for a common set of wind speeds. Thus, zip codes are essentially differentiated by their probability distribution of wind speeds.

The Engineering component produces damage matrices that are used as input in the insured loss model. Damage matrices are provided for building structure, contents, appurtenant structures and additional living expenses. A separate damage matrix is provided for each construction type. But within a certain range of building ages, a particular construction type will have the same damage matrix across all the zip codes in the same region. The cells of the matrix provide probabilities of damage outcomes for a given wind speed. The damages are specified in intervals or classes of ratios. The row represents a given interval of damage ratios and the column represent a given wind speed. In practice, the damage probabilities are assigned to the mid point of the interval of damage ratios. The probabilities of all possible damage outcomes must add up to 1. Therefore, the sum of the cells in any given vector column (for a wind speed) add up to 100%.

It should be noted that both the damages and wind speeds are initially specified as a set of discrete points. If needed one can interpolate to get a rough continuous function by using either some standard smoothing techniques (e.g. by defining the jump of the distribution function and using it with a kernel function and optimal bandwidth to estimate a smooth PDF) or by specifying an empirical set of ranges or intervals where each interval has an associated probability. The latter method is used by the engineering component and its output is specified as a set of damage ratio intervals with associated probabilities.

The third major set data utilized are the insurance policy and claim data provided by several property and casualty insurance companies operating in Florida.

4.2.1 ILM Implementation Steps

- **The Non-Parametric Algorithm for Generating Expected Loss Costs for a Given Exposure**

Here, the exposure data by zip code is the only given observed data. The wind probabilities, and damage matrices are all modeled. In practice to generate expected loss costs the method we adopt involves an algorithm with the following steps.

(1) Start with a particular insurance company m.

(2) Next pick a residential policy exposure unit k from the insurance policy database.

(3) Determine the zip code j of the policy.

(4) Extract the distribution of wind speeds for the zip code j from the wind database.

(5) Next determine the building type i and the building construction date d (if available) for the selected policy.

(6) Select the damage matrix for *structure* of type i based on its construction date d. If the construction date d is not available, another set of vulnerability matrix is used. The matrix is provided by the Engineering team and consists of the simulated probabilities for various damage ratio intervals and wind speeds. The row represents a given interval n of damage ratios and the column represent a given wind speed w. Each cell represents the probability P_{nw}. Let X_{ij} be the vector of the mid points of the interval of damage ratios for structure type i in zip code j. It has N elements. Now rather than use the MDR (Mean Damage Ratio) of the whole matrix, the mid point of the damage ratio interval n, X_n, is used to represent an outcome, and the probability of this outcome for a given wind speed is P_{nw}. In general, for structure i in zip code k, the mid point of damage intervals is X_{ijn} and its probability of outcome for a given wind speed is P_{ijnw}.

(7) Select the damage matrix for *contents* for structure of type i on its construction date d. If the construction date d is not available, another set of vulnerability matrix is used. The matrix is provided by the Engineering team and consists of the simulated probabilities for various content damage ratio intervals and wind speeds. The row represents a given interval n of content damage ratios and the column represent a given wind speed w. The interpretation of the cells values etc is similar to the description given above for structure damage matrix. Although the content damage depend indirectly on structural damage, there is no stipulated functional relationship between the two matrices and their damage intervals.

(8) Select the AP and ALE damage matrices accordingly. The Engineering team has generated independent matrices for AP and ALE based on indirect relationships between structural damage and both ALE and AP.

(9) From the insurance policy file, get the property value V_{ijk}, its policy limits LM_{ijk}, and its deductible D_{ijk}. The limit LM is the default value of the property k (default is V = LM) if value is not available. Value is contingent on the type of policy specified and is either replacement cost or actual cash value (replacement cost minus depreciation).

(10) Select a wind speed from the distribution. Apply the damage ratio vector X_{ij} to the property k (of type i in zip code j). For each damage interval n, calculate the \$ damage: $DM_{ijknw} = V_{ijk} \times X_{ijnw}$. Thus, a Nx1 \$ damage vector DM_{ijk} is generated for property k. This vector is associated with the chosen wind speed.

(11) For the above selected wind speed, estimate the row vector of wind conditional mean \$ content damages, where each element is the mean content damage for the given wind speed: $meanC_{ijkw} = LM_c \times meanC_{ijnw}$ ratio.

(12) For the selected wind speed, estimate the row vector of wind conditional mean \$ AP damages, where each element is the mean AP damage for the given wind speed: $meanAP_{ijkw} = LM_{AP} \times meanAP_{ijnw}$ ratio.

(13) For the selected wind speed, estimate the row vector of wind conditional mean \$ ALE damages, where each element is the mean ALE damage for the given wind speed: $meanALE_{ijkw} = LM_{ALE} \times meanALE_{ijnw}$ ratio.

(14) Using the wind conditional mean \$ structural damage DM_{ijk}, and combining it with the wind conditional mean C, mean AP_{ijkw} and mean ALE_{ijkw}: calculate the deductibles D_S, D_C, D_{AP} on a pro-rata basis to the respective damages as follows:

$$D_s = \left[DM_S / \left(DM_S + C + AP\right)\right] \times D$$
$$D_c = \left[C / \left(DM_s + C + AP\right)\right] \times D$$
$$D_{AP} = \left[AP / \left(DM_s + C + AP\right)\right] \times D$$
$$D_{ALE} = 0$$

(15) Apply the pro-rata structure deductible D_{sijk} and limits LM_{ijk} to each of the cells of the \$ damage Matrix DM_{ijk}. Calculate the structure loss L_{sijkn} net of deductible, and truncate it on the upside by LM_{ijk} and on the downside by D_{sijk}. Thus, a vector L_{sijk} of insured losses is generated for property k. Its elements are L_{sijkn}**.** If L_{sijkn} is ≥ L_{mijk}, then $L_{sijkn} = L_{mijk}$. If L_{sijkn} is ≤ 0, then let $L_{sijkn} = 0$.

(16) Repeat step (15) for C, AP, and ALE. Here, these variables are means conditional on the wind speed. Generate L_c, L_{AP}, and L_{ALE}.

(17) Next, to get the *expected insured loss for a given wind speed* w, multiply each element L_{ijkn} of the vector L_{ijk} by its corresponding probability P_{ijkwn} to compute L_{ijknw}, and then sum over the N intervals. Steps 15 - 17 can be represented by:

$$\text{Expected Structure Loss} = \text{E (L}_{s}) = \sum_{Ds}^{L+Ds}(DM_i - D_s)P_s + \sum LM_S P_S$$

$$\text{Expected Content Loss} = \text{E (L}_{c}) = \sum (C - D_c)P_c + \sum LM_c P_c$$

$$\text{Expected Appurtenant Loss} = \text{E (L}_{AP}) = \sum (AP - D_{AP})P_{AP} + \sum LM_{AP} P_{AP}$$

$$\text{Expected ALE Loss} = \text{E (L}_{ALE}) = \sum (ALE - D_{ALE})P_{ALE} + \sum LM_{ALE} P_{ALE}$$

where L_{ijkwn} = LM_{ijk} if (DM_{ijn} - D_{sijk}) ≥ L_{mijk}, and if (DM_{ijn} - D_{Sijk}) ≤ 0, then let DM_{ijn} - D_{sijk} = 0 , i.e replace negative values of net of deductible loss with zero. The same applies to C, AP, and ALE.

(18) Expected Loss $E(L) = E(L_S) + E(L_C) + E(L_{AP}) + E(L_{ALE})$

(19) Repeat step (10) through (18) for all the wind speeds to generate a row of *expected insured loss for all wind speeds*.

(20) Multiply the Expected Loss E(L_{ijkw}) for a given wind speed by the probability of the wind speed, p_w. Next sum over all wind speeds to get the property k *Expected Loss*:

$$\text{Property k Expected Loss} \quad E(L_{ijk}) = \sum_{w}^{W}\left[E(L_{ijkw} \times P_w)\right]$$

(21) Steps (7) through (20) are repeated for all dwellings of type i in zip code j to generate E (L_{ijk}) for all properties k =1,...,K.

(22) The expected losses are then summed to get the Expected Aggregate Loss for property type i in zip code j:

$$ExpectedAggregateLoss = E(Lij) = \sum^{K} E(L_{ijk})$$

(23) Variance will now need to be computed empirically, since all the terms in the calculations for the expected losses are correlated. We compute the variance as follows:
The variance for all dwellings of type i in zip code j will be:

$$\sigma_{ij}^2 = \{1/(K-1)\}\left\{\sum_{k=1}^{K}(L_{ijk})^2 - (1/k)\left(\sum_{k=1}^{K}(L_{ijk})\right)^2\right\}$$

(24) To get the Expected Loss (mean loss) for structure type i in zip code j, the E(AL) is calculated as the weighted average of the Expected Loss of all properties of type i. The weight is the relative value of the structure $\text{V}_{\text{ijk}} \big/ \sum \text{V}_{\text{ijk}}$ (or relative exposure of the structure $\text{LM}_{\text{ijk}} \big/ \sum \text{LM}_{\text{ijk}}$):

Expected Loss for property type i in zip code j $= \sum_{k=1} \left(v_{ijk} \Big/ \sum v_{ijk} \right) \mathrm{L}_{\mathrm{ijk}}$

(25) To estimate the expected loss as a percentage of exposure for structure type i in zip code j, use:

$$\%E\left(L_{ij}\right) = \left\{ \sum_{k=1} L_{ijk} \Big/ \sum_{k=1} LM_{ijk} \right\}$$

(26) Repeat steps (5) through (23) for all property types i = 1, ..., I to get the Expected Aggregate Loss and Expected loss for all property types in zip code j.

(27) Sum the E(AL_{ij}) across all property types i to get the Expected Aggregate Loss for all exposure in zip code j:

$$E\left(AL_j\right) = \sum_{i=1}^{I} E\left(AL_{ij}\right)$$

(28) Sum σ_{ij}^2 to get σ_j^2, the variance for all exposure in zip code j.

(29) Pick another zip code and repeat steps (4) through (28) to generate E (AL_j) for all zip codes. Sum across the zip codes j=1,....,J to get the Expected Aggregate Loss for insurance company m.

E (AL_m) for company $m = \sum E\left(AL_j\right) \quad j = 1,....J$

(30) Sum σ_j^2 to get σ_m^2, the variance for insurance company m.

(31) Pick another insurance company m and repeat steps (1) through (30). Sum across the insurance companies to get the Overall Expected Loss.

- **The Non-Parametric Algorithm for Generating Scenario Based Expected Loss Costs**

In this section we develop the algorithm for estimating expected loss costs for a given scenario. Typically the scenario refers to a particular hurricane with a given set of characteristics. *Hence, both the exposure data and the wind speeds by zip code are given observed data. The damage matrices, as before, are modeled*. Most of the steps in this algorithm are the same as in the prior section.

(1) Start with a particular insurance company m.

(2) Next pick a residential policy exposure unit k from the insurance policy database.

(3) Determine the zip code j of the policy.

(4) Extract the distribution of wind speeds for the zip code j from the wind database.

(5) Next determine the building type i and the building construction date d (if available) for the selected policy.

(6) Select the damage matrix for *structure* of type i based on its construction date d. If the construction date d is not available, another set of vulnerability matrix is used. The matrix is provided by the Engineering team and consists of the simulated probabilities for various damage ratio intervals and wind speeds. The row represents a given interval n of damage ratios and the column represent a given wind speed w. Note that in the **scenario analysis the observed wind speeds are used**. Thus, only the column corresponding to the observed wind speed for the zip code is used. Let X_{ij} be the vector of the mid points of the interval of damage ratios for structure type i in zip code j. It has N elements. Now rather than use the MDR (Mean Damage Ratio) of the whole matrix, the mid point of the damage ratio interval n, X_n, is used to represent an outcome, and the probability of this outcome for a given observed wind speed is P_{nw}. In general, for structure i in zip code k, the mid point of damage intervals is X_{ijn} and its probability of outcome for a given observed wind speed is P_{ijnw}.

(7) Select the damage matrix for *contents* for structure of type i based on its construction date d. If the construction date d is not available, another set of vulnerability matrix is used. The matrix is provided by the Engineering team and consists of the simulated probabilities for various content damage ratio intervals and wind speeds. The row represents a given interval n of content damage ratios and the column represent a given wind speed w. The interpretation of the cells values etc is similar to the description given above for structure damage matrix. Although the content damage depends indirectly on structural damage, there is no stipulated functional relationship between the two matrices and their damage intervals.

(8) Select the AP and ALE damage matrices accordingly. The Engineering team has generated independent matrices for AP and ALE based on indirect relationships between structural damage and both ALE and AP.

(9) From the insurance policy file, get the property value V_{ijk}, its policy limits LM_{ijk}, and its deductible $D_{ijk.}$. The limit LM is the default value of the property k (default is V = LM) if value is not available. Value is contingent on the type of policy specified and is either replacement cost or actual cash value (replacement cost minus depreciation).

(10) Select the damage vector for the observed wind speed. Apply the damage ratio vector X_{ij} to the property k (of type i in zip code j). For each damage interval n, calculate the \$ damage: $DM_{ijknw} = V_{ijk} \times X_{ijnw}$. Thus, a Nx1 \$ damage vector DM_{ijk} is generated for property k. This vector is associated with the observed wind speed.

(11) For the observed wind speed, estimate the row vector of wind conditional mean \$ content damages, where each element is the mean content damage for the given wind speed: $\text{mean}\, C_{ijkw} = LM_C \times \text{mean}\, C_{ijnw}$ ratio.

(12) For the observed wind speed, estimate the row vector of wind conditional mean \$ AP damages, where each element is the mean AP damage for the given wind speed: $\text{mean AP}_{\text{ijkw}} = \text{LM}_{\text{AP}} \times \text{mean AP}_{\text{ijnw}}$ ratio.

(13) For the observed wind speed, estimate the row vector of wind conditional mean \$ ALE damages, where each element is the mean ALE damage for the given wind speed: $\text{mean ALE}_{\text{ijkw}} = \text{LM}_{\text{ALE}} \times \text{mean ALE}_{\text{ijnw}}$ ratio.

(14) Using the wind conditional mean \$ structural damage DM_{ijk}, and combining it with the wind conditional mean C, mean AP_{ijkw} and mean ALE_{ijkw}: calculate the deductibles D_S, D_C, D_{AP}, D_{ALE} on a pro-rata basis to the respective damages as follows:

$$D_s = [DM_S/(DM_S + C + AP)] \times D$$
$$D_c = [C/(DM_s + C + AP)] \times D$$
$$D_{AP} = [AP/(DM_s + C + AP)] \times D$$
$$D_{ALE} = 0$$

(15) Apply the pro-rata structure deductible D_{sijk} and limits LM_{ijk} to each of the cells of the \$ damage Matrix DM_{ijk} . Calculate the structure loss L_{sijkn} net of deductible, and truncate it on the upside by LM_{ijk} and on the downside by D_{sijk}. Thus, a vector L_{sijk} of insured losses is generated for property k. Its elements are L_{sijkn}. If L_{sijkn} is ≥ L_{mijk}, then $L_{sijkn} = L_{mijk}$. If L_{sijkn} is ≤ 0, then let $L_{sijkn} = 0$.

(16) Repeat step (15) for C, AP, and ALE. Here, these variables are means conditional on the wind speed. Generate L_c, L_{AP}, and L_{ALE} .

(17) Next, to get the *expected insured loss for the observed wind speed* w, multiply each element L_{ijkn} of the vector L_{ijk} by its corresponding probability P_{ijkwn} to compute L_{ijknw}, and then sum over the N intervals. Steps 15 - 17 can be represented by:

$$\text{Expected Structure Loss} = \text{E (L}_s) = \sum_{\text{Ds}}^{\text{L+Ds}} (DM_i - D_s)P_s + \sum LM_S P_S$$

$$\text{Expected Content Loss} = \text{E (L}_c) = \sum (C - D_c)P_c + \sum LM_c P_c$$

$$\text{Expected Appurtenant Loss} = \text{E (L}_{AP}) = \sum (AP - D_{AP})P_{AP} + \sum LM_{AP} P_{AP}$$

$$\text{Expected ALE Loss} = \text{E (L}_{ALE}) = \sum (ALE - D_{ALE})P_{ALE} + \sum LM_{ALE} P_{ALE}$$

where $L_{ijkwn} = LM_{ijk}$ if $(DM_{ijn} - D_{sijk}) \geq L_{mijk}$, and if ($DM_{ijn} - D_{Sijk}$) ≤ 0, then let $DM_{ijn} - D_{sijk} = 0$, i.e replace negative values of net of deductible loss with zero. The same applies to C, AP, and ALE.

(18) $ExpectedLoss = E\left(L_{ijk}\right) = E\left(L_S\right) + E\left(L_C\right) + E\left(L_{AP}\right) + E\left(L_{ALE}\right)$ for property k

(19) Steps (7) through (18) are repeated for all dwellings of type i in zip code j to generate E (L_{ijk}) for all properties k =1,...,K.

(20) The expected losses are then summed to get the Expected Aggregate Loss for property type i in zip code j:

$$\text{Expected Aggregate Loss} = \mathrm{E}\left(\mathrm{L}_{ij}\right) = \sum^{K} \mathrm{E}\left(\mathrm{L}_{ijk}\right)$$

(21) Variance will now need to be computed empirically, since all the terms in the calculations for the expected losses are correlated. We compute the variance as follows:
The variance for all dwellings of type i in zip code j will be:

$$\sigma_{ij}^2 = \{1/(K-1)\}\left\{\sum_{k=1}^{K}\left(L_{ijk}\right)^2 - (1/k)\left(\sum_{k=1}^{K}\left(L_{ijk}\right)\right)^2\right\}$$

(22) To get the Expected Loss (mean loss) for structure type i in zip code j, the E(AL) is calculated as the weighted average of the Expected Loss of all properties of type i. The weight is the relative value of the structure $\mathrm{V}_{\mathrm{ijk}} \big/ \sum \mathrm{V}_{\mathrm{ijk}}$ (or relative exposure of the structure $\mathrm{L}M_{\mathrm{ijk}} \big/ \sum \mathrm{L}M_{\mathrm{ijk}}$
Expected Loss for property type i in zip code $j = \sum_{k=1}\left(v_{ijk} \big/ \sum v_{ijk}\right)\mathrm{L}_{\mathrm{ijk}}$

(23) To estimate the expected loss as a percentage of exposure for structure type i in zip code j, use:

$$\%E\left(L_{ij}\right) = \left\{\sum_{k=1} L_{ijk} \Big/ \sum_{k=1} LM_{ijk}\right\}$$

(24) Repeat steps (5) through (21) for all property types i = 1,....,I to get the Expected Aggregate Loss and Expected loss for all property types in zip code j.

(25) Sum the E(AL_{ij}) across all property types i to get the Expected Aggregate Loss for all exposure in zip code j:

$$E\left(AL_j\right) = \sum_{i=1}^{I} E\left(AL_{ij}\right)$$

(26) Sum σ_{ij}^2 to get σ_j^2 , the variance for all exposure in zip code j.

(27) Pick another zip code and repeat steps (4) through (26) to generate E (AL_j) for all zip codes. Sum across the zip codes j=1,....,J to get the Expected Aggregate Loss for insurance company m.

E (AL_m) for company m $= \sum \left(AL_j \right) \quad j = 1, \ldots . J$

(28) Sum σ_j^2 to get σ_m^2 , the variance for insurance company m.

(29) Pick another insurance company m and repeat steps (1) through (28). Sum across the insurance companies to get the Overall Expected Loss.

Section 4.3

Computer Model Design

4.3.1 Use case View of Insurance Loss Model (ILM)

A. Actors:

There is one actor (engineers) in ILM. Engineers use this use case to find the expected losses for particular companies for all wind speeds.

B. Use Case:

It represents the expected losses for particular companies for given (scenario or non scenario) wind speeds. The total expected loss is actually the summation of expected loss of the property for a given wind speed, which is calculated by aggregating the losses at different intervals with respect to the corresponding damage probabilities.

C. Use Case Diagram:

Figure 4.1.1 shows the use case diagram for ILM.

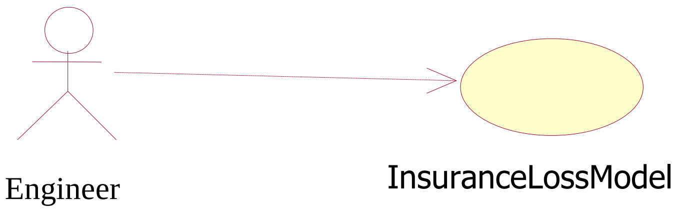


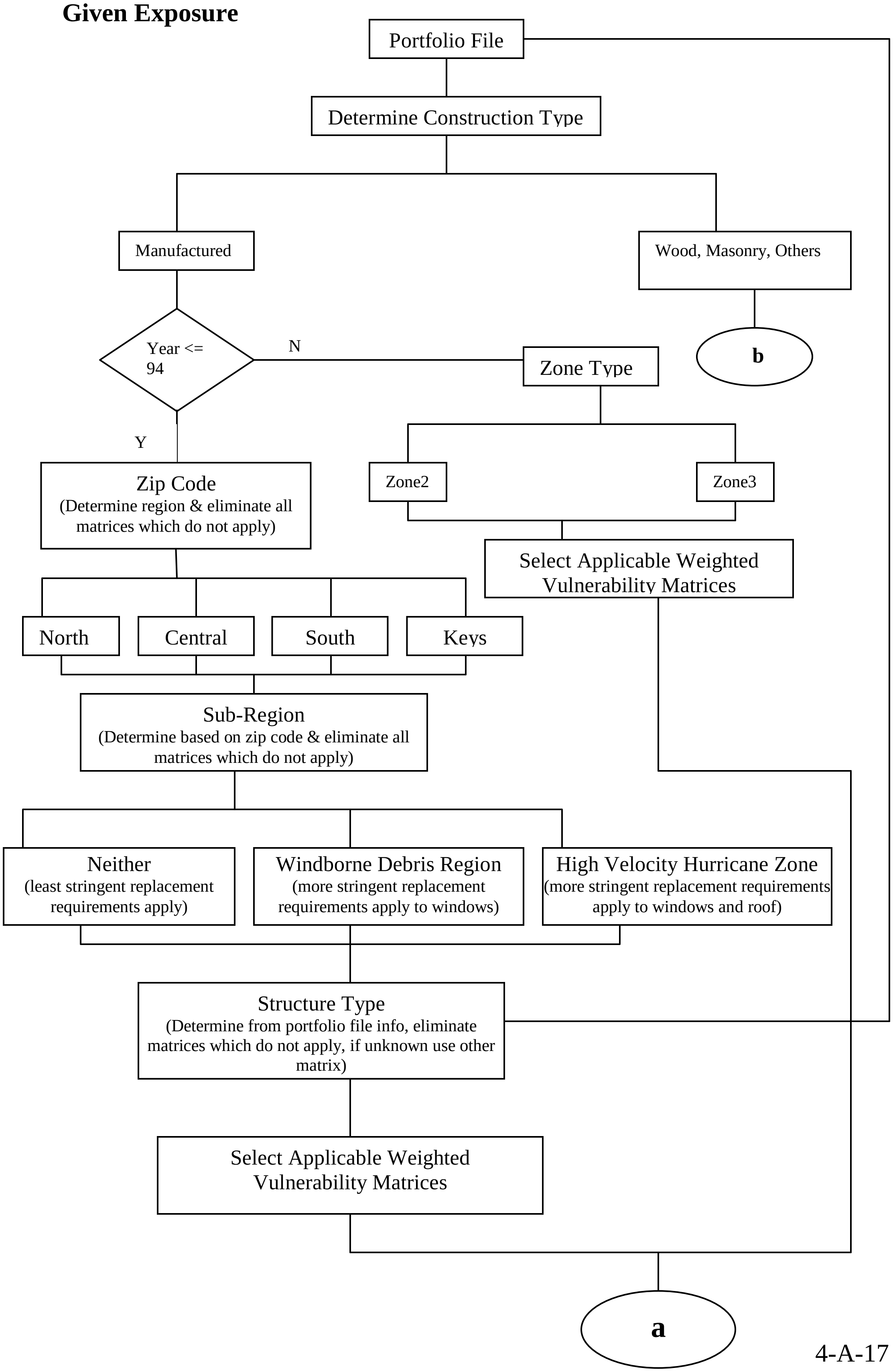
Figure 4.1.1: Use Case Diagram For ILM

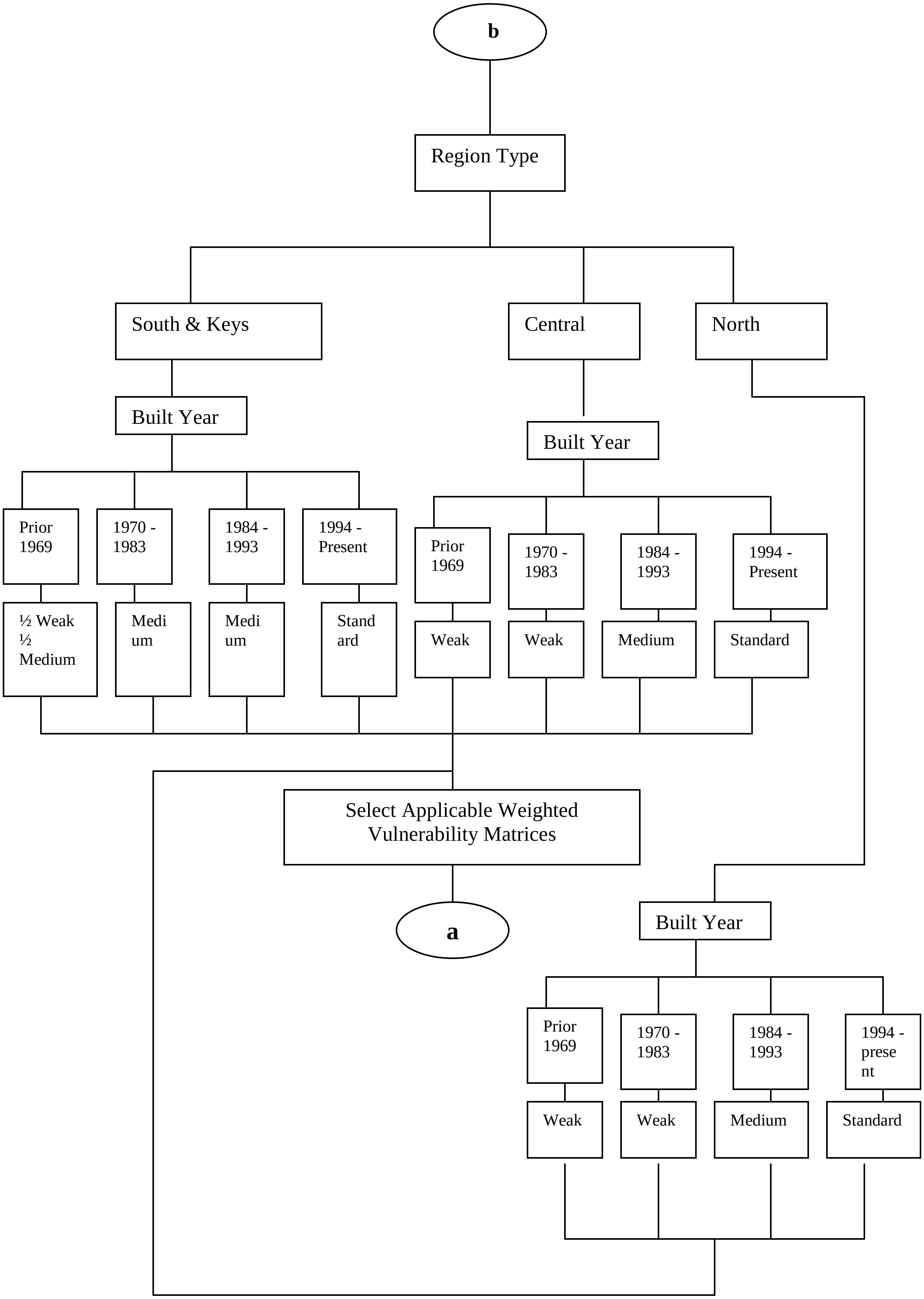
4.3.2 System Design

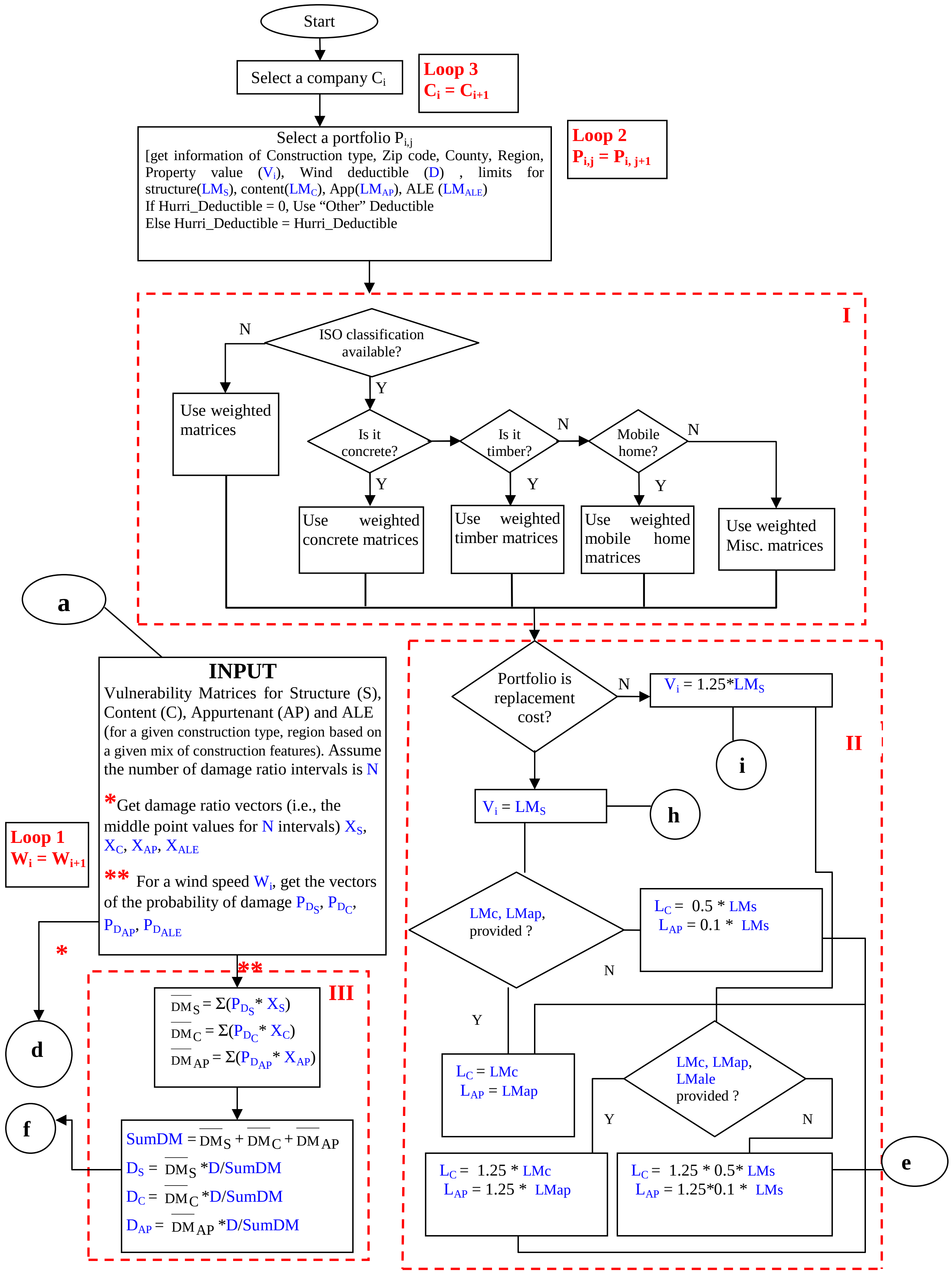
This portion describes the system design. The overall Flowchart, Classes and activities for ILM is provided.

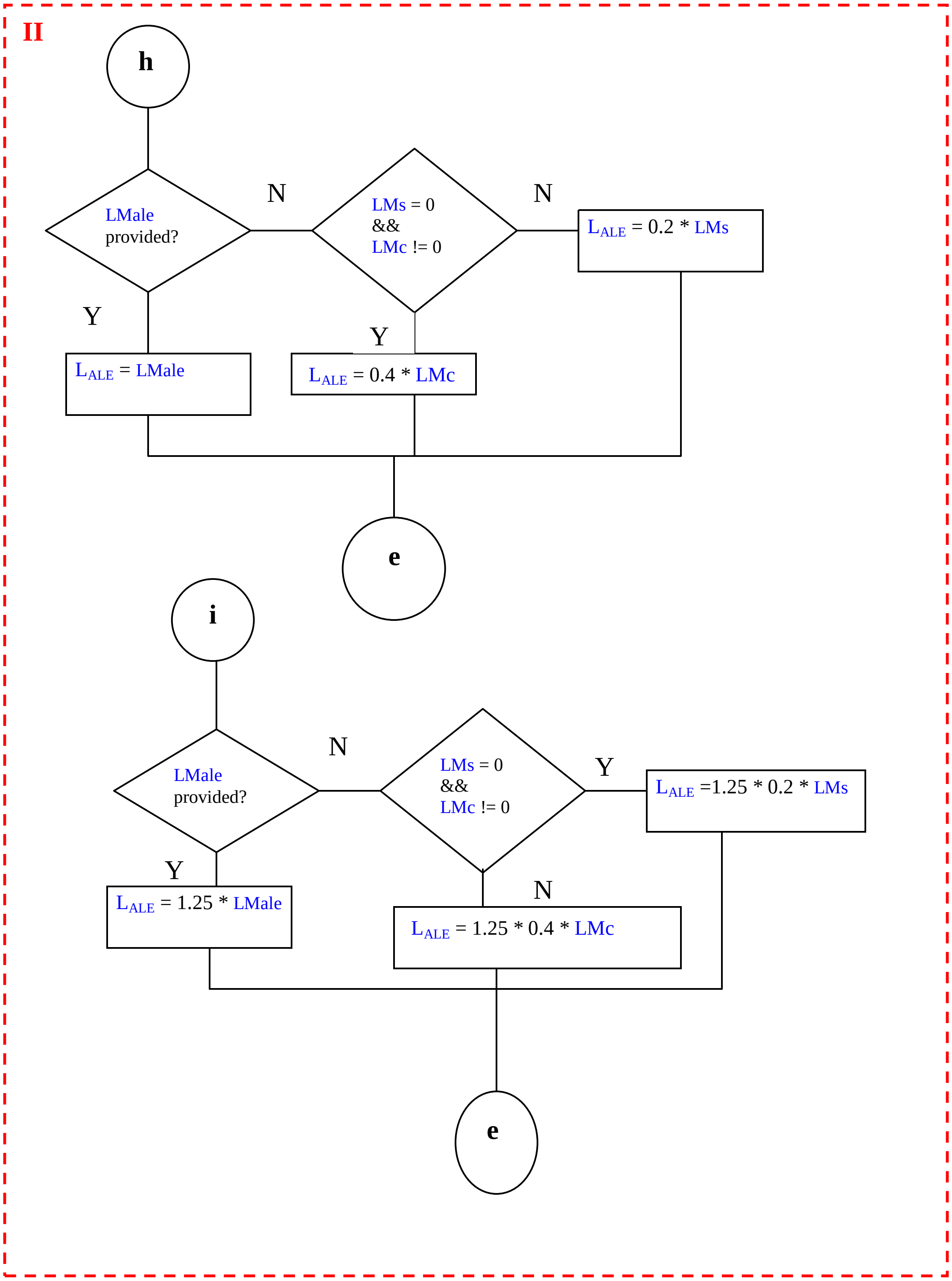
A. Program Flowchart of ILM

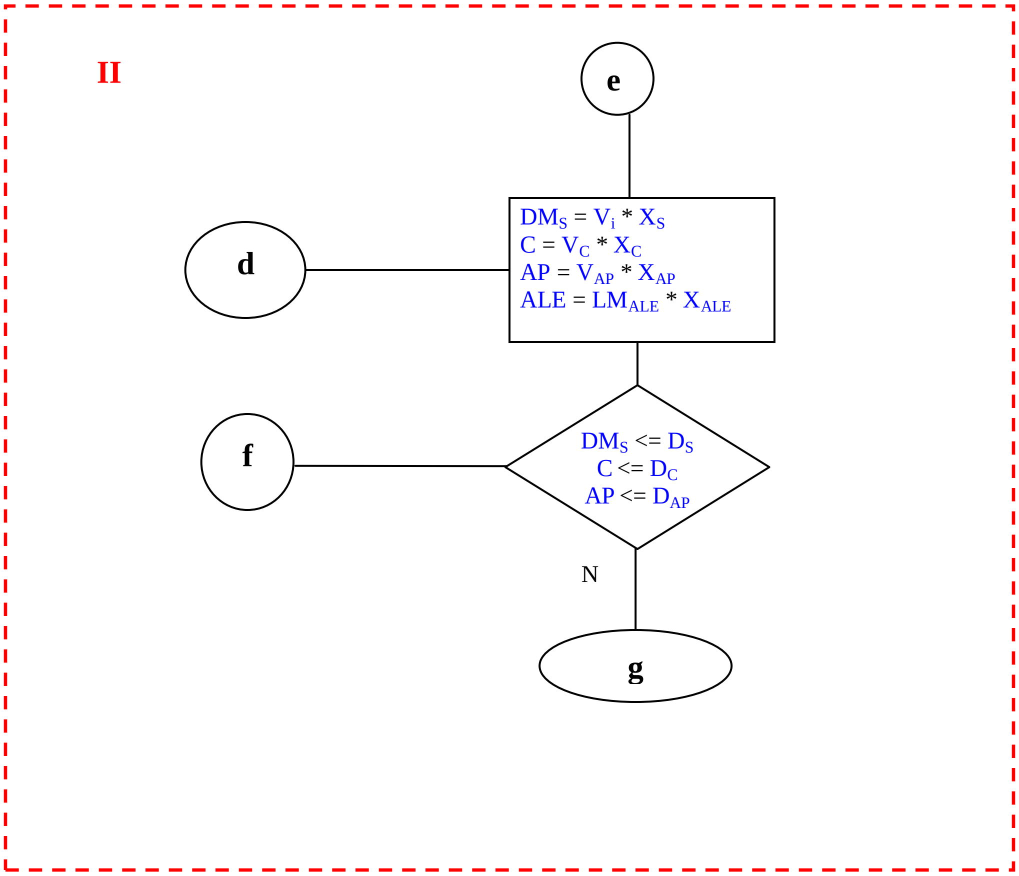
- **The Non-Parametric Approach for Generating Expected Loss Costs for a Given Exposure**

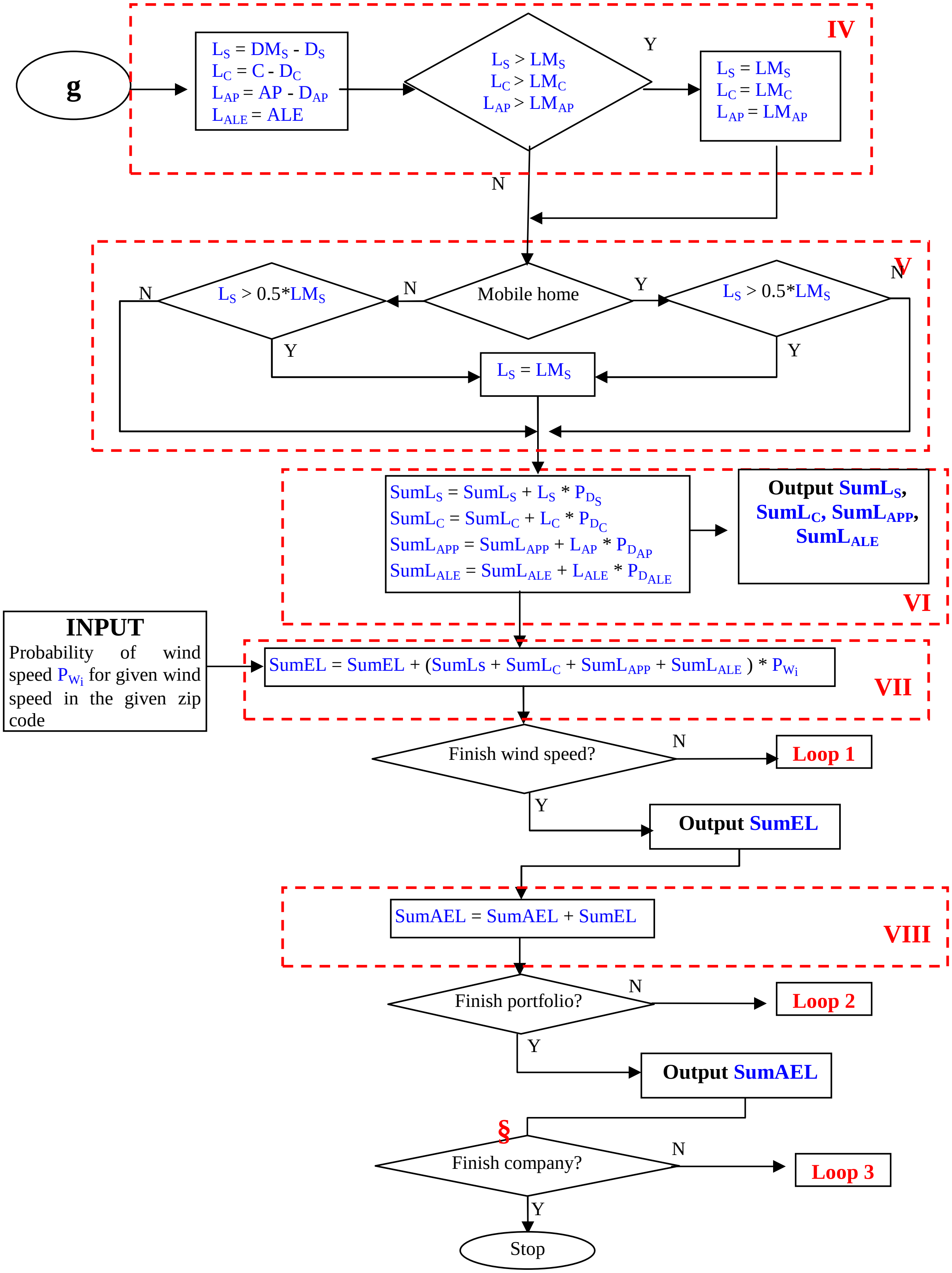

Portfolio File
Determine Construction Type
Manufactured
Wood, Masonry, Others
b
Year <= 94
N
Zone Type
Y
Zip Code
(Determine region & eliminate all matrices which do not apply)
Zone2
Zone3
Select Applicable Weighted Vulnerability Matrices
North
Central
South
Keys
Sub-Region
(Determine based on zip code & eliminate all matrices which do not apply)
Neither
(least stringent replacement requirements apply)
Windborne Debris Region
(more stringent replacement requirements apply to windows)
High Velocity Hurricane Zone
(more stringent replacement requirements apply to windows and roof)
Structure Type
(Determine from portfolio file info, eliminate matrices which do not apply, if unknown use other matrix)
Select Applicable Weighted Vulnerability Matrices
a


b
Region Type
South & Keys
Central
North
Built Year
Prior 1969
1970 - 1983
1984 - 1993
1994 - Present
½ Weak ½ Medium
Medi um
Medi um
Stand ard
Built Year
Prior 1969
1970 - 1983
1984 - 1993
1994 - Present
Weak
Weak
Medium
Standard
Select Applicable Weighted Vulnerability Matrices
a
Built Year
Prior 1969
1970 - 1983
1984 - 1993
1994 - prese nt
Weak
Weak
Medium
Standard


Start
Select a company C_i
Loop 3
$C_i = C_{i+1}$
Select a portfolio $P_{i,j}$
[get information of Construction type, Zip code, County, Region, Property value (V_i), Wind deductible (D) , limits for structure(LM_S), content(LM_C), App(LM_{AP}), ALE (LM_{ALE})
If Hurri_Deductible = 0, Use "Other" Deductible
Else Hurri_Deductible = Hurri_Deductible
Loop 2
$P_{i,j} = P_{i, j+1}$
I
ISO classification available?
N
Y
Use weighted matrices
Is it concrete?
Y
Use weighted concrete matrices
Is it timber?
N
Y
Use weighted timber matrices
Mobile home?
N
Y
Use weighted mobile home matrices
Use weighted Misc. matrices
a
INPUT
Vulnerability Matrices for Structure (S), Content (C), Appurtenant (AP) and ALE (for a given construction type, region based on a given mix of construction features). Assume the number of damage ratio intervals is N
*Get damage ratio vectors (i.e., the middle point values for N intervals) X_S, X_C, X_{AP}, X_{ALE}
** For a wind speed W_i, get the vectors of the probability of damage P_{D_S}, P_{D_C}, $P_{D_{AP}}$, $P_{D_{ALE}}$
Loop 1
$W_i = W_{i+1}$
*
**
d
III
$\overline{DM}_S = \Sigma(P_{D_S} * X_S)$
$\overline{DM}_C = \Sigma(P_{D_C} * X_C)$
$\overline{DM}_{AP} = \Sigma(P_{D_{AP}} * X_{AP})$
f
$SumDM = \overline{DM}_S + \overline{DM}_C + \overline{DM}_{AP}$
$D_S = \overline{DM}_S *D/SumDM$
$D_C = \overline{DM}_C *D/SumDM$
$D_{AP} = \overline{DM}_{AP} *D/SumDM$
II
Portfolio is replacement cost?
N
$V_i = 1.25*LM_S$
i
$V_i = LM_S$
h
LMc, LMap, provided ?
$L_C = 0.5 * LMs$
$L_{AP} = 0.1 * LMs$
N
Y
$L_C = LMc$
$L_{AP} = LMap$
LMc, LMap, LMale provided ?
Y
N
$L_C = 1.25 * LMc$
$L_{AP} = 1.25 * LMap$
$L_C = 1.25 * 0.5* LMs$
$L_{AP} = 1.25*0.1 * LMs$
e


II
h
LMale provided?
N
Y
LMs = 0 && LMc != 0
N
Y
L_{ALE} = 0.2 * LMs
L_{ALE} = LMale
L_{ALE} = 0.4 * LMc
e
i
LMale provided?
N
Y
LMs = 0 && LMc != 0
Y
N
L_{ALE} =1.25 * 0.2 * LMs
L_{ALE} = 1.25 * LMale
L_{ALE} = 1.25 * 0.4 * LMc
e


II
e
d
$DM_S = V_i * X_S$
$C = V_C * X_C$
$AP = V_{AP} * X_{AP}$
$ALE = LM_{ALE} * X_{ALE}$
f
$DM_S <= D_S$
$C <= D_C$
$AP <= D_{AP}$
N
g


g
$L_S = DM_S - D_S$
$L_C = C - D_C$
$L_{AP} = AP - D_{AP}$
$L_{ALE} = ALE$
$L_S > LM_S$
$L_C > LM_C$
$L_{AP} > LM_{AP}$
Y
$L_S = LM_S$
$L_C = LM_C$
$L_{AP} = LM_{AP}$
IV
N
N
$L_S > 0.5*LM_S$
N
Mobile home
Y
$L_S > 0.5*LM_S$
N
V
Y
$L_S = LM_S$
Y
$SumL_S = SumL_S + L_S * P_{D_S}$
$SumL_C = SumL_C + L_C * P_{D_C}$
$SumL_{APP} = SumL_{APP} + L_{AP} * P_{D_{AP}}$
$SumL_{ALE} = SumL_{ALE} + L_{ALE} * P_{D_{ALE}}$
Output $SumL_S$, $SumL_C$, $SumL_{APP}$, $SumL_{ALE}$
VI
INPUT
Probability of wind speed P_{W_i} for given wind speed in the given zip code
$SumEL = SumEL + (SumLs + SumL_C + SumL_{APP} + SumL_{ALE}) * P_{W_i}$
VII
Finish wind speed?
N
Loop 1
Y
Output SumEL
SumAEL = SumAEL + SumEL
VIII
Finish portfolio?
N
Loop 2
Y
Output SumAEL
§
Finish company?
N
Loop 3
Y
Stop

REMARKS:

I

Map the portfolio construction type to the ISO classification and select the corresponding vulnerability matrix

II

The structure limit LM_S is applied based on the portfolio type (replacement cost or actual cash value) to obtain V_i, V_C, and V_{AP}

III

The deductibles applied to structure, content and appurtenant (D_S, D_C, and D_{AP}) are calculated based on the mean damages obtained from the vulnerability matrices

IV

The damages of structure, content, appurtenant and ALE (DM_S, C, AP and ALE) are calculated at different damage ratio intervals. Here, DM_S, C and AP are calculated based on V_i, V_C, and V_{AP}, respectively. Then the losses of structure, content and appurtenant (L_S, L_C and L_{AP}) are computed by applying the deductibles. Two exceptions are handled as follows:

1. If L_S (L_C or L_{AP}) < 0, set it to 0
2. If L_S (L_C or L_{AP}) $>$ LM_S (LM_C or LM_{AP}), set it to LM_S (LM_C or LM_{AP})

The loss of ALE (L_{ALE}) is set to ALE, which is calculated based on the ALE limit (LM_{ALE})

V

The structure loss L_S is compared to the structure limit LM_S. Two situations are handled:

1. For mobile home, if $L_S > 0.5*LM_S$, L_S is set to LM_S
2. For other construction type, if $L_S > 0.5*LM_S$, L_S is set to LM_S

NOTE: It will be implemented. However, at this stage, it will be commented out for later use.

VI

SumL ($SumL_S$, $SumL_C$, $SumL_{APP}$ or $SumL_{ALE}$) is expected loss of the property for a given wind speed, which is calculated by aggregating the losses at different damage intervals with respect to the corresponding damage probabilities.

VII

SumEL is across all wind speeds, which is obtained by aggregating all the expected losses at different wind speed with respect to the corresponding probability for the wind speed.

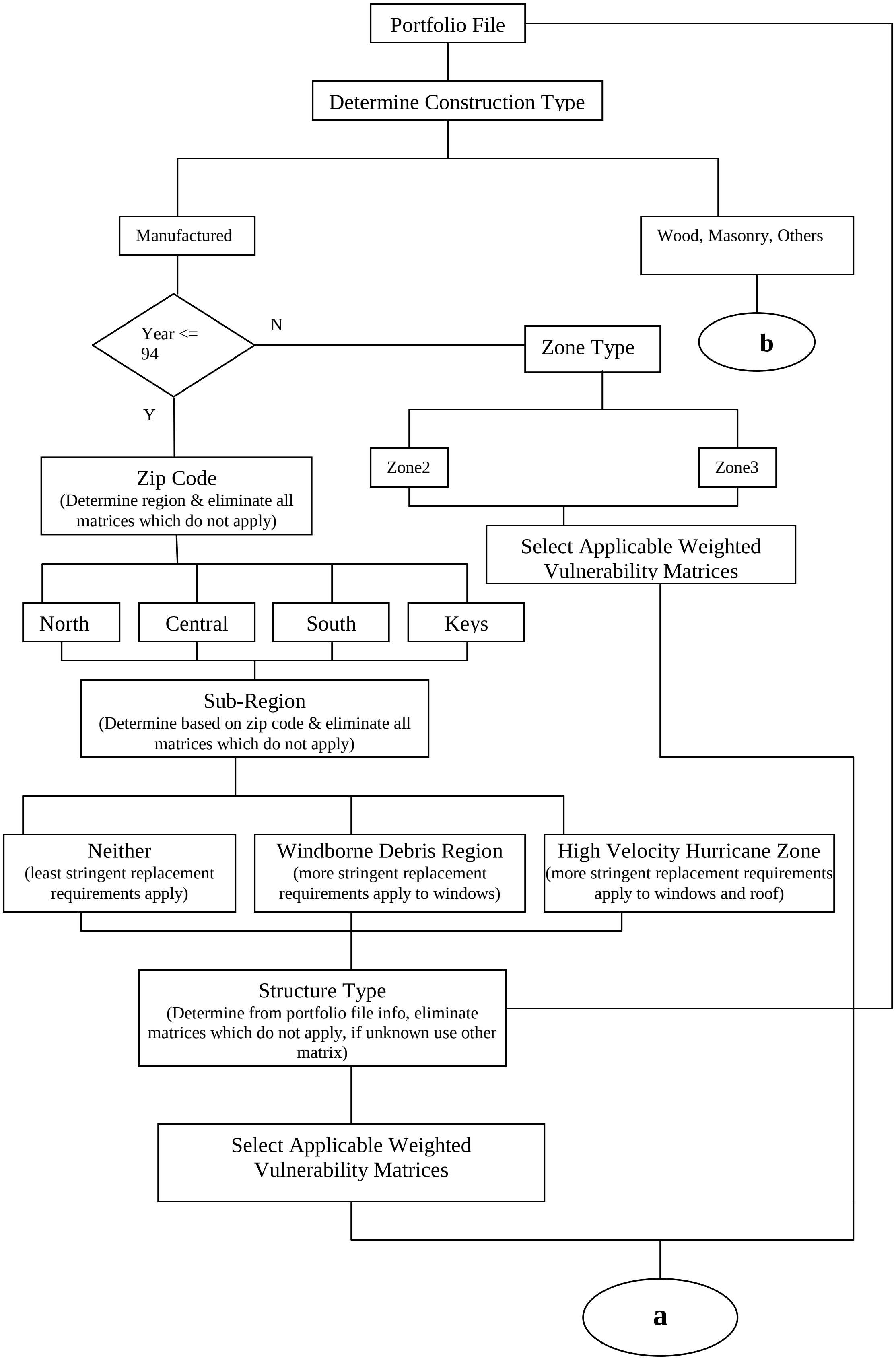
VIII

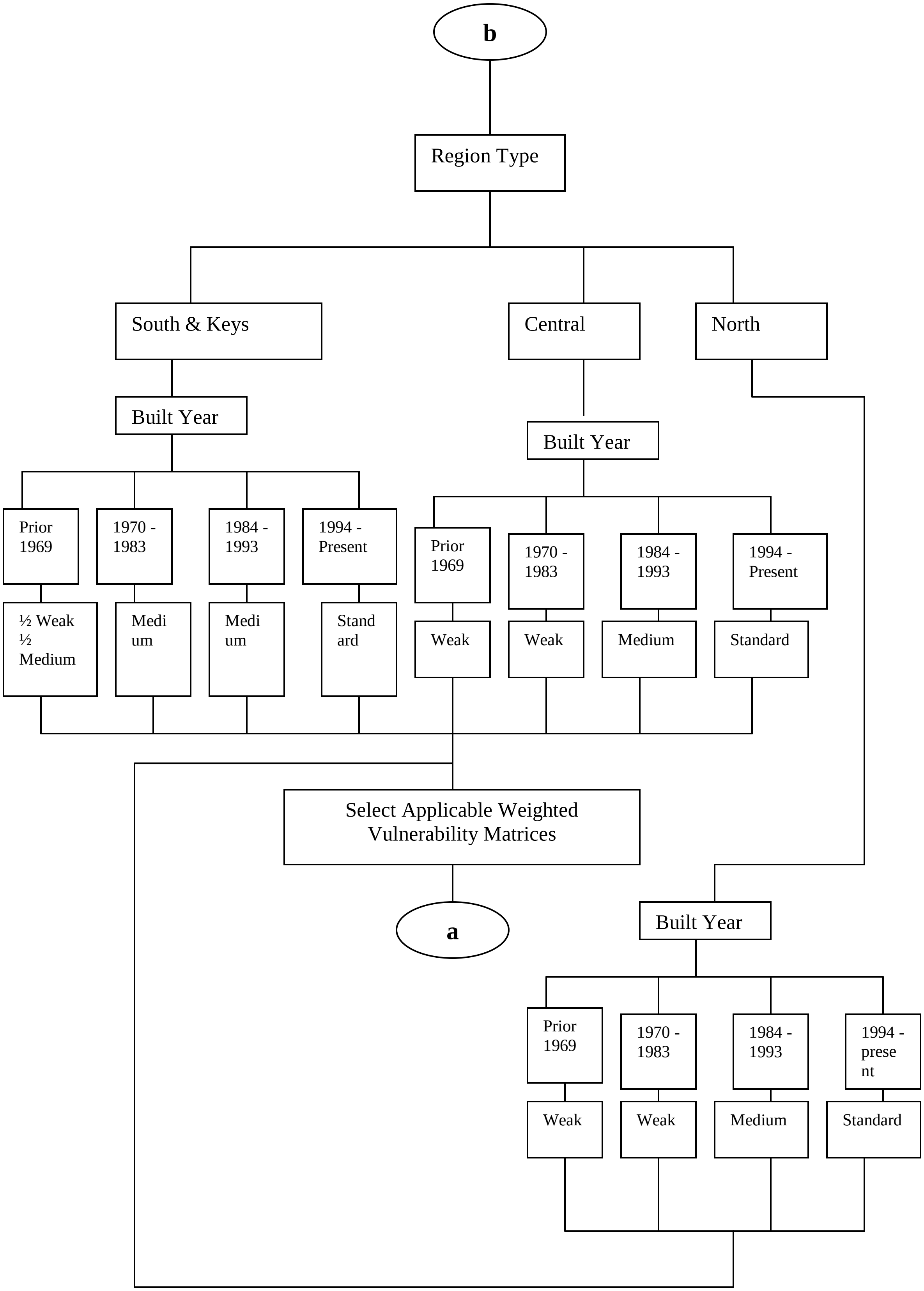
SumAEL aggregates all expected losses for one company.

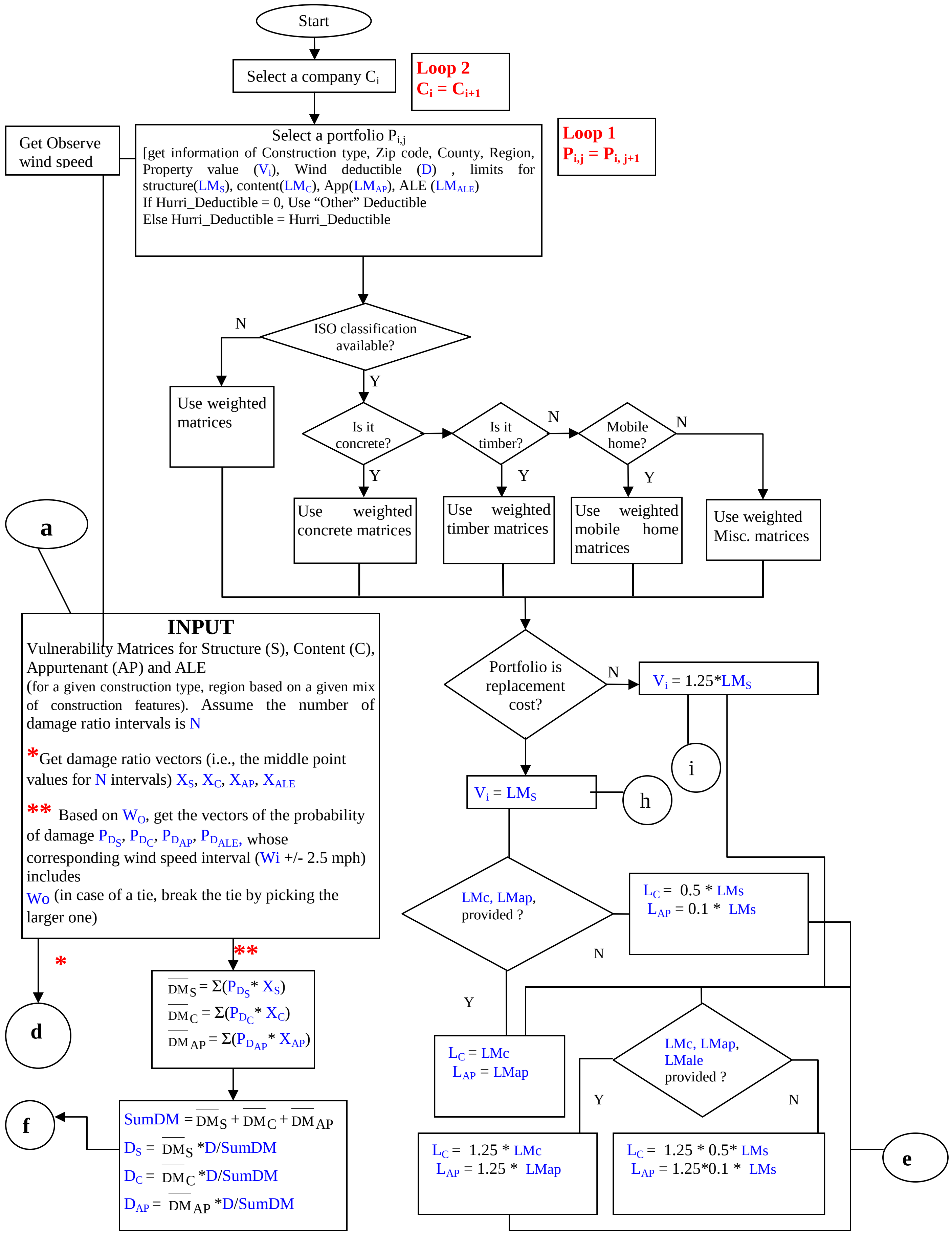
NOTE: Save information (zip code, county, region, construction type, 4 types of coverages, property value, company) for **SumL$_S$, SumL$_C$, SumL$_{APP}$, SumL$_{ALE}$** and **SumEL**. For **SumL$_S$, SumL$_C$, SumL$_{APP}$** and **SumL$_{ALE}$**, save wind speed too and for **SumEL**, save (V_i/sum of V_i) where sum of V_i is for each construction type (Masonry, Timber, Mobile home) and is calculated offline.

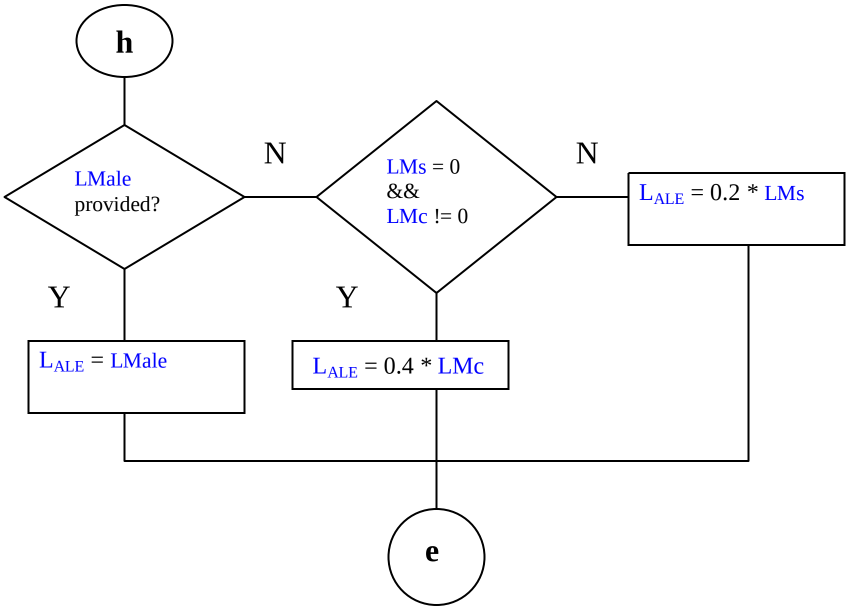
§ Variance of **SumAEL** can be calculated for a company, for a Zipcode or for a construction type.

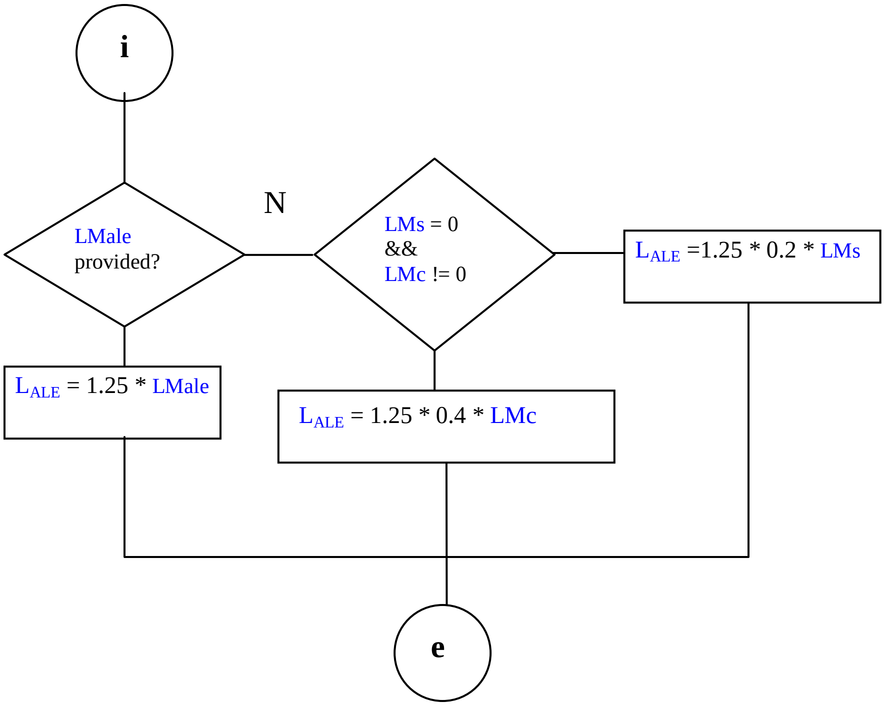
➢ The non-parametric approach for generating Scenario Based Expected Loss Costs

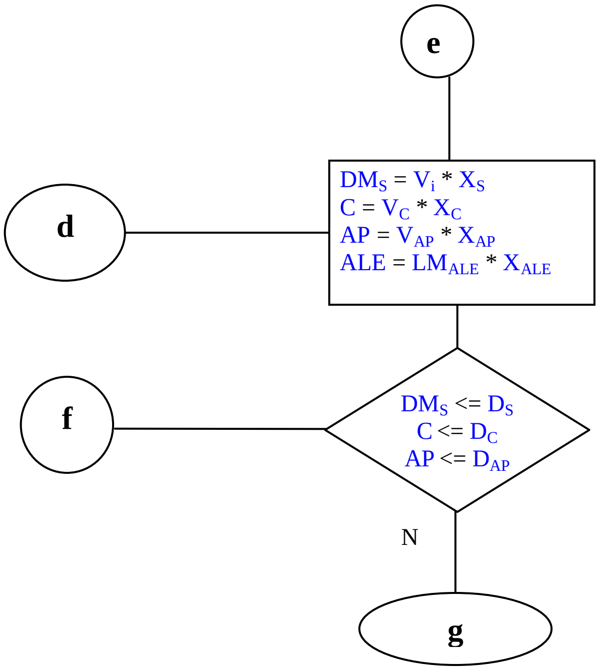

Portfolio File
Determine Construction Type
Manufactured
Wood, Masonry, Others
b
Year <= 94
N
Y
Zone Type
Zone2
Zone3
Select Applicable Weighted Vulnerability Matrices
Zip Code
(Determine region & eliminate all matrices which do not apply)
North
Central
South
Keys
Sub-Region
(Determine based on zip code & eliminate all matrices which do not apply)
Neither
(least stringent replacement requirements apply)
Windborne Debris Region
(more stringent replacement requirements apply to windows)
High Velocity Hurricane Zone
(more stringent replacement requirements apply to windows and roof)
Structure Type
(Determine from portfolio file info, eliminate matrices which do not apply, if unknown use other matrix)
Select Applicable Weighted Vulnerability Matrices
a

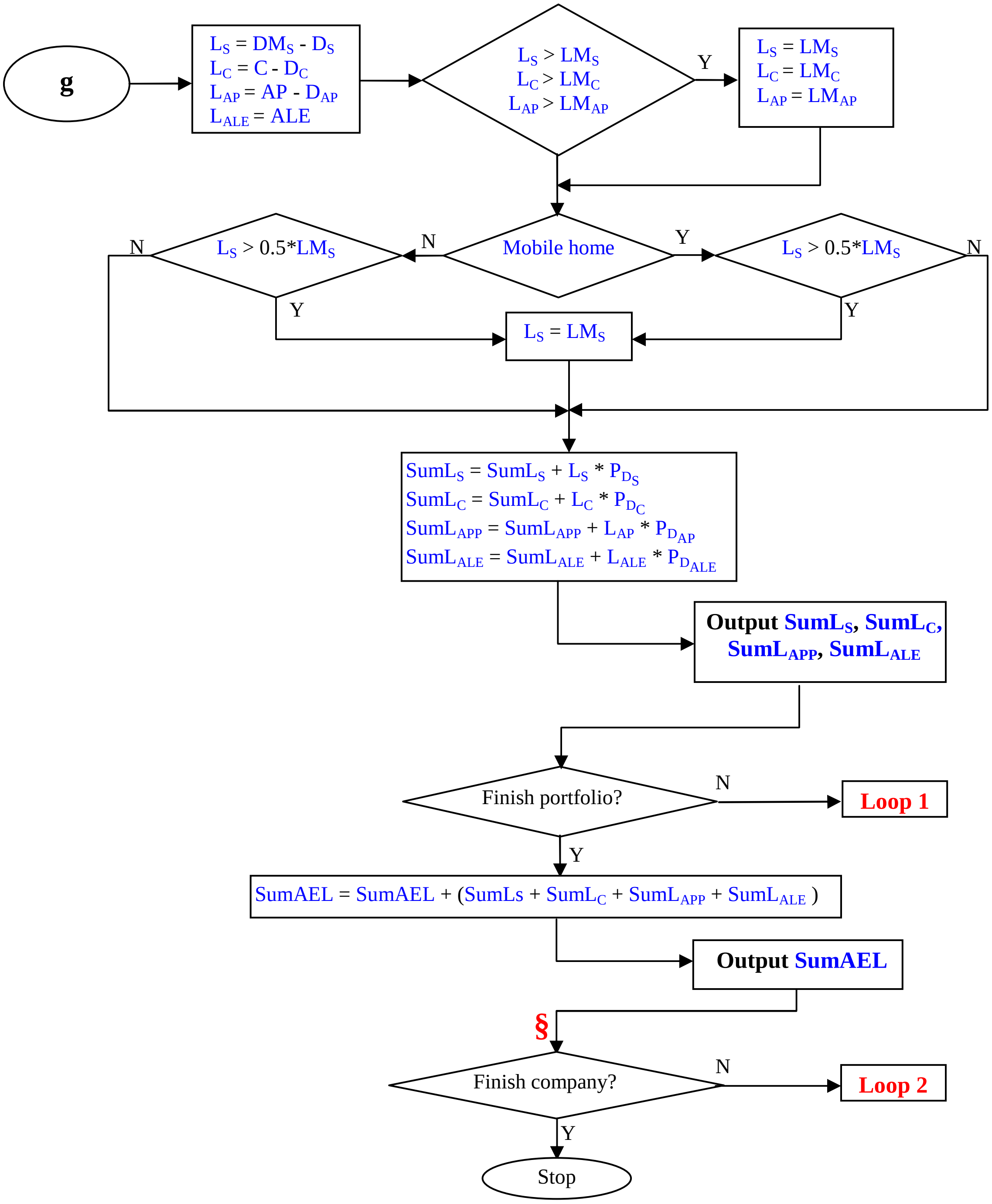

b
Region Type
South & Keys
Central
North
Built Year
Prior 1969
1970 - 1983
1984 - 1993
1994 - Present
½ Weak ½ Medium
Medi um
Medi um
Stand ard
Built Year
Prior 1969
1970 - 1983
1984 - 1993
1994 - Present
Weak
Weak
Medium
Standard
Select Applicable Weighted Vulnerability Matrices
a
Built Year
Prior 1969
1970 - 1983
1984 - 1993
1994 - prese nt
Weak
Weak
Medium
Standard


Start
Select a company C_i
Loop 2
$C_i = C_{i+1}$
Loop 1
$P_{i,j} = P_{i,\ j+1}$
Get Observe wind speed
Select a portfolio $P_{i,j}$
[get information of Construction type, Zip code, County, Region, Property value (V_i), Wind deductible (D) , limits for structure(LM_S), content(LM_C), App(LM_{AP}), ALE (LM_{ALE})
If Hurri_Deductible = 0, Use "Other" Deductible
Else Hurri_Deductible = Hurri_Deductible
ISO classification available?
N
Y
Use weighted matrices
Is it concrete?
Y
Use weighted concrete matrices
Is it timber?
N
Y
Use weighted timber matrices
Mobile home?
N
Y
Use weighted mobile home matrices
Use weighted Misc. matrices
a
INPUT
Vulnerability Matrices for Structure (S), Content (C), Appurtenant (AP) and ALE
(for a given construction type, region based on a given mix of construction features). Assume the number of damage ratio intervals is N
*Get damage ratio vectors (i.e., the middle point values for N intervals) X_S, X_C, X_{AP}, X_{ALE}
** Based on W_O, get the vectors of the probability of damage P_{D_S}, P_{D_C}, $P_{D_{AP}}$, $P_{D_{ALE}}$, whose corresponding wind speed interval (Wi +/- 2.5 mph) includes
Wo (in case of a tie, break the tie by picking the larger one)
*
d
**
$\overline{DM}_S = \Sigma(P_{D_S} * X_S)$
$\overline{DM}_C = \Sigma(P_{D_C} * X_C)$
$\overline{DM}_{AP} = \Sigma(P_{D_{AP}} * X_{AP})$
f
$SumDM = \overline{DM}_S + \overline{DM}_C + \overline{DM}_{AP}$
$D_S = \overline{DM}_S$ *D/SumDM
$D_C = \overline{DM}_C$ *D/SumDM
$D_{AP} = \overline{DM}_{AP}$ *D/SumDM
Portfolio is replacement cost?
N
$V_i = 1.25*LM_S$
i
$V_i = LM_S$
h
LMc, LMap, provided ?
N
Y
$L_C = 0.5 * LMs$
$L_{AP} = 0.1 * LMs$
$L_C = LMc$
$L_{AP} = LMap$
LMc, LMap, LMale provided ?
Y
N
$L_C = 1.25 * LMc$
$L_{AP} = 1.25 * LMap$
$L_C = 1.25 * 0.5 * LMs$
$L_{AP} = 1.25*0.1 * LMs$
e


h
LMale provided?
N
LMs = 0 && LMc != 0
N
L_{ALE} = 0.2 * LMs
Y
Y
L_{ALE} = LMale
L_{ALE} = 0.4 * LMc
e


i
LMale provided?
N
LMs = 0 && LMc != 0
L_{ALE} =1.25 * 0.2 * LMs
L_{ALE} = 1.25 * LMale
L_{ALE} = 1.25 * 0.4 * LMc
e


e
d
$DM_S = V_i * X_S$
$C = V_C * X_C$
$AP = V_{AP} * X_{AP}$
$ALE = LM_{ALE} * X_{ALE}$
f
$DM_S <= D_S$
$C <= D_C$
$AP <= D_{AP}$
N
g



REMARKS:

The steps in scenario-based ILM are similar to the general ILM except that the wind speed for a certain portfolio is given. **SumL** is expected loss of the property for a given wind speed, **SumAEL** aggregates all expected losses for one company. Save information (zip code, county, region, construction type, 4 types of coverages, property value, company) for **SumL**$_S$, **SumL**$_{APP}$, **SumL**$_{ALE}$.

§ Variance of **SumAEL** can be calculated for a company, for a Zipcode or for a construction type.

4.3.3 Class Diagram and Description

A. Class Diagram for ILM

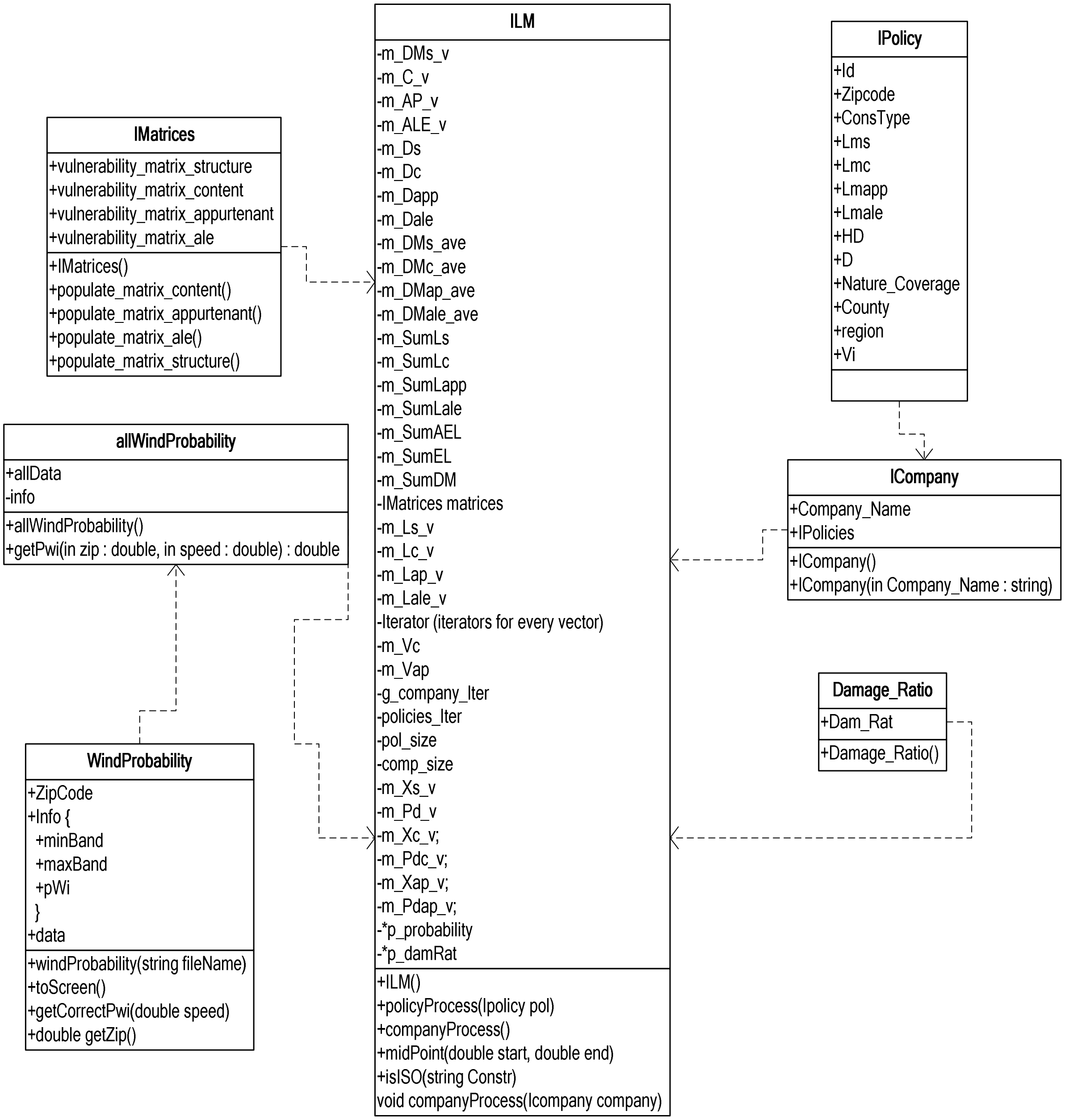


Figure 4.1.2: Class Diagram for ILM-1

♦ Class Description

This section addresses the major classes used and their functionalities.

- **ILM:**
  This is the main class of the application which instantiates all the other classes and performs all the operations specified by the flowchart.

- **Matrices:**
  This class forms the vulnerability matrices for Content, Appurtenant, Ale and Structure.

- **Damage_Ratio:**
  This class reads and stores the Damage Ratio required in calculating the expected losses.

- **Wind_Probability:**
  This class reads and stores the wind probability.

- **Policy:**
  This class reads the input file and categorizes data.

- **Company:**
  This class gets the input data and formulates it for each company in the proper format.

B. Class Diagram for Generating Expected Loss Costs for a Scenario ILM

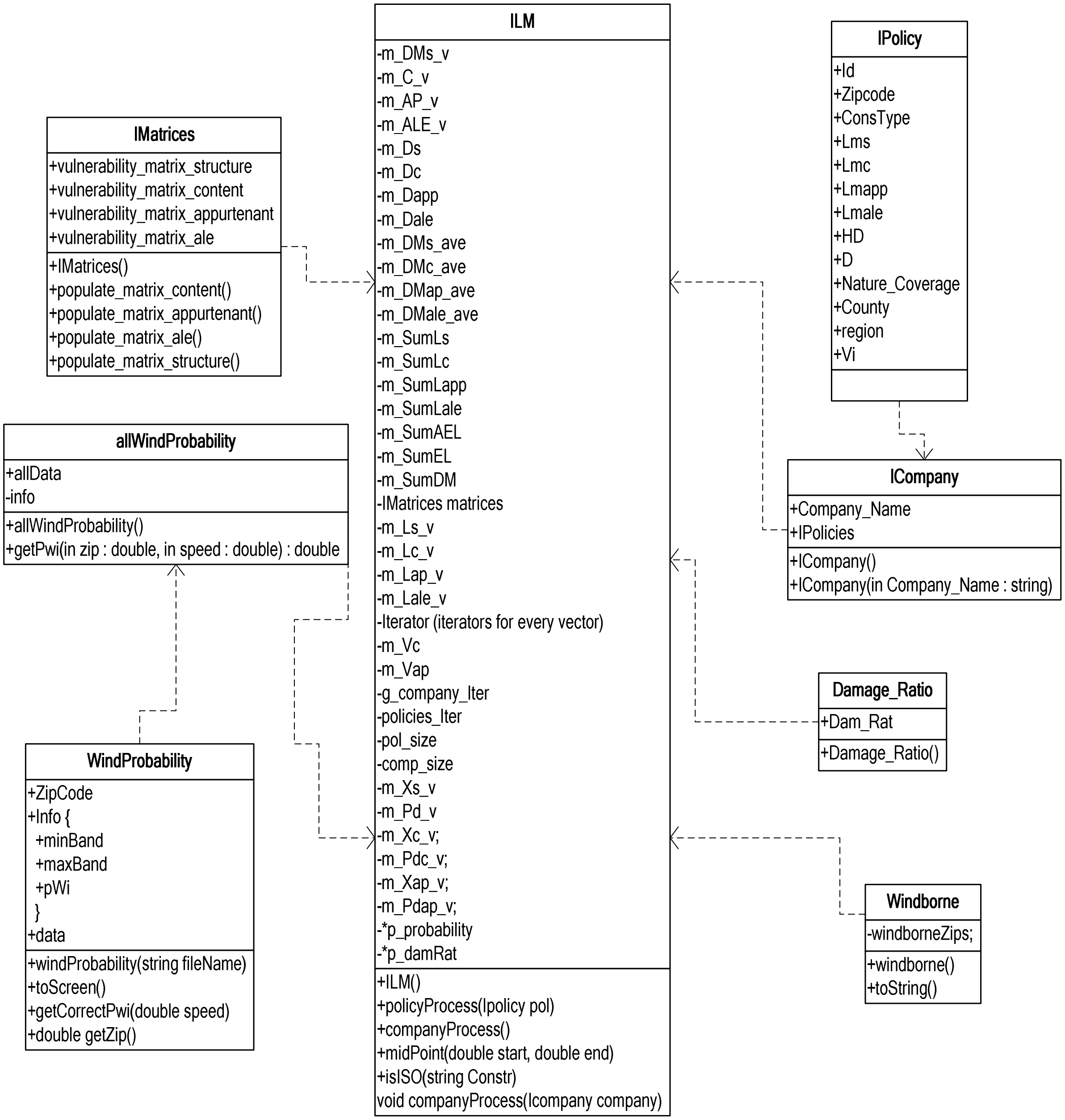


Figure 4.1.3 Class Diagram for ILM-Scenario Based

♦ Class Description

This section addresses the major classes used and their functionalities.

- **ILM:**
  This is the main class of the application which instantiates all of the other classes.

- **Matrices:**
  This class forms the vulnerability matrices for Content, Appurtenant, Ale and Structure in the required format.

- **Damage_Ratio:**
  This class reads and stores the Damage Ratio required in calculating the expected loss of properties.

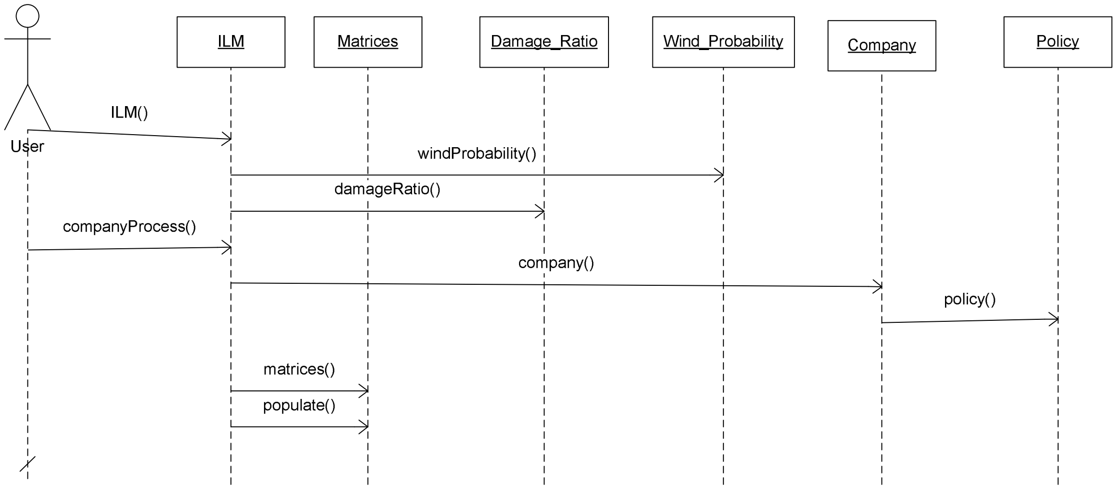
- **Windspeed:**
  This class stores the wind speed corresponding to each zip code.

- **Policy:**
  This class reads the input file and categorizes data.

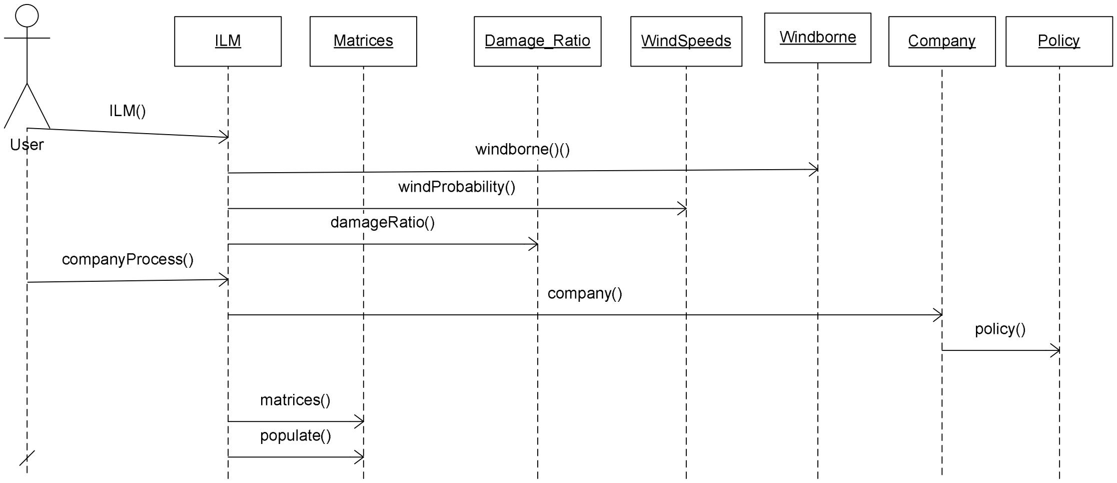
- **Company:**
  This class gets the input data and formulates it for each company in the proper format.

- **Windborne:**
  This class holds the windborne derby region information

4.3.4 Sequence Diagram for ILM



4.3.5 Sequence Diagram for Scenario ILM



References

[1] 2004 National Renovation & Insurance Repair Estimator, J. Russell, Craftsman Book Company, Carlsbad, CA

[2] CEIA Cost 2002, R. Langedyk, V. Ticola, Construction Estimating Institute, Sarasota, FL

Section 5

Database Document

5.1 Specification for the Project

The North Atlantic ‘best track’ is maintained by the forecasters and researchers at the National Hurricane Center in Miami, Florid and the National Climatic Data Center in Asheville, North Carolina. Currently, the Database extends from 1851 to 2001. Based on the provided data, we are going to develop a Database system using Oracle software so that the user of the application can query the Database and get the statistic reports from the Database.

In this project, an interface is provided for the users to select any data series from five data sets. The application then retrieves data from Oracle Database, uses two probability distributions to fit the selected data set, and returns the fit result to users. The simulation results are plotted as graphs.

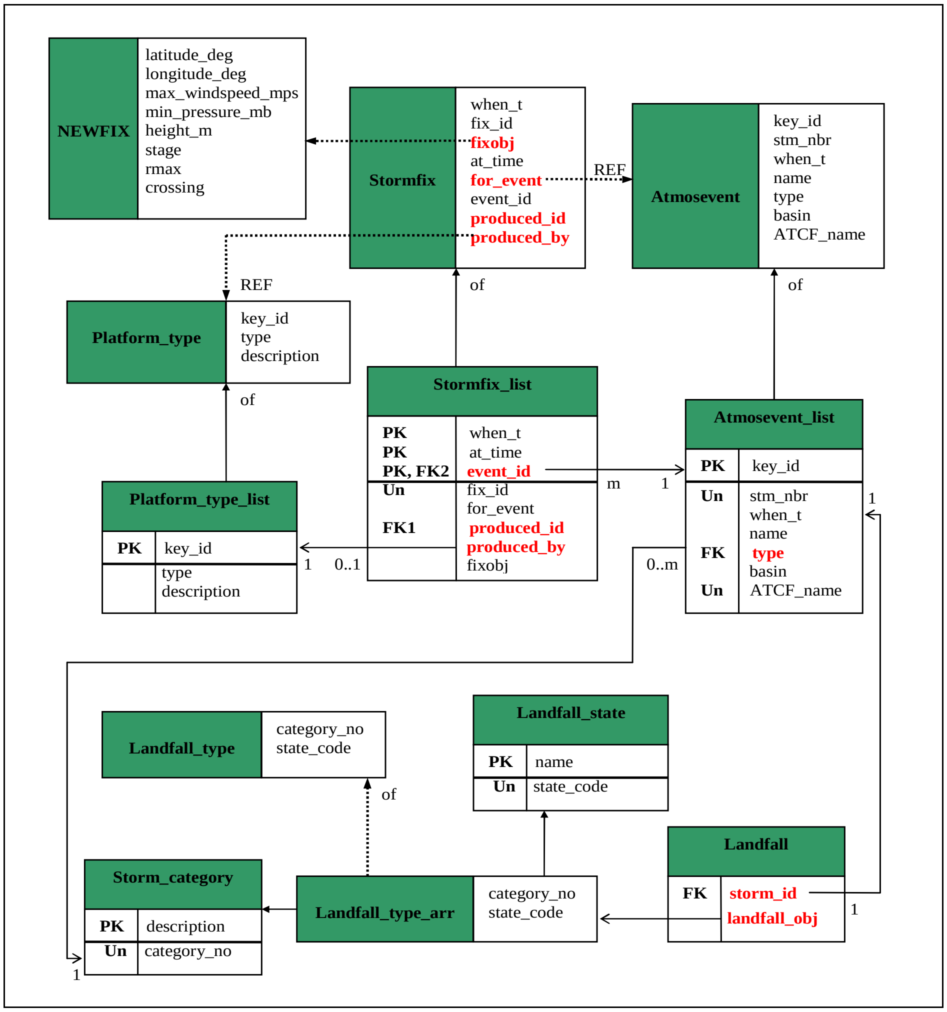
5.2 Data Modeling

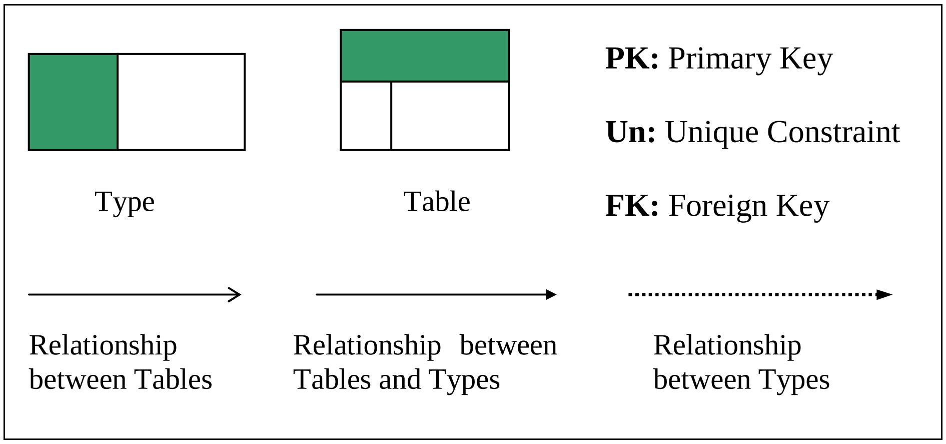
Since the data modeling is the most important part of the system’s development process, the characteristics of data captured during data modeling are crucial in the design of database, programs, and other system components. The facts and rules captured during the process of data modeling are essential to assuring data integrity in an information system.

Data rather than processes are the most complex aspects of many modern information systems, and hence play a central role in structuring system requirements. An Object Relational Model is based on the traditional Oracle Relational Database and is extended to include Object Oriented concepts and structures, such as abstract data types, nested tables and varying arrays.

In this project we use the Object Oriented concept due to the following reasons:

1. Object Reuse: Creating Object Oriented Database objects will facilitate the reuse of the Database objects.
2. Standard Adherence: If multiple applications or tables use the same set of Database objects, a standard must be created for those Database objects. For example, you can create a standard data type used for all address data.
3. Defined Access Paths: For each object you can define the procedures and functions that act upon it, which means you can unite the data object and the methods that access it. Having the access paths defined in this manner allows you to standardize the data access methods and enhances the reusability of the objects.




Type
Table
Relationship between Tables
Relationship between Tables and Types
Relationship between Types

PK: Primary Key

Un: Unique Constraint

FK: Foreign Key

5.3 Description of the Objects and Tables

There are 6 object types in HURDAT Schema:

1.

NEWFIX	**latitude_deg** **longitude_deg** **max_windspeed_mps** **min_pressure_mb** **height_m** **stage** **rmax** **crossing**

TYPE: NEWFIX

latitude_deg	NUMBER(10,4)
longitude_deg	NUMBER(10,4)
max_windspeed_mps	NUMBER(10,4)
min_pressure_mb	NUMBER(6)
height_m	NUMBER(8,3)
stage	NUMBER(2)
rmax	NUMBER(4)
crossing	VARCHAR2(10)

2.

Stormfix	**when_t** **fix_id** **fixobj** **at_time** **for_event** **event_id** **produced_id** **produced_by**

TYPE: STORMFIX

fix_id	NUMBER
when_t	DATE
at_time	CHAR(6)
event_id	NUMBER(6)
for_event	REF ATMOSEVENT
produced_id	NUMBER(4)
produced_by	REF PLATFORM_TYPE
fixobj	NEWFIX

3.

Atmosevent	**key_id** **stm_nbr** **when_t** **name** **type** **basin** **ATCF_name**

TYPE: ATMOSEVENT

key_id	NUMBER(6)
stm_nbr	NUMBER(6)
when_t	DATE
name	VARCHAR2(30)
type	NUMBER(2)
basin	NUMBER(2)
ATCF_name	VARCHAR2(20)

4.

Platform_type	**key_id** **type** **description**

TYPE: PLATFORM_TYPE

key_id	NUMBER(4)
type	VARCHAR2(50)
description	VARCHAR2(50)

5.

Landfall_type	**category_no** **state_code**

TYPE: LANDFALL_TYPE

category_no	NUMBER(2)
state_code	RCHAR2(4)

6.

Landfall_type_arr	**category_no** **state_code**

TYPE: LANDFALL_TYPE_ARR

category_no	NUMBER(2)
state_code	RCHAR2(4)

This object is an array of the LANDFALL_TYPE object

There are 5 tables in HURDAT Schema:

Table 1:

Platform_type_list	
PK	**key_id**
	type **description**

TABLE 1: PLATFORM_TYPE_LIST

Key_id	PRIMARY KEY CONSTRAINT

This table is based on Object Platform_type

Table 2:

Stormfix_list	
PK	**when_t**
PK	**at_time**
PK, FK2	**event_id**
Un	**fix_id**
	for_event
FK1	**produced_id**
	produced_by
	fixobj

TABLE 2: STORMFIX_LIST

when_t	NOT NULL
constraint	FIX_ID_UN UNIQUE (FIX_ID)
constraint	EVENT_ID_FK FOREIGN KEY (EVENT_ID)
constraint	PRODUCED_ID_FK FOREIGN KEY (PRODUCED_ID)
constraint	FIX_ID_PK PRIMARY KEY (EVENT_ID,WHEN_T,AT_TIME)

Table 3:

Atmosevent_list	
PK	key_id
Un	stm_nbr
	when_t
	name
FK	type
	basin
Un	ATCF_name

TABLE 3: **ATMOSEVENT_LIST**

constraint	AL_KEY_ID_PK PRIMARY KEY(KEY_ID)
constraint	AL_stm_nbr_UN UNIQUE (STM_NBR)
constraint	AL_ATCF_name_UN UNIQUE (ATCF_NAME)

Table 4:

Storm_category	
PK	description
Un	category_no

TABLE 4: STORM_CATEGORY

category_no	NUMBER(2)
constraint	S_CATEGORY_NO_UN UNIQUE
description	VARCHAR2(30)
constraint	S_DESCRIPTION_PK PRIMARY KEY)

Table 5:

Landfall	
FK	storm_id
	landfall_obj

TABLE 5: LANDFALL

storm_id	NUMBER(6)
landfall_obj	LANDFALL_TYPE_ARR

constraint	LD_EVENT_ID_FK FOREIGN KEY(STORM_ID) REFERENCES ATMOSEVENT_LIST(KEY_ID) ON DELETE CASCADE

NESTED TABLE landfall_obj STORE AS landfall_obj_list

Table 6:

Landfall_state	
PK	**name**
Un	**state_code**

TABLE 6: LANDFALL_STATE

state_code	VARCHAR2(4)
constraint	STATE_CODE_UN UNIQUE
name	VARCHAR2(30)
constraint	LD_NAME_PK PRIMARY KEY)

5.4 Data Processing

5.4.1 Original Data Processing

Following is the data format of the original text file recording the storm tracks of Atlantic basin. In order to populate the data into the database schema, we have to process the data and convert them into some suitable format according to database schema.

```
00005 06/25/1851 M= 1  1 SNBR=   1 NOT NAMED   XING=1 SSS=1
00010 06/25*  0   0   0    0*  0   0   0    0*285 965  70    0*  0   0
0    0
00015 HRBTX1
00020 07/05/1851 M= 1  2 SNBR=   2 NOT NAMED   XING=0 SSS=0
00025 07/05*  0   0   0    0*  0   0   0    0*222 976  80    0*  0   0
0    0
00030 HR
00035 07/10/1851 M= 1  3 SNBR=   3 NOT NAMED   XING=0 SSS=0
00040 07/10*  0   0   0    0*  0   0   0    0*120 600  50    0*  0   0
0    0
00045 TS
00050 08/16/1851 M=12  4 SNBR=   4 NOT NAMED   XING=1 SSS=3
00055 08/16*134 480  40    0*137 495  40    0*140 510  50    0*144 528
50    0
00060 08/17*149 546  60    0*154 565  60    0*159 585  70    0*161 604
70    0
00065 08/18*166 625  80    0*169 641  80    0*172 660  90    0*176 676
90    0
00070 08/19*180 693  90    0*184 711  70    0*189 726  60    0*194 743
60    0
```

```
00075 08/20*199 759  70    0*205 776  70    0*212 790  70    0*219 804
70    0
00080 08/21*226 814  60    0*232 825  60    0*239 836  70    0*244 843
70    0
00085 08/22*250 849  80    0*256 855  80    0*262 860  90    0*268 863
90    0
00090 08/23*274 865 100    0*280 866 100    0*285 866 100    0*296 861
100    0
00095 08/24*307 851  90    0*316 841  70    0*325 830  60    0*334 814
50    0
00100 08/25*340 800  40    0*348 786  40    0*358 770  40    0*368 751
40    0
00105 08/26*378 736  40    0*389 718  40    0*400 700  40    0*413 668
40    0
00110 08/27*428 633  40    0*445 602  40    0*464 572  40    0*485 542
40    0
00115 HRAFL3 GA1
```

There are three basic types of data lines in the original storm track file.

TYPE A:

92620 08/16/1992 M=13 2 SNBR= 899 ANDREW   XING=1 SSS=4

1.92620	Card#
2.08/16/1992	MM/DD/Year Days
3.M=13	S#
4.2	Total#
5.ANDREW	Name
6.XING=1	US Hit
7.SSS=4	Hi US category

Card#:	Sequential card number starting at 00010 in 1851
MM/DD/Year:	Month, Day, and Year of storm
Days:	Number of days in which positions are available (note that this also means number of lines to follow of type B and then one line of type C)
S#:	Storm number for that particular year (including subtropical storms)
Total#:	Storm number since the beginning of the record (since 1886)
Name:	Storms only given official names since 1950

US Hit:	'1' Made landfall over the United States as tropical storm or hurricane. '0' did not make U.S. landfall
Hi US category:	'9' Used before 1899 to indicate U.S. landfall as a hurricane of unspecified Saffir-Simpson category. '0' Used to indicate U.S. landfall as tropical storm, but this has not been utilized in recent years '1' to '5' = Highest category on the Saffir-Simpson scale that the storm made landfall along the U.S. '1' is a minimal hurricane, '5' is a catastrophic hurricane

TYPE B:

92580 04/22S2450610 30 1003S2490615 45 1002S2520620 45 1002S2550624 45 1003*

1. 92580	Card#
2. 04/22	MM/DD
3. S	Storm category
4. 2450610 30 1003	LatLongWindPress
5. 2490615 45 1002	LatLongWindPress
6. 2520620 45 1002	LatLongWindPress
7. 2550624 45 1003	LatLongWindPress

Card#:	Sequential card number starting at 00010 in 1851
MM/DD	Month, Day, and Year of storm
Storm category	'S' (Subtropical stage), '*' (tropical cyclone stage), 'E' (extra tropical stage), 'W' (wave stage - rarely used)
LatLong	Position of storm: 24.5N, 61.0W
Wind	Maximum sustained (1 minute) surface (10m) windspeed in knots (in general, these are to the nearest 5 knots).
Press	Central surface pressure of storm in mb (if available). Since 1979, central pressures are given every time even if a satellite estimation is

	needed.
Position and intensity	Positions and intensities are at 00Z, 06Z, 12Z, 18Z

TYPE C:

92760 HRCFL4BFL3 LA3

1.	92760	Card#
2.	HR	Tp
3.	CFL, BFL, LA	Hit
4.	4, 3	Storm Category

Card#:	Sequential card number starting at 00010 in 1851
Tp	Maximum intensity of storm ('HR' = hurricane, 'TS' = tropical storm, 'SS' = subtropical storm)
Hit	U.S. landfallings as hurricane ('LA' = Louisiana, etc.) and Saffir-Simpson category at landfall ('1' = minimal hurricane '5' = super hurricane). (Note that Florida and Texas are split into smaller regions: 'AFL' = Northwest Florida, 'BFL' = Southwest Florida, 'CFL' = Southeast Florida, 'DFL' = Northeast Florida, 'ATX' = South Texas, 'BTX' = Central Texas, 'CTX' = North Texas.)

The first step is to extract the useful data and to remove the unwanted data or format symbols. For Table 'atmosevent_list' which records the high-level information for all storms, we need to extract the following corresponding data fields from the original data file:

1. Storm number
2. Begin date of that storm or hurricane
3. Type of the storm or hurricane (The type of the hurricane or storm is based on a category criterion.), which is calculated by converting the maximum wind speed of each storm to its corresponding storm category according to some criteria.

We use a C++ program to retrieve the data and then categorize the storm type based on its maximum wind speed.

Table ‘stormfix_list’ stores the detailed information about each storm or hurricane. For example, it records how many days a storm lasts, the exact latitude and longitude, the wind speed and the central pressure at different fix point of each day. We therefore need to obtain this information from the original data file. A java program is developed to achieve this goal.

In order to make sure the extracted data consistent with the original data file, we have done a lot of checking, either manually or by programs.

5.4.2 New Data Processing

On 04/24/03, we received a new data file “rmax.dat” which contains Rmax value for each fix and the crossing point for specific points. In addition, some intermediate fixes which are not in the HURDAT database have been included. Therefore, the data file needs to be processed and two attributes have to be added into the FIX object in the database as follows:

Name	Length
rmax	NUMBER(4)
Crossing	VARCHAR2(10)

Following is the data format of the new text file:

```
Storm Name  # Year Mo Dy Time    Lat    Lon  Wsp  Pmn RMW Cat Crossg
NOT NAMED   3 1903/ 9/ 9 0600   21.4   72.4  50    0   0 TSt
NOT NAMED   3 1903/ 9/ 9 1200   21.8   73.4  50    0   0 TSt
NOT NAMED   3 1903/ 9/ 9 1800   22.2   74.0  50    0   0 TSt
NOT NAMED   3 1903/ 9/10 0000   22.6   74.7  55    0   0 TSt
NOT NAMED   3 1903/ 9/10 0300   22.9   75.0  60    0   0 TSt ISLAND new
NOT NAMED   3 1903/ 9/10 0600   23.2   75.3  60    0   0 TSt
NOT NAMED   3 1903/ 9/10 1200   23.8   76.0  65    0   0 Hu1
NOT NAMED   3 1903/ 9/10 1800   24.0   76.5  70    0   0 Hu1
NOT NAMED   3 1903/ 9/11 0000   24.4   76.9  80    0   0 Hu1
NOT NAMED   3 1903/ 9/11 0600   24.9   77.5  85    0   0 Hu2
NOT NAMED   3 1903/ 9/11 1000   25.3   78.1  85    0   0 Hu2 ISLAND new
NOT NAMED   3 1903/ 9/11 1200   25.4   78.4  85    0   0 Hu2
NOT NAMED   3 1903/ 9/11 1800   25.8   79.1  85    0   0 Hu2
NOT NAMED   3 1903/ 9/11 2200   26.1   80.0  85    0   0 Hu2 LAND   new
NOT NAMED   3 1903/ 9/12 0000   26.4   80.3  75    0   0 Hu1
NOT NAMED   3 1903/ 9/12 0600   26.9   81.2  65    0   0 Hu1
```

Where the “RMW” column represents the rmax value and the “Crossg” column represents the crossing value. And the word “new” next to the “Crossg” column indicates the new fixes.

The data processing steps are similar to the ones mentioned in the previous subsection.

5.5 Data Loading

5.5.1 Original Data Loading

The output of data pre-processing is the desired data format we need for populating the data into the new schema. For testing purpose, we first loaded the data into the FDOI at georges.cs.fiu.edu using SQL Loader. All the constraints in the database schema have been disabled in order to facilitate the loading process. The loading codes for the three major database tables (atmosevent_list, stormfix_list, landfall) are listed as follows:

1. Loading data into Table ‘atmosevent_list’:

```
load data
infile 'atmosevent.dat'
append
into table atmosevent_list
fields terminated by ","
trailing nullcols
(stm_nbr,when_t date "mm/dd/yyyy",name,type,basin, key_id
"atm_key_seq.nextval")
```

2. Loading data into Table ‘landfall’:

```
LOAD DATA
INFILE 'landfall.dat'
TRUNCATE INTO TABLE landfall
trailing nullcols
   (
   storm_id TERMINATED BY ',',
   landfall_obj nested table TERMINATED BY ','
      (
      dummy_name COLUMN OBJECT
         (
         state_code   TERMINATED BY ':',
         category_no TERMINATED BY ':'
         )
      )
   )
```

3. Loading data into Table ‘stormfix_list’:

```
load data
infile 'stormfix_list_test.dat'
append
into table STORMFIX_LIST_TEST
fields terminated by ","
trailing nullcols
(event_id, when_t date "mm/dd/yyyy",at_time,
fixobj column object
(
LATITUDE_DEG,
LONGITUDE_DEG,
MAX_WINDSPEED_MPS,
MIN_PRESSURE_MB,
stage),
fix_id "obsid_seq.nextval")
```

After finishing the data loading, all the constraints and data references will be enabled.

5.5.2 New Data Loading

Since “rmax.dat” contains only the updated or supplemented information for the hurricanes stored in the database, the new data loading process is different from the original data loading process. Basically, two tables in database need to be altered: atmosevent_list and stormfix_list. The updating steps are discussed as follows:

II. Create a temporary table oldstormfix_list by copying all the data from table stormfix_list:

```
Create table oldstormfix_list as select * from stormfix_list
```

III. Create a new data type NEWFIX, which has two more attributes (rmax, crossing) than FIX.

IV. Replace table stormfix_list by using new data type NEWFIX instead of the original data type FIX.

V. Copy all the data in table oldstormfix_list to table stormfix_list. The values are set as NULL for rmax and crossing:

```
insert into stormfix_list
    (fix_id,when_t,at_time,event_id,fixobj)
 select fix_id,when_t,at_time,event_id,
newfix(c.fixobj.latitude_deg,c.fixobj.longitude_deg,c.fixobj.max
    _windspeed_mps,
    c.fixobj.min_pressure_mb,null,c.fixobj.stage,null,null)
from oldstormfix_list c
```

VI. Get the according fix_id in table stormfix_list for each record in “rmax.dat” except the records marked as new. Update table stormfix_list.

VII. Append the records marked as new into table stormfix_list.

VIII. Update table atmosevent_list based on the updated table stormfix_list.

5.6 Export and Import the Data

The next step is to migrate the whole database from fdoi.georges.cs.fiu.edu to hldp.andrew.cs.fiu.edu. We make use of the Oracle export and import utility to complete the task.

Before we begin using the Export utility, the following steps are necessary:

Export the Schema:

Step 1: Run catexp.sql

This job is done by Lin Luo, the DBA of HLDP database. The script performs the following tasks to prepare the database for export:

- Creates the necessary export views in the data dictionary
- Creates the EXP_FULL_DATEBASE role
- Assigns all necessary privileges to the EXP_FULL_DATEBASE
- Assigns EXP_FULL_DATEBASE to the DBA roll
- Records the version of catexp.sql that has been installed

Step 2: Ensure that there is enough disk space to write the export file

Since our database is not very big in size, there is no problem about the storage.

Step 3: Verify that we have the required access privileges

To use Export, you must have the CREATE SESSION privilege on an Oracle database. To export tables owned by another user, the EXP_FULL_DATEBASE role has to be granted to the user who will perform the export.

Step 4: Prepare the parameter file

We specify all needed parameters and their values in a parameter file. Storing the parameters in a file allows the parameters to be easily modified or reused, which is the recommend method for invoking Export.

We create the parameter file using the DOS text editor as follows:

```
FILE=dba.dmp                 // the name of the Exported dump file
OWNER=czhang02    // we export the schema from czhang02's account
GRANTS=y             // exports objects grants
ROWS=y            // rows of table data are exported
COMPRESS=y          // compress the exported file
log=dbaemp              // save export reports and error information to file dbaemp
```

Step 5: Invoking the Export Utility

In our case, we use the User mode to export the entire schema from Chengcui's account on the FDOI to HLDP. As described above, the parameter file method was used to invoke the export utility.

Execute the following command in DOS:

```
> exp username/password    PARFILE = params.dat
```

Import the Schema:

Through the above 5 steps, we successfully export the entire schema from Chengcui's account. The next step is to use the Import utility to read dba.dmp file into the HLDP account.

Step 1: Verify that we have the required access privileges

To use Import, you must have the CREATE SESSION privilege on an Oracle database. To Import tables owned by another user, the IMP_FULL_DATEBASE role has to be granted to the user who will perform the export.

Step 2: Prepare the parameter file

We specify all needed parameters and their values in a parameter file. Storing the parameters in a file allows us to be easily modified or reused, and is the recommend method for invoking Import.

We create the parameter file using the DOS text editor as follow:

```
FILE=dba.dmp          // the name of the export dump file
OWNER=czhang02    // we import the schema from czhang02's account
IGNORE=n               // display object creation errors
SHOW=y                   // list the contents of the export file which are not imported
GRANTS=y             // imports objects grants
ROWS=y             // rows of table data are imported
LOG=dbaemp         // save the import report and error information to file dbaemp
```

Step 3: Invoking the Import Utility

In our case, we use the User mode to import the exported dump file dba.dmp to HLDP.

Execute the following command in DOS:

```
> imp username/password    PARFILE = paramsi.dat
```

5.7 Data Checking

Since the import was terminated with warnings, we have to check that the entire schema in the old account is moved to the new account. After importing, Chengcui (a team member) made a first pass check to make sure that the schema in the new account is the same as the one in the old account. Although there are warnings with the import, but actually all the data and tables as well as database objects are all successfully imported.

It is very important to ensure that the imported data is consistent with the original data file. We randomly retrieved some records from the table in the imported schema, and compared then with the original data file. Three main tables have been checked by this way, and two of them were found correctly imported. But for the third one (‘stormfix_list’), there is a problem with one of the attributes. Some values of that attribute are not consistent with the original data. So we double checked the database and realized that the problem is due to the format of the original file. After changing the program, the needed data can be extracted correctly.

5.8 Queries

5.8.1 Change the Query Based on the New Schema

Once the new schema has been successfully migrated onto the new database server, the next step is to provide the database queries based on new schema. Since the original queries are based on the old schema, we need to revise the original queries according to the new schema.

The following is the query for the old schema.

```
Select     Year,  count(1)
From
    select    to_char(s.when_t, 'yyyy') Year
    from    fdoifiu.stormfix_list s
              where  s.for_event.basin=1 and when_t between '01-JAN-1851'
               and '31-DEC-2000' and
              (s.fixobj.stage like 'H%' or s.fixobj.stage='Tropical Storm')
                        group by to_char(s.when_t, 'yyyy'), event_id  )
         group by Year
         order by Year
```

In the old schema, the storm category is represented by string instead of category id. For example, the string “Hurricane” or “Tropical Storm” was used to record the type of tropical cyclones. But in the new schema, the numbered id is used to categorize the type of the tropical cyclones. Instead of using string, we can use number ‘4’ to represent a “tropical storm”, and number 5~11 to represent hurricane level 1~5.

According to the new schema, we change all **s.fixobj.stage like 'H%' or s.fixobj.stage='Tropical Storm'** statements in the old schema to **s.fixobj.stage >=4.** In the new schema, **s.fixobj.stage >=4** functions the same way as the old one using string matching. Shown below is the revised query, which works correctly in the new schema and is more effective compared with the original queries.

```
select   Year, count(*)"Cyclones"
from    (select   to_char(when_t,'yyyy') Year
         from     atmosevent_list s
         where    s.basin=1 and s.when_t between '01-JAN-1851'
                  and '31-DEC-2001' and  s.type >=4  )
group    by Year
```

We made the same changes for all the queries of Use Case One and Use Case Two. The execution speed of queries is nearly three times faster than before.

5.9 Database Tuning

5.9.1 Tuning SQL Statements

Although the execution speeds of the SQL statements have been greatly improved by revising the schema, additional SQL tuning efforts are necessary to improve the performance of the statements.

The Goals of SQL Tuning

Oracle SQL tuning is a phenomenally complex subject, and we will begin with a high-level description of the goals of SQL tuning and get into details later on. There are some general guidelines that all Oracle SQL developers must follow in order to improve the performance of their systems. The goals of SQL tuning are as follows:

- **Remove Unnecessary Large-table Full-table Scans**

Unnecessary full-table scans cause a huge amount of I/O and can drag down an entire database. We first evaluate the SQL query statements in terms of the number of rows returned by the query. If the query returns less that 40 percent of the table rows on an ordered table, or 7 percent of the rows in an un-ordered table, the query can be tuned to use an index in lieu of the full-table scan. The most common tuning remedy for unnecessary full-table scan is adding indexes. Standard B-tree indexes, bitmapped indexes and function-based indexes can all be added into the tables in order to eliminate full-table scans. In some cases, an unnecessary full-table scan can be converted to an index scan by adding an indexes hint to the SQL statement.

- **Share SQL Statements**

ORACLE holds SQL statements in memory after it has parsed them, so the parsing and analysis do not have to be repeated if the same statement is issued again. The single shared context area in the shared buffer pool of the System Global Area (SGA) is shared by all the users. We have to set the appropriate INIT.ORA parameters for the context areas. The larger the area, the more statements can be retained there and the more likely statements are to be shared.

- **Use Hints**

In general, hints serve a dual purpose. They can be used to alter the execution plan for a SQL statement. They can be used as an alternative to stored outlines to permanently change the execution plan for a SQL statement. When a hint is added to a SQL statement during tuning, the tuning changes will take effect.

- **Verify Optimal Join Techniques**

Some queries will perform faster with nested loop joins, while others may work better with Hash joins or merge/star joins. In general, it is better to use simple join whenever it is possible.

- **Review Sub queries**

Every correlated and non-correlated sub query should be examined to determine if the SQL query could be rewritten as a simple table joins.

Having shown the goals of SQL tuning, the followed section is to tune the SQL statements for the database queries. Oracle Corporation has developed a lot of utilities to facilitate the SQL tuning process. In this project, we mainly use the SQL Trace, TKPROF, and the Timing Environments Parameter for SQL tuning.

Using the Timing Environments Parameter

SQL timing environments parameter is used to record total time elapsed for a SQL statement. For the purpose of testing, we turn on this timing parameter, and run the desired SQL statement. Based on the total time used, we change the structure of the SQL statement, and run it again. Then the two results are compared to decide which statement has a better performance.

Example:
Structure 1:

```
SQL> select /*+ first_rows */ Year, count(*)"Cyclones" from --to
response with the first row quickly
  2    (select  to_char(s.when_t, 'yyyy') Year
  3     from  oscillation_constant_list o, atmosevent_list s
  4       where    to_number(to_char(when_t,'yyyy'))=os_year and
s.basin=1 and  s.type >=4
  5       )
  6    group by Year
  7  /

Elapsed: 00:00:00.02
```

Structure 2:

```
SQL> select /*+ first_rows */ Year, count(*)"Cyclones" from
  2    (select  to_char(s.when_t, 'yyyy') Year
  3     from    atmosevent_list s
  4     where   exists
  5           ( select  os_year from oscillation_constant_list
  6             where   os_year=to_number(to_char(when_t,'yyyy'))
  7            ) and s.basin=1 and  s.type >=4
  8     )
  9    group by Year
 10  /

Elapsed: 00:00:00.05
```

From the example above, it is obvious which statement has a better performance. However, this environment parameter cannot show us how much time the CPU uses for the issued statement, and how much time used on the I/O, and those detailed information is very important. We solve this problem by using SQL Trace and TKPROF facilities.

Using SQL Trace and TKPROF

The SQL trace and TKPROF facilities enable us to accurately assess the efficiency of the SQL statements.

SQL Trace Facility

The SQL trace facility provides performance information for individual SQL statements. It generates the following statistics for each SQL statement:

- Parse, execute, and fetch counts
- CPU elapsed time
- Physical reads and logical reads
- Number of rows processed
- Misses on the library cache
- Username under which each parse occurred
- Each commit and rollback

We can enable the SQL trace facility for a session or for an instance. When the SQL trace facility is enabled, performance statistics for all SQL statements executed in a user session or in an instance are placed into a trace file.

The additional overhead of running the SQL trace facility against an application with performance problems is normally insignificant, compared with the inherent overhead caused by the application's inefficiency.

TKPROF Facility

After executing the SQL trace, we need to run the TKPROF facility to format the contents of the trace file and to place the output into a readable output file. Optionally, TKPROF can also:

- Determine the execution plans for SQL statements
- Create a SQL script that stores the statistics in the database

TKPROF reports each statement executed with the resources it has consumed, the number of times it was called, and the number of rows it processed. This information lets us easily locate those statements that are using the most resource.

The steps to use the SQL trace and TKPROF facilities:

1. Set initialization parameters for trace file management.
2. Enable the SQL trace facility for the desired session and run your application. This step produces a trace file containing statistics for the SQL statements issued by the application.
3. Run TKPROF to translate the trace file created in Step 2 into a readable output file. This step can optionally create a SQL script that stores the statistics in the database.
4. Interpret the output file created in Step 3.

Step 1: Set Initialization Parameters for Trace File Management

Before enabling the SQL trace facility, one should check the settings of the TIMED_STATISTICS, USER_DUMP_DEST, and MAX_DUMP_FILE_SIZE parameters.

- **TIMED_STATISTICS**

  This parameter enables and disables the collection of timed statistics, such as CPU elapsed time by the SQL trace facility, and the collection of various statistics in the dynamic performance tables. The default value of FALSE disables timing. The value of TRUE enables timing. Enabling timing causes extra timing calls for low-level operations. This is a session parameter.

- **MAX_DUMP_FILE_SIZE**

  When the SQL trace facility is enabled at the instance level, every call to the server produces a text line in a file in your operating system's file format. The maximum size of these files (in operating system blocks) is limited by the initialization parameter MAX_DUMP_FILE_SIZE. The default is 500. If you find that your trace output is truncated, increase the value of this parameter before generating another trace file. This is a session parameter.

- **USER_DUMP_DEST**

  This parameter specifies fully the destination for the trace file according to the conventions of your operating system. The default value for this parameter is the default destination for system dumps on your operating system. This value can be modified with ALTER SYSTEM SET USER_DUMP_DEST=*newdir*. This is a system parameter.

Step 2: Enable the SQL Trace Facility

Enabling the SQL Trace Facility for Current Session

To enable the SQL trace facility for our current session, we use the following command:

```
ALTER SESSION SET SQL_TRACE = TRUE;
```

Alternatively, one can enable the SQL trace facility for a session by using the DBMS_SESSION.SET_SQL_TRACE procedure.

To disable the SQL trace facility, we use the following command:

```
ALTER SESSION SET SQL_TRACE = FALSE;
```

The SQL trace facility is automatically disabled for the tuning session when the application disconnects from Oracle.

Step 3: Format Trace Files with TKPROF

TKPROF accepts as input a trace file produced by the SQL trace facility and produces a formatted output file. Once the SQL trace facility has generated a number of trace files, we can:

- Run TKPROF on each individual trace file, producing a number of formatted output files, one for each session.
- Concatenate the trace files and then run TKPROF on the result to produce a formatted output file for the entire instance.

TKPROF does not report COMMITs and ROLLBACKs that are recorded in the trace file. The syntax for TRPROF is as follows:

```
TRPROF <input_tracefile> <output_filename> Explain =user/password
```

Step 4: Interpret TKPROF OutputTabular Statistics

TKPROF lists the statistics for a SQL statement returned by the SQL trace facility in rows and columns. Each row corresponds to one of the three steps of SQL statement processing. The step for which each row contains statistics is identified by the value of the CALL column:

- PARSE

  This step translates the SQL statement into an execution plan. This step includes checks for proper security authorization and checks for the existence of tables, columns, and other referenced objects.

- EXECUTE

  This step is the actual execution of the statement by Oracle. For INSERT, UPDATE, and DELETE statements, this step modifies the data. For SELECT statements, the step identifies the selected rows.

- FETCH

  This step retrieves rows returned by a query. Fetches are only performed for SELECT statements.

The other columns of the SQL trace facility output are combined statistics for all parses, all executes, and all fetches of a statement. These values are zero (0) if TIMED_STATISTICS is not turned on. The sum of *query* and *current* is the total number of buffers accessed.

- COUNT

  Number of times a statement was parsed, executed, or fetched.

- CPU

  Total CPU time in seconds for all parses, executes, or fetch calls for the statement.

- ELAPSED

  Total elapsed time in seconds for all parses, executes, or fetch calls for the statement.

- DISK

  Total number of data blocks physically read from the data files on disk for all parses, executes, or fetch calls.

- QUERY

  Total number of buffers retrieved in consistent mode for all parses, executes, or fetch calls. Buffers are usually retrieved in consistent mode for queries.

- CURRENT

  Total number of buffers retrieved in current mode. Buffers are retrieved in current mode for statements such as INSERT, UPDATE, and DELETE.

Example:

```
select /*+ first_rows */ Year, count(*)"Cyclones" from --to
response with the first row quickly
 (select  to_char(s.when_t, 'yyyy') Year
  from    atmosevent_list s, oscillation_constant_list o
  where   os_year=to_number(to_char(when_t,'yyyy')) and s.basin=1
          and  s.type >=4
 )
  group by Year
```

```
call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        1      0.00       0.18          0          0          0           0
Execute      1      0.00       0.01          0          0          0           0
Fetch        5      0.03       0.04          0         17          0          47
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total        7      0.03       0.24          0         17          0          47

Misses in library cache during parse: 1
Optimizer goal: FIRST_ROWS
Parsing user id: 29  (CZHANG02)

Rows     Row Source Operation
-------  ---------------------------------------------------
     47  SORT GROUP BY
    466   NESTED LOOPS
   1274    TABLE ACCESS FULL ATMOSEVENT_LIST
    466    INDEX RANGE SCAN (object id 28035)


Rows     Execution Plan
-------  ---------------------------------------------------
      0  SELECT STATEMENT   GOAL: HINT: FIRST_ROWS
     47   SORT (GROUP BY)
    466    NESTED LOOPS
   1274     TABLE ACCESS (FULL) OF 'ATMOSEVENT_LIST'
    466     INDEX (RANGE SCAN) OF 'ENSO_STORM_IDX' (NON-UNIQUE)

***********************************************************************
```

Rows

Statistics about the processed rows appearing in the ROWS column.

- ROWS

  Total number of rows processed by the SQL statement. This total does not include the number of rows processed by subqueries of the SQL statement.

For SELECT statements, the number of rows returned appears for the fetch step. For UPDATE, DELETE, and INSERT statements, the number of rows processed appears for the execute step.

Resolution of Statistics

Timing statistics have a resolution of one hundredth of a second; therefore, any operation on a cursor that takes a hundredth of a second or less may not be timed accurately. Keep this in mind when interpreting statistics. In particular, one should be careful when interpreting the results from simple queries that execute very quickly.

Recursive Calls

Sometimes in order to execute a SQL statement issued by a user, Oracle must issue additional statements. Such statements are called *recursive calls* or *recursive SQL statements*. For example, if you insert a row into a table that does not have enough space to hold that row, Oracle makes recursive calls to allocate the space dynamically. Recursive calls are also generated when data dictionary information is not available in the data dictionary cache and must be retrieved from disk.

If recursive calls occur while the SQL trace facility is enabled, TKPROF produces statistics for the recursive SQL statements and marks them clearly as recursive SQL statements in the output file. You can suppress the listing of recursive calls in the output file by setting the SYS statement-line parameter to NO. The statistics for a recursive SQL statement are included in the listing for that statement, not in the listing for the SQL statement that caused the recursive call. So when you are calculating the total resources required to process a SQL statement, you should consider the statistics for that statement as well as those for recursive calls caused by that statement.

The following examples are two formatted SQL statements with TRPROF:

Example:

Structure 1

```
select /*+ first_rows */ Year, count(*)"Cyclones" from --to
response with the first row quickly
 (select  to_char(s.when_t, 'yyyy') Year
  from    atmosevent_list s, oscillation_constant_list o
  where   os_year=to_number(to_char(when_t,'yyyy')) and s.basin=1
          and  s.type >=4
 )
  group by Year
```

```
call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        1      0.00       0.18          0          0          0           0
Execute      1      0.00       0.01          0          0          0           0
Fetch        5      0.03       0.04          0         17          0          47
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total        7      0.03       0.24          0         17          0          47

Misses in library cache during parse: 1
Optimizer goal: FIRST_ROWS
Parsing user id: 29  (CZHANG02)

Rows     Row Source Operation
-------  ---------------------------------------------------
     47  SORT GROUP BY
    466   NESTED LOOPS
   1274    TABLE ACCESS FULL ATMOSEVENT_LIST
    466    INDEX RANGE SCAN (object id 28035)


Rows     Execution Plan
-------  ---------------------------------------------------
      0  SELECT STATEMENT   GOAL: HINT: FIRST_ROWS
     47   SORT (GROUP BY)
    466    NESTED LOOPS
   1274     TABLE ACCESS (FULL) OF 'ATMOSEVENT_LIST'
    466     INDEX (RANGE SCAN) OF 'ENSO_STORM_IDX' (NON-UNIQUE)

***********************************************************************
```

Structure 2

```
select /*+ first_rows */ Year, count(*)"Cyclones" from
  (select  to_char(s.when_t, 'yyyy') Year
   from    atmosevent_list s
   where   exists
      (select   os_year from oscillation_constant_list
       where    os_year=to_number(to_char(when_t,'yyyy'))
                and   s.basin=1 and  s.type >=4
      )
   )
group by Year
```

```
call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        1      0.00       0.05          0          0          0           0
Execute      1      0.00       0.00          0          0          0           0
Fetch        5      0.06       0.07          0       1286          0          47
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total        7      0.06       0.12          0       1286          0          47

Misses in library cache during parse: 1
Optimizer goal: FIRST_ROWS
```

```
Parsing user id: 29  (CZHANG02)

Rows     Row Source Operation
-------  ---------------------------------------------------
     47  SORT GROUP BY
    466   FILTER
   1274    TABLE ACCESS FULL ATMOSEVENT_LIST
    464    FILTER
    464     INDEX RANGE SCAN (object id 28035)



Rows     Execution Plan
-------  ---------------------------------------------------
      0  SELECT STATEMENT   GOAL: HINT: FIRST_ROWS
     47   SORT (GROUP BY)
    466    FILTER
   1274     TABLE ACCESS (FULL) OF 'ATMOSEVENT_LIST'
    464     FILTER
    464      INDEX (RANGE SCAN) OF 'ENSO_STORM_IDX' (NON-UNIQUE)
```

We examined the trace file carefully, and chose the statement that consumes less resource and has the better overall performance.

Section 6

PHRLM Quality Assurance

Section 6.1

Coding Guide Lines

6.1.1 About the Coding Guidelines

This document is prepared as a part of the PHRLM project. All the developers involved in the system development are asked to read and follow the instructions given in here. In general this document may be read as a guide to writing robust and readable codes. Examples given in here are mainly focused on programs written in C flavors, but the content is generally applicable for programs written in any other programming language.

6.1.2 File Organization

6.1.2.1 Source files

- Keep your classes/files short, don't exceed 2000 lines of code
- Divide your code up, make structures clearer
- Put every class in a separate file and name the file like the class name. This convention makes things much easier.

6.1.2.2 Directory Layout

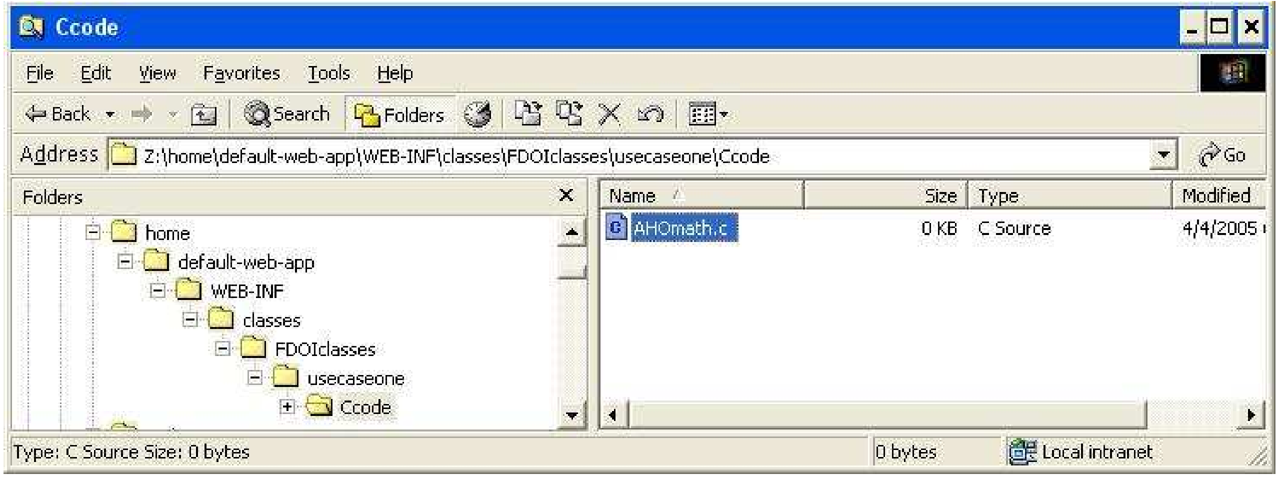
Developer's own structure

- Create a directory for every use case and keep all the related codes in that.
- For each major revision create a subfolder with the revision number.
- Do the CVS before any change and keep your own backup always.

System Directory Structure

- All the codes C and JAVA codes: Create a subdirectory under the use case name inside /home/irene1b/oracle/j2ee/home/default-web-app/WEB-INF/classes/FDOIclasses/
- Save the latest copy of the codes there
- All the JSP files has to be saved in /home/irene1b/oracle/j2ee/home/default-web-app/FDOI/*useCaseName*
- Do CVS

Example:



Address: Z:\home\default-web-app\WEB-INF\classes\FDOIclasses\usecaseone\Ccode

Name	Size	Type	Modified
AHOmath.c	0 KB	C Source	4/4/2005

Type: C Source Size: 0 bytes | 0 bytes | Local intranet

6.1.3 Code Indentation

6.1.3.1 Wrapping Lines

When an expression will not fit on a single line, break it up according to these general principles:

- Break after a comma
- Break after an operator
- Prefer higher-level breaks to lower-level breaks
- Align the new line with the beginning of the expression at the same level on the previous line

Example: Breaking up method calls:

```
longMethodCall(expr1, expr2,
            expr3, expr4, expr5);
```

Example: Breaking an arithmetic expression:

PREFER:

```
var = a * b / (c - g + f) +
        4 * z;
```

BAD STYLE – AVOID:

```
var = a * b / (c - g +
     f) + 4 * z;
```

The first is preferred, since the break occurs outside the parenthesized expression (higher level rule). Note that you indent with tabs to the indentation level and then with spaces to the breaking position in our example this would be:

```
> var = a * b / (c - g + f) +
> ........4 * z;
```

Where '>' are tab chars and '.' are spaces.

6.1.3.2 White Spaces: Don't use spaces for indentation - use tabs!

An indentation standard using spaces never was achieved. Always use tabs. Tab characters have some advantages:

- Everyone can set his or her own preferred indentation level
- It is only 1 character and not 2, 4, 8 … therefore it will reduce typing (even with smart indenting you have to set the indentation manually sometimes, or take it back or whatever)
- If you want to increase the indentation (or decrease), mark one block and increase the indent level with Tab with Shift-Tab you decrease the indentation. This is true for almost any text editor.

Here, we define the Tab as the standard indentation character.

6.1.4 Comments

6.1.4.1 Block Comments

- When you wish to use block comments you should use the following style:

  ```
  /* Line 1
  * Line 2
  * Line 3
  */
  ```

  As this will set off the block visually from code for the (human) reader.

- Alternatively you might use this old fashioned C style for single line comments, even though it is not recommended. In case you use this style, a line break should follow the comment, as it is hard to see code preceded by comments in the same line:

  ```
  /* blah blah blah */
  ```

- In case, this kind of block comment is not applicable, it is recommended to follow a similar standard.

6.1.4.2 Single Line Comments

- You should use the // comment style to "comment out" code. It may be used for commenting sections of code too.
- Single line comments must be indented to the indent level when they are used for code documentation.
- A rule of thumb says that generally, the length of a comment should not exceed the length of the code explained, as this is an indication of too complicated, potentially buggy, code.

6.1.4.3 In line File Documentation

- At the beginning of the each file the purpose of the file should be documented using the following template.
- For each code revision, <Revision History> has to be updated

```
//=====================================================================
// <Filename>                                       <Creation Date>
// WProbability.cc                                   05/13/2004

// <description>
// Calculates wind speed probabilities.
// Input : Surface corrected wind speeds (3S gust) from the WSC module
// Output: Probabilities of wind speeds from 20-300mph, interval is 4
mph

// <Revision History>
// <date>         <developer>   <Description>
// 05/19/2004      kwick001      initial code
//=====================================================================
```

6.1.4.4 In line Function Documentation

- At the beginning of the each file the purpose of the file should be documented using the following template.
- For each revision <Revision History> has to be updated

```
//=====================================================================
// <Function>                                   <Creation Date>
//  count_zip                                     02/04/2005
//
// <Parameters>
//  none
//
// <Return>
//  Number of lines in the "zipcodes.txt" file
//
// <Description>
//  read the zipcodes.txt file count the number of lines in the file
//
// <Revision History>
// 02/04/2005     kwick001    generate initial code
//=====================================================================
```

6.1.5 Variable Declarations

6.1.5.1 Number of Declarations per Line

- One declaration per line is recommended since it encourages commenting. In other words,

  ```
  int level; // indentation level
  int size; // size of table
  ```

- Do not put more than one variable or variables of different types on the same line when declaring them.

  Example:

  ```
  int a, b; //What is 'a'? What does 'b' stand for?
  ```

- The above example also demonstrates the drawbacks of non-obvious variable names. Be clear when naming variables.

6.1.5.2 Initialization

- Try to initialize local variables as soon as they are declared. For example:

  ```
  int val = 10;
  ```

6.1.6 Statements

6.1.6.1 Simple Statements

Each line should contain only one statement.

6.1.6.2 Return Statements

A return statement should not use outer most parentheses.

Don't use:

```
return (n * (n + 1) / 2);
```

Use:

```
            return n * (n + 1) / 2;
```

6.1.6.3 If, if-else, if else-if else Statements

if and if-else statements should look like this:

```
if (condition)
{
            DoSomething();
            ...
}
if (condition)
{
            DoSomething();
            ...
}
else
{
            DoSomethingOther();
            ...
}
```

6.1.6.4 For Statements

A for statement shoud have following form :

```
for (int i = 0; i < 5; ++i)
 {
            DoSomething();
            ...
}
```

Note: Generally use brackets even if there is only one statement in the loop.

6.1.6.5 While Statements

A while statement should be written as follows:

```
while (condition)
{
        DoSomething();
        ...
}
```

6.1.6.6 Try-catch Statements

A try-catch statement should follow this form:

```
try {
...
} catch (Exception e) {
...
}
-OR -
try {
...
} catch (Exception e) {
...
} finally {
...
}
```

6.1.7 White Space

6.1.7.1 Blank Lines

Blank lines improve readability. They set off blocks of code which are in themselves logically related. Two blank lines should always be used between:

- Logical sections of a source file
- Class and interface definitions (try one class/interface per file to prevent this case)
  One blank line should always be used between:
- Functions/ methods
- Logical sections inside a method to improve readability. Note that blank lines must be indented, as they would contain a statement. This makes insertion in these lines much easier.

6.1.7.2 Inter-term spacing

- There should be a single space after a comma or a semicolon.

Example:

```
Use:     TestMethod(a, b, c); or TestMethod( a, b, c );
Don't use: TestMethod(a,b,c)
```

- Single spaces surround operators (except unary operators like increment or logical not)

Example:

```
Use:        a = b;
Don't use:  a=b;
Use:        for (int i = 0; i < 10; ++i)
Don't use: for (int i=0; i<10; ++i)  //or  for(int i=0;i<10;++i)
```

6.1.8 Naming Conventions

6.1.8.1 Naming Guidelines

- Use Camel Casing: This convention capitalizes the first character of each word except the first one.
  E.g. testCounter.
- Do use descriptive names, which should be enough to determine the variable meaning and it’s type. But prefer a name that’s based on the parameter’s meaning.
- Remember: a good variable name describes the semantic not the type.
- An exception to this rule is GUI code. All fields and variable names that contain GUI elements like button should be post-fixed with their type name without abbreviations.

Example:

```
System.Windows.Forms.Button cancelButton;
System.Windows.Forms.TextBox nameTextBox;
```

6.1.8.2 Variable Names

- Counting variables are preferably called i, j, k, l, m, n when used in 'trivial' counting loops.

Note: Indexer variables generally should be called i, j, k etc. But in some cases, it may make sense to reconsider this rule. In general, when the same counters or indexers are reused, give them meaningful names.

6.1.8.3 Method Names

- Name methods with verbs or verb phrases.

6.1.9 Reference

The “C# Coding Style Guide” by: Salman Ahmed is used as a template for this guideline development.

Section 6.2

Data Validation and Verification

6.2.1 About the Document

This document is prepared as a part of the PHRLM project. The primary audience for this guidance is practitioners directly involved in implementing or managing data verification or data validation efforts. This guidance should provide this audience with a conceptual overview on "how-to" verify and validate the data. All the personals involved in the implementing or managing data are asked to read and follow the instructions given in here.

6.2.2 Introduction

If the information being used is not credible, there is no point in using it. Decisions based on inaccurate or unreliable data can adversely affect the decision making process. Data verification and validation is used to evaluate whether data has been generated according to specifications, satisfy acceptance criteria, and are appropriate and consistent with their intended use.

6.2.2.1 Data Verification

Data verification is a systematic process for evaluating performance and compliance of a set of data when compared to a set of standards to ascertain its completeness, correctness, and consistency using the methods and criteria defined in the project documentation [1].

6.2.2.2 Data Validation

Data validation follows the data verification process and uses information from the project documentation to ascertain the usability of the data in light of its measurement quality objectives and to ensure that results obtained are scientifically defensible [1].

6.2.3 Procedures

In the context of PHRLM project data validation and verification is mostly a one-time process. Under mentioned procedures may not applicable in all the instances. But in general most of these procedures are applicable and should be followed by the developers. In case all these procedures are not applicable, it is advised to develop your own methods and properly document the procedure followed.

- **Format check**

  Check if the data is in the right format. This can be done manually or using any commercially available data manipulating tools such as Excel or Access. Mainly data is received in text file format. If the input data set is too large do the format test on randomly selected files.

- **Length check**

Check the data isn't too short or too long. For this check the whole file and then check the expected length of the each field. This is applicable to text fields only.

- **Range check**

  Checks a number isn't too big or too small. For an example, a zip code has to be greater than 0 and less than 40000.
- **Presence check**

  Checks that a field has been entered.

Once the above checks are completed and successful it is ready to be imported to the system. It is recommended to use data manipulating software or a simple program written by the developer for the data importing process rather than manual entry. If it is unavoidable you may use manual entry. In either case it is recommended to double check the imported data. The above mentioned steps can be repeated and in addition following tests are recommended.

- **Double entry**

  Type the data in twice and compare the two copies. This can take much more time and means higher costs.
- **Proofreading data**

  This method involves somebody checking what is in the system is the same as the original input. Always make sure to make a copy of the data in the data after importing them to the system and give this copy and the original copy of the data to a person who is not involved in the data manipulating process to compare and certify the correctness.

6.2.4 Data Security and Integrity

This section describes precautionary measures that must be taken in the event that computer malfunctions, natural disasters, or human error or actions occur that could affect collected data.

- **Duplicate copies or back-up system for data**

  Florida International University, School of Computer Science takes regular backups generally every Friday. All the databases and data files are included in the backup. Developer must make sure that they store all the data in those places that are backed up.

- **Data security protocols are in place and effective**

  Firewalls/password protection, access levels, etc. are established.

  Accountability for data integrity clearly rests with the person entering the data, and the responsible program specialist and manager. Only those who are skilled and trained in proper data handling procedures are allowed the direct access to the database.

6.2.5 References

[1] EPA Quality System, Quality Management Tools - Data Verification and Validation (http://www.epa.gov/quality/vandv.html)

Section 6.3

Model Maintenance and Revision

6.3.1 Model Maintenance and Revision

PHRLM has developed a clearly documented policy for model revision with respect to methodology and data. Any enhancement to the model that results in a change in any Florida residential hurricane loss costs also results in a new model version number. PHRLM uses version control and tracking software to identify all errors, as well as modifications to code, data, and documentation.

1. PHRLM employs consistent methods for data and documentation control for all software development, including both server and client programs (written in C++ and Java). The installation date, program specification, personnel involved, current version number and date of most recent changes are documented for the individual components in the system.

2. The data and model is maintained and updated each year. At each year, the ZIP Code information is updated to reflect the most recent changes within the past 12 months. In particular, the ZIP Code boundaries and the centroids are updated, and using this updated information, the ZIP Code related features are updated, including distance from the coastline, population centroid, elevation, and surface roughness, etc. The historical hurricane data (for Atlantic Basin) is periodically updated to take into account new hurricane events.

3. Updates to ZIP codes, historical meteorological events, and the related characteristics will also trigger the updates to the model results. Whenever the new data or new modeling methodologies become available which results in a non-trivial improvement in the modeling results, a new model version number is assigned. The PHRLM project development team will maintain, archive, and document the features of each model version.

4. The first version is released with version number 1 (*PHRLM Version 1*). Version number will be incremented by one at each yearly update to the system. Each time a new version is released mid of the year the version number is incremented by one decimal fraction.

5. When a new model version is released, a release document, with detailed documentation for users, and the programs and data that are used in this release will be packaged and tested by crosschecking. Standard test cases are also packaged with the release, to allow later verification. This assures the correctness and consistency of each release.

6. PHRLM's software development team employs source revision and control software for all software development. In particular, PHRLM employs Concurrent Versioning System (CVS), an accepted and effective system for managing simultaneous development of files. It is in common use in large programming projects to track modifications of all source code. CVS maintains a record of the changes to each file, allowing the user to revert to a previous version, merge

versions, and track changes. This software is able to record the information for each file, the date of each change, author of the change, file version, and the comparison of the file before and after the change.

7. The software development process is carefully monitored by designated personnel using CVS tracking tools and procedures. For example, '*cvs annotate*' provides a quick of finding who made what changes and when by displaying the last change information for each line of a file in the repository. Such information includes the revision number for the last change of each line, the user, the date, and the contents of the line. The CVS history file records commits, merges, conflicts, tagging, updates (to the working directory), additions, deletions, and modifications of files in the repository. The *loginfo* file controls where log information for '*cvs commit*' is sent. This allows the project manager to keep track of the changes made by the development team, and to maintain a central log of the project progress.

8. PHRLM employs an access control mechanism that allows only authorized user accounts to modify parts of the hierarchy in the repository. Authorization control is for commits only; everyone can check out any part of the repository. That is to say, for user accounts other than the designated ones, they do not have write access to the restricted area. An access list is maintained to record all the access rights and responsibilities of each CVS user. Therefore, in PHRLM, general users can submit patches to a/the maintainer (authorized user), and the maintainer will commit changes directly to the repository. In the future, we plan to implement an easy-to-use CVS commit log search interface for scanning CVS commit logs form any part of the repository over any time period, for all users or for a particular user.

It is required that all CVS users need to use **ssh** to access a repository on a remote machine. This is set in CVS's configuration file. In addition, the development team members are required to add meaningful change-notes for the appropriate files. By doing that, it is much easier to locate the correct version in roll back operations when needed.

Section 6.4

PHRLM Testing Procedures

6.4.1 Software Testing Procedures

PHRLM software testing and verification is done in three stages.

(i) Code inspection and the verification by the code developer

Code developer should carry out sufficient amount of testing on the code and should not deliver the code until and unless he/she is convinced of proper functionality and robustness of the code.

In this level of testing should code-level debugging, walk through the code to ensure proper flow, inspection of internal variables through intermediate output printing and error logging, use of exception handling mechanisms, calculation cross checks and verification of the output against sample calculations provided by the system modeler. It is the developer’s responsibility to collect at least one sample calculation from the system modeler and to compare the results against the results generated through the code.

(ii) Verification of results by the person who developed the system model.

Once the first level of testing is done the developer should send the sample inputs and the generated results back to the modeler. Then the system modeler should double-check the results against his/her model. Code is not put in to the production environment with out the ‘OK’ from the modeler.

(iii) Review and extensive testing of the code by external group of software engineers.

System is rigorously checked for correctness, precision of the output and robustness & stability of the whole system. Calculations are performed outside the system and compared against the system generated results to ensure the system correctness. Extreme and unexpected inputs are given to the system to check the robustness. Wide series of test cases are developed to check the stability and the consistency of system.

Unit testing, Regression testing, and Aggregation testing (both white-box and black-box) should be performed and documented.

Any flaws in the code are reported to the developer and the bug corrected code is again sent to the tester. The tester should perform unit testing again on the modified units and also Regression testing should be carried to check if the modification affects any other parts of the code.

Note: Please refer to the testing document for more details.

Section 6.5

Code Count Tables

Use Case I - Annual Hurricane Occurrence (AHO)

Filename:	Source:	Comment:	Both:	Blank:	Total:
fitDistriBean.h	18	22	0	5	45
AHOmath.c	417	149	22	95	683
calcMVSBean.java	39	84	6	24	153
dataEntry.java	22	97	0	20	139
fitDistriBean.java	81	150	1	56	288
getDBean.java	239	193	7	66	505
myplot.java	87	72	4	32	195
OutputFormatBean.java	11	29	0	12	52
plotBean.java	7	11	0	2	20
simulationBean.java	40	122	11	32	205
storeAHOBean.java	72	115	7	38	232
banner.jsp	8	0	0	0	8
DSSelection.jsp	41	0	0	4	45
plotSimulation.jsp	109	18	0	19	146
simuSelection.jsp	85	12	0	12	109

Use Case II - Storm Genesis Time (SGT)

Filename:	Source:	Comment:	Both:	Blank:	Total:
SGTBean.h	14	13	0	3	30
sgtmath.c	302	106	2	222	632
getSGTDataBean.java	39	35	2	28	104
SGTBean.java	23	97	0	15	135
SGTDataEntry.java	140	63	10	44	257
SGTbanner.jsp	8	0	0	0	8
SGTindex.jsp	45	0	0	3	48
SGTsimulation.jsp	91	0	1	10	102

Use Case V - Wind Speed Correction (WSC)

Filename:	Source:	Comment:	Both:	Blank:	Total:
WSCCalVamphBean.java	269	202	3	80	554
WSCSpeedCheckBean.java	78	99	5	29	211
WindSpeed.jsp	75	580	1	10	666
WSCindex.jsp	210	0	0	31	241

Use Case VI - Wind Speed Probability (WSP)

Filename:	Source:	Comment:	Both:	Blank:	Total:
WPHeader.h	7	10	0	3	20
WPStruct.h	103	14	6	31	154
WProbability.cpp	520	198	59	351	1128
WPStruct.cpp	7	10	0	10	27
WPUtils.cpp	6	11	0	2	19
WSPCalc.jsp	123	98	2	11	234
WSPtask.jsp	154	0	0	24	178

General Insurance Loss Module (ILM)					
Filename:	Source:	Comment:	Both:	Blank:	Total:
ILMInputs.h	437	196	28	65	726
InsuranceLossModel.cpp	5	4	0	1	10
InsuranceLossModel.h	1082	258	42	138	1520
Scenario Insurance Loss Module (ILM)					
Filename:	Source:	Comment:	Both:	Blank:	Total:
ILMInputs.h	431	176	26	53	686
InsuranceLossModel.cpp	11	18	0	6	35
InsuranceLossModel.h	1133	212	45	148	1538

Use Case IV - Wind Field Model					
Filename:	Source:	Comment:	Both:	Blank:	Total:
dus.pro	12	3	1	3	19
fixshots15.pro	27	4	1	9	41
gemf.m	5	2	0	0	7
gemf.pro	10	5	1	5	21
gemfplex.pro	17	4	1	4	26
genstrex.m	11	7	0	3	21
lltoxy.pro	14	5	1	1	21
mnrdsg.pro	6	2	1	2	11
mnrdu.pro	6	2	1	2	11
obc.m	7	3	0	0	10
onefix.m	27	12	0	7	46
pkwinds.pro	75	28	1	20	124
reach.pro	7	0	1	0	8
rsdsg.pro	22	5	1	6	34
rsdu.pro	21	5	1	6	33
selset.m	3	3	0	0	6
sgdvs.pro	14	6	1	6	27
shift.m	6	2	0	0	8
suv.pro	31	10	1	9	51
tek.m	7	2	0	0	9
thinner.pro	51	12	1	12	76
track.pro	69	15	1	16	101
udvs.pro	14	6	1	6	27
usadv.m	7	5	0	4	16
usg.pro	28	9	2	8	47
usnoadv.m	6	6	0	4	16
vghgen.pro	19	5	1	5	30
zmar2zot.pro	12	6	1	5	24

Engineering Module

Filename:	Source:	Comment:	Both:	Blank:	Total:
Site-Built\ContUtilities_Validation_Prog_112704.m	405	96	15	176	692
Site-Built\Final_VM_Plot_Prog_101704.m	140	21	5	58	224
Site-Built\Matrix_Weight_Prog_1212005_final.m	923	76	101	296	1396
Site-Built\Vulnerability_Fragility_Plot_Prog_111904_type1.m	386	22	6	85	499
Site-Built\Vulnerability_Prog_020405.m	1108	373	14	382	1877

Montecarlo Codes

Filename:	Source:	Comment:	Both:	Blank:	Total:
capacity_manuf_house.m	54	28	7	19	108
capacity_opening.m	103	8	6	16	133
capacity_r2w.m	61	17	6	7	91
capacity_roofcover.m	17	4	2	7	30
capacity_sheathing.m	17	4	2	8	31
capacity_wall.m	118	20	8	21	167
capacity_wall_sheathing.m	15	6	3	6	30
debris_model_input.m	29	91	2	5	127
missile_impact.m	40	21	1	8	70
pressures.m	6	3	0	7	16
r2w_conn_uplift.m	76	4	5	5	90
r2w_conn_uplift_hip5638.m	127	15	2	22	166
r2w_conn_uplift_hip5644.m	127	17	2	20	166
r2w_conn_uplift_hip6038.m	125	15	2	23	165
r2w_conn_uplift_hip6044.m	129	18	2	20	169
redist_gable.m	12	2	0	3	17
rooflayout6044.m	106	37	13	28	184
wall_loading.m	100	45	13	16	174
window_pressure_check.m	174	20	101	11	306

Use Case III - Storm Track Module

Filename:	Source:	Comment:	Both:	Blank:	Total:
stormgen.f	823	274	275	203	1575
genpdf.f	1142	403	244	280	2069
genpdf.h	13	34	5	1	53

Section 7

Security

7.1 Security Procedures

PHRLM has implemented security procedures for access to code, data, and documentation that are in accordance with standard industry practices. PHRLM employs a number of physical and electronic security measures to protect all code, data and documentation against both internal and external potential sources of damage.

Summary:

1. The application server (IRENE) and the database server (ANDREW), as shown in Table 7.1, are considered "mission critical servers" (see its definition in the Security Procedures Manual, Section II) and are kept and maintained in a secure server room which limits non-authorized access. Access to the server room is granted by electronic key card and is limited to essential personnel only. All servers and desktops are protected with Norton Antivirus software.

Table 7.1. PHRLM servers

HOSTNAME	Operating System	Purpose
andrew.cs.fiu.edu	Solaris 8	DB Server and File Storage
irene.cs.fiu.edu	Red Hat Linux	Application Server and File Storage

2. As outlined in the "Security Procedures Manual", section IV part 6, backups are performed on a daily basis and are kept for six weeks. Nightly backups of all UNIX data disks and selected Windows data disks (at user request) are performed over the network onto Exabyte Mammoth M2 tapes. Full dumps are taken periodically (it works out to every 2-3 weeks) and incrementals are taken daily between them. Off-site backups are performed at the end of every semester and stored off-site in the PC building at the University Park Campus, which is the Monroe County hurricane shelter and has emergency power and climate control.

3. The tape drives have built in diagnostics and verification to ensure that the data is written correctly to tape. This ensures that if the tape is written successfully, it will be readable, provided no physical damage occurs to the tape. The off-site backup procedure performs a level 0 (full) dump of every disk in the department. This means that each disk in the department will be backed up to tape in its entirety. The dumps can be restored from tape, preserving the original file structure and all permissions. All read errors during the backup process are reported, so if a file system fails to dump correctly, the dump can be re-done. In the past, we have successfully restored data from both our offsite and daily backups for many times, and no problems have been occurred.
4. In case of disasters, we have implemented a set of preparation procedures and recovery plans as outlined in "FIU SCS Hurricane Preparation Procedures". The computing equipment associated with PHRLM will be secured and safeguarded by designated personnel such as the Lab Manager when the hurricane warning is issued. When hurricane warning is lifted the lab manager will return to FIU and take in charge of system recovery.

5. Security policies are documented and all PHRLM personnel are trained in security requirements and procedures. When personnel (Graduate Research Assistants supported by PHRLM, Professional Programmers, etc.) leave the PHRLM project, they are required to sign non-disclosure agreements to not keep nor disclose any confidential information/documents at the proprietary level and above. For details, please check the attached documents.

6. Any sensitive or confidential data (insurance data, for example) are kept on a local, unshared disk on a system which has user access control and requires a login. Screen locks are used whenever the machine is not attended. Backups are done for that disk at daily basis. In addition, sensitive data should never be sent via unencrypted email.

7. Access to all PHRLM computers/workstations is controlled by passwords. A screen/keyboard lock or login screen should be active on all machines when they are not in use. A designated project manager is responsible for providing initial approval for a PHRLM computer account and for notifying the computing center of a change in status of users.

8. In addition, for system security and reliability purpose, we also deploy a development environment besides the production environment. Modifications to the code and data are done in the development environment and tested by in-house developers. The final production code and data can only be checked into the production environment by authorized personnel. Baseline tests are always run to ensure the model is functioning properly and reproducing known results.

9. The models resulted from PHRLM project can only be used by authorized users. Authorized user accounts are created by the project manager. The models are accessible to authorized users via web applications using JSP. The source code is stored in server side and cannot be tampered with by unauthorized users. The output of the models is always coupled with the analysis parameters and other information needed to reproduce the analysis results, which are documented in each technical report to maintain the information integrity.

10. Passwords will be kept private in a not shared disk. Passwords will consist of a minimum of 6 alphanumeric characters (no common names or phrases). Passwords will be changed every 120 days; this will be enforced by an automatic expiration procedure to prevent repeated or reused passwords. User accounts will be frozen after 3 failed logon attempts. All erroneous password entries will be recorded in an audit log for later inspection and action, as necessary. Sessions will be suspended after 30 minutes (or other specified period) of inactivity and require the password to be reentered. Successful logons should display the date and time of the last logon and logoff. All user logons will be recorded for future audit.

For detailed information please check the following documents:

7.2 FIU SCS Computer and Networking Security Procedures Manual

Draft Revised: 08/23/2002

I. Responsibilities and Scope of Work

The role of our system administrators is to provide technical support for our diverse network and computing systems, technical consulting services for faculty and researchers, and education for users on the use of our systems. System administrators are responsible for the day-to-day operation and maintenance of our systems and networking environment which include, but not limited to: Operating system installation, configuration, updates, security, monitoring and automation of services. The systems administers goal is to provide a reliable, state-of-the-art computing environment for instructional and research use. The following positions are assigned the computer and networking security responsibilities for the School of Computer Sciences computer and networking facilities.

1. Associate Director for Computing:

Responsible for the policy and procedures established by the School of Computer Science to assure the security of employee and student information and intellectual property, to minimize loss of staff and student productivity due to computer and networking security violations and educate staff and students on “best practices” to secure their critical data. Consults with the School Director and SCS faculty on computer and networking security requirements and directs the development of the policy and procedure needs with the Systems and Networking Group Manager. Reports to the SCS director and other University Management security violations and liaisons with law enforcement should the violation require such interaction.

2. Systems and Networking Group Manager:

Responsible for the engineering of computer and networking security services for the School of Computer Science. The Group Manager establishes day-to-day procedures necessary to maintain computer and networking security for the School. Makes recommendations to the Assoc. Director on policy and procedures and deploys commercial, open-source or in-house developed technology to implement computer and networking security policies. The Group Manager will liaison with other technology groups on campus to coordinate security efforts.

3. Systems/Networking Administrator:

Responsible for day-to-day monitoring of security reports and logs and responds to security alerts as indicated in the SCS Computer and Networking Security Procedures Manual. The administrator reports security anomalies to Group Manager and conducts security investigation, collecting additional log information, correlating security data, and providing recommendations as directed by Group Manager.

II. Definitions:

“Computer Account”: A username and password credential used to identify an authorized user of FIU/SCS computer and networking resources.

“Unauthorized Use”: term used to describe when an unauthorized person utilizes computer and/or networking resources restricted by FIU/SCS.

“Authentication”: The process of providing correct computer account credentials to obtain access to FIU/SCS computer or networking resources.

“Security incident”: Any unauthorized utilization of FIU/SCS computer and networking resources.

“Mission Critical Server”: Any computer server which provides the majority of SCS users computer services which if down would result in 8 hours of lost user productivity.

III. Policies:

1. All computer and networking resource usage on the FIU/SCS network must be authenticated.

2. Each computer user must be assigned one unique computer account. Exceptions may be made in order to manage software/hardware services but account ownership is documented.

3. Critical computer systems and networking devices are to be monitored regularly to insure security is maintained.

4. Root access to the primary trusted system “goedel.cs.fiu.edu” is by permission of the Assoc. Director for Computing only. All work on the primary trusted system must be conducted via the “sudo” utility. No root console logins on goedel are authorized except for scheduled installations and emergency work (which is disclosed to the A.D).

5. All security incidents will be log in the FIU SCS Computer and Security Activities Log. Depending on severity security incidents will be reported to the SCS Director and/or other FIU management.

6. Computer or networking devices whose security has been compromised may be disconnected from the FIU SCS network until the system security is restored.

7. Computer accounts whose security has been compromised may be disabled until the appropriate credentials are properly reassigned.

8. All computer system operating systems will be maintained with critical security patches as indicated by OS provider and/or security community.

9. Mission critical servers will be maintained in a physical location, which limits non-authorized access. Access to the server room is granted by electronic key card and is limited to essential personnel only.

10. If a mission critical server goes down the server room security system will immediately page the system administrators to report the incident.

11. The School maintains anti-virus software on all networked computers and regularly updates the anti-virus software.

12. The School's security policies shall be consistent with those security policies which govern the State of Florida and Florida International University Academic Affairs.

13. Student must adhere to the FIU Code of Computing Practice, a policy produced by University Technology Services.

14. If a computer security violation is suspected, the Systems and Networking Group Manager and/or the Associate Director for Computing have the authority to investigate the suspected violation by reviewing and modifying system and user files in an effort to ascertain the extent of the violation and restore system security.

If a violation of the computer security policy has occurred, the Assoc. Dir. for Computing notifies the SCS Director, Assoc. Director(s), and UTS Security Officer of the security violation. Once the appropriate steps are taken to restore security, the SCS Director is notified and a public statement is made to the SCS user community of the incident as deemed appropriate by the SCS Director.

15. SCS systems which require presentation of credentials should use the appropriate encrypted channels (SSL, SSH, etc). SCS will be discontinuing application/service support of unencrypted logins as we are able to migrate legacy applications/services.

7.3 FIU SCS Hurricane Preparation Procedures

During Hurricane season (June-November) the Director (Associate Director or designee) may issue an alert to the staff to prepare for an impending storm. The Lab Manager may use his master key (or one will be made available to him/her) to enter Faculty offices to begin preparations to safeguard computing equipment. The Director/designee will contact the Lab Manager with final instructions to begin securing the School's equipment.

In the case where the Director/designee is unable to contact the Lab Manager, the Lab Manager shall report to campus when the National Weather Service issues a **Hurricane Watch** (<36 hours before land-fall). If necessary, the Lab Manager will call in additional personnel to assist securing equipment. (If the Lab Manager is out-of-town, an alternate staff member will be designated to respond.)

If the hurricane passes without major incident to our area and the hurricane warning is lifted the lab manager will return to FIU. Additional personnel may be requested to assist in restarting systems. Damage assessment will occur during this period.

If, however, the area suffers severe damage the ability of lab personnel to return to FIU may be hampered. Communication with the above mentioned will be attempted. If that fails an attempt to reach the campus within 72 hours after the lifting of the Hurricane Warning will be made. If it is safe to enter the building damage assessment will occur and the systems restarted.

7.4 Non-Disclosure Agreement

NON-DISCLOSURE AGREEMENT

The undersigned hereby agrees and acknowledges:

1. That during the course of my employment at Public Hurricane Risk and Loss Model (PHRLM) there may be disclosed to me certain confidential information consisting but not necessarily limited to:

(a) Technical information: Methods, processes, formulae, compositions, systems, techniques, inventions, machines, computer programs and research projects.

(b) Business information: Insurance data, customer lists, pricing data, and financial data.

2. I agree that I shall not during, or at any time after the termination of my employment with the PHRLM, use for others, or myself or disclose or divulge to others including future employees, any confidential information, or any other proprietary data of PHRLM in violation of this agreement.

3. That upon the termination of my employment from PHRLM:

(a) I shall return to the project manager all documents and property of PHRLM, including but not necessarily limited to: drawings, blueprints, reports, manuals, correspondence, computer programs, business data, and all other materials and all copies thereof relating in any way to PHRLM, or in any way obtained by me during the course of employment. I further agree that I shall not retain copies, notes or abstracts of the foregoing.

(b) This agreement shall be binding upon me and my personal representatives and successors in interest, and shall inure to the benefit of PHRLM, its successors and assigns.

Signed this on ________/ 20____(MM/DD/YYYY)

________________________________   ________________________________
Project Manager or Professor   Employee Name (Print) / Signature

Section 8

System Hardware and Software Configurations

8.1 System Architecture

System is implemented in three-tier architecture. Following diagram gives a big picture of the system software arrangement.

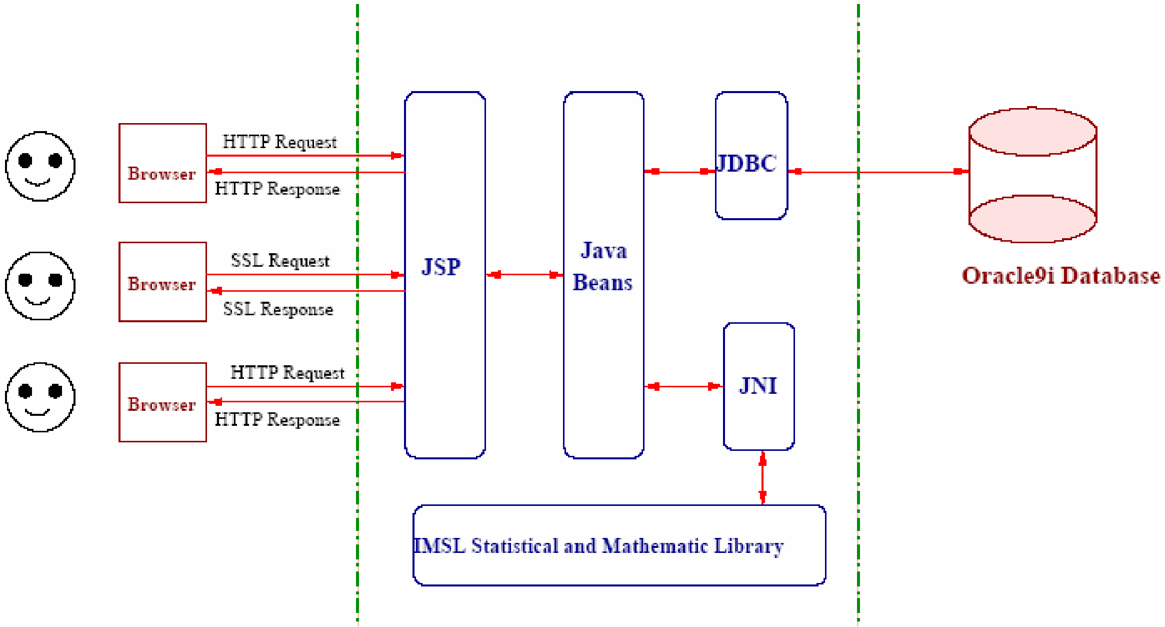


Figure 8.1: System Architecture

8.2 Software List

Java 1.5
JDK 1.3.1
IMSL library CNL 5.0
OC4J v1.0.2.2.1
Oracle 9i AS 9.2
JNI 1.3.1
IDL Version 6.0
MapInfo Data – 2004 Dynamic Census Block Group & Zip code Boundaries
Math works Real-Time Workshop 6.2
Geronesoft's Code Counter Pro Software 1.23
Matlab 7.0

8.3 Hardware Configuration

PHRLM is a large-scale system, which is supposed to store, retrieve, and process huge amount of hurricane historical data and the simulated data. And also intensive computations are required for hurricane analysis and projection. Correspondingly, high-speed CPU and large RAM are necessary.

The hurricane data may be regularly updated and the related mathematical models for the hurricane data model and the projection results are also potentially changeable.

The system aims to support both professional and general users in a very convenient way. Therefore, a distributed environment and high bandwidth network are needed to handle the simultaneous requests.

Considering all these facts following hardware configurations are employed in the system.

- **Oracle application server runs on a Linux Server:**

IRENE:
Dual CPU P4 Xeon 3.06GHz
2GB RAM
146GB * 6 SCSI Disks
100Mbps connection to network
Runs Linux Fedora Core 2

- **Oracle database runs on a Sun Workstation**

ANDREW:
SunFire V250
Dual CPU UltraSparc III Processors
73GB * 2 SCSI disks
100Mbps connection to network
2GB RAM

Detailed information about the disk partition for the database server is showed in Table 8.1.

DISK	SIZE	CONTROLLER	MOUNTED ON	TABLE SPACE
c0t4d0s6	36GB	controller 0	/home/andrew1	ORACLE01 04G
				ORACLE03 16G
				ORACLE05 16G
c0t5d0s6	36GB	controller 0	/home/andrew2	ORACLE02 2G
				ORACLE04 12G
				ORACLE06 12G
				ORACLE07 10G
c1t1d0s7	36GB	controller 1	/home/andrew	

Table 8.1. Detailed disk partition for the Oracle database server

- **Other machines:**

  CHARLEY: (Backup Sun Server)
  Sun UltraSparc Blade 1000
  Dual CPU UltraSPARC III @ 750MHz
  1GB RAM
  35GB * 2 7200RPM Ultra160 SCSI disks
  35GB * 1 7200RPM FC-AL internal disk
  100Mbps connection to network
  Runs Sun Solaris 2.8 Generic_108528-29

  ISABEL: (Backup Application Server)
  Dual CPU Intel Pentium III 1.2GHz
  1GB RAM
  35GB * 4 10K RPM Ultra160 SCSI disks
  100Mbps connection to network
  Runs Redhat Linux SCS 7.3 kernel 2.4.26

  IBM Cluster: (Property of School of Computer Science)
  IBM RS/6000SP running AIX 5.1/PSSP 3.5 with 35 nodes.
  8 wide nodes with dual 375MHz Power3-II Winterhawk-II processors
  27 thin nodes with single 375MHz Power3-II Winterhawk-II processors

- **Personal Computers (10 machines)**

  2 Machines with following Configuration
  Dell Dimension 4550 / 21 inches Monitor
  Windows XP operating System
  Pentium 4, 3.06GHz Processor, 1GB RAM
  230GB Disk Space
  16X DVD-ROM, 3.5”, 1.44 MB floppy drive

  3 Machines with following Configuration
  Dell 1400 GX 400/Minitower/21 inches Monitor
  Windows 2000 Operating System
  Pentium 4, 1.4 GHz Processor, 1 GB RAM, 256K Cache
  Two 40 GB EIDE 7200 rpm ATA/100 Hard Drive
  16X DVD-ROM
  Harman-Kardon 19.5 Speakers

  10 Machines with following Configuration
  Dimension 4100 Series/ 19 inches Monitor
  Windows 2000 Operating System
  Pentium III 1GHz Processor, 256MB SDRAM
  40GB Ultra ATA 7200 rpm hard drive
  CD-ROM, 3.5” floppy drive

- **Laptops (4 machines)**

  Two Laptops with following Configuration

  Latitude C600
  Windows 2000 Operating System
  Pentium III, 850 MHz, 256MB RAM
  14.1” TFT
  20GB Hard Drive
  8X DVD with software
  Nylon Carrying Case

  Two Laptops with following Configuration

  Dell Latitude D800
  Pentium 256 M Processor 755 (2.0GHz) w/ 15.4 WSXGA+ Display
  1024MB,DDR SDRAM 2 DIMMS
  NVIDIA\256 128MB DDR Video Memory 128MB
  60GB,Hard Drive, 9.5MM, 5400RPM
  DELL LOGITECH USB OPTICAL MOUSE
  Internal 56K Modem
  8-24-24-24X SWDVD/CDRW Combo Drive
  Intel\256 PRO/Wireless 2200 WLAN (802.11b/g, 54Mbps) miniPCI Card
  NYLON DELUXE CASE
  3 Year Limited Warranty plus 3 Year NBD On-Site Service

- **Printers**

  Two Printers With following Configuration

  LaserJet Printer 4100N
  2 Extra Memory for two printers

- **Other Accessories**

  2 DVD RW Drives

8.4 Safety and Backups

Nightly backups of all UNIX data disks and selected Windows data disks (at user request) are performed over the network onto Exabyte Mammoth M2 tapes. Full dumps are taken periodically (it works out to every 2-3 weeks) and incrementals are taken daily between them.

A separate full dump of all department UNIX data disks and selected Windows data disks (at user request) is taken once per semester, including IRENE and ANDREW, and kept at an off-site location, usually a different building on campus.

Section 9

Training Plan

9.1 Introduction

Training of PHRLM must be made available to the development team members (computer group) who will implement the system and to those who will use it to successfully complete their tasks. This document describes the plan that will be used to train the technical staff on PHRLM system. It also describes the plan being used to train the other professional group members (engineering group, meteorology group, statistics group, and finance group).

9.2 Technical Training Plan

The Technical training plan is intended for the development team members who will assist with the system development, installation, configuration, and maintenance of PHRLM. These staff will include the personnel assigned to tasks such as loss model implementation, client-side user interface implementation, web design, report development and maintenance, database development and maintenance, and mainframe system integration. They may also include the personnel with duties relating to the administration and maintenance for PHRLM and its components. The objective of the training is for the student to gain enough knowledge and hands-on experience to get started on his/her tasks relating to system development.

The training will include several seminars and demos. A lecture will be presented in each seminar. Following the lecture, each student will have an overall idea about the system structure and its individual components. Then one-to-one computer based instruction will be entailed to reveal the appropriate technical details that the student needs to know. The following is an overview of the topics that will be presented in each of the training seminars.

- Introduction to PHRLM: system architecture, system configuration, software components, web application interface, etc.
- Database Component: HURDAT data, engineering data, wind-field data, and insurance data, Oracle 9i basics, Oracle DBA basics (for DBA only).
- AHO (Annual Hurricane Occurrence) Module: Statistical models, model implementation, IMSL C++ library, JNI, web interface design using JSP.
- SGT (Storm Genesis Time) Module: Statistical models, model implementation, query design, query optimization, web user interface design, web graphical demo applet design.
- Engineering Module, Wind-field Module, and Insurance Module: model implementation, performance optimization, data characteristics, system integration.
- The policies for using CVS and documenting program/data files.
- The security policies.

Each student will receive a copy of all the seminar presentations and a copy of the most recent technical report maintained by the designated administrator/coordinator. The training will be conducted in ECS building at Florida International University. A training feedback form will be given to each student upon the completion of the training session. The form will be designed to gather feedback on the seminar content and the instructors. The feedback will be used to update and/or improve future training sessions.

9.3 End User Training Plan

The end user training plan is intended for PHRLM end users and managers. The intended audience will consist of the professional group members (engineering group, meteorology group, statistics group, and finance group). They may also expand to Federal customers as needed. The objective of this training plan is to train the end user with PHRLM so they can use it for their tasks immediately.

The training will entail one-to-one instruction between attendees and technical instructors using computer based training. Before the actual training session, a computer usage survey will be passed to each attendee to collect the information such as their computer skills. This is very useful for the instructor to assess the student's needs and to better meet his/her training expectations.

A user's manual is under development and will be used in the training. The following topics will be covered in detail:

- Logging into the system
- Performing simulation and specifying the model parameters
- Displaying the simulation results through web interface
- Viewing the documentations and PHRLM related publications
- Using the on-line "Question and Answer" facility to submit questions/answers and browse other people's questions/answers
- The security policies

Each student will receive a user's manual at the training session. The materials used in technical training program will be tailored and used as the auxiliary guide in the end user training sessions. All the training materials can be accessed via PHRLM's web documentation system. We also implemented an on-line Q&A facility for end users to submit their questions and get answers by using the same interface. In addition, an end user request-response policy is set up for user questions submitted by email. The designated technical staff will be responsible for answering users' PHRLM related question during regular working hours and try to response to end users as soon as possible. Similar to technical training program, there is also a training feedback given to each student upon the completion of the training session.

Section 10

PHRLM Related Publications

10. PHRLM Related Publications

[1]. Shu-Ching Chen, Sneh Gulati, Shahid Hamid, Xin Huang, Lin Luo, Nirva Morisseau-Leroy, Mark D. Powell, Chengjun Zhan, Chengcui Zhang, "A Web-based Distributed System for Hurricane Occurrence Projection," Software: Practice and Experience, Volume 34, Issue 6, pp. 549-571, May 2004.

[2]. Shu-Ching Chen, Shahid Hamid, Sneh Gulati, Na Zhao, Min Chen, Chengcui Zhang, and Paresh Gupta, "A Reliable Web-based System for Hurricane Analysis and Simulation," IEEE International Conference on Systems, Man and Cybernetics 2004, pp. 5215-5220, October 10-13, 2004, Hague, The Netherlands.

[3]. Mei-Ling Shyu, Shu-Ching Chen, Min Chen, Chengcui Zhang, and Chi-Min Shu, "MMM: A Stochastic Mechanism for Image Database Queries," Proceedings of the IEEE Fifth International Symposium on Multimedia Software Engineering (MSE2003), pp. 188-195, December 10-12, 2003, Taichung, Taiwan, ROC.

[4]. Shu-Ching Chen, Sneh Gulati, Shahid Hamid, Xin Huang, Lin Luo, Nirva Morisseau-Leroy, Mark Powell, Chengjun Zhan, and Chengcui Zhang, "A Three-Tier System Architecture Design and Development for Hurricane Occurrence Simulation," Proceedings of the IEEE International Conference on Information Technology: Research and Education (ITRE 2003), pp. 113-117, August 10-13, 2003, Newark, New Jersey, USA.