DEVELOPMENT OF A COMPUTER PROGRAM THAT CALCULATES NET IRRIGATION EFFICIENCIES

BY

OGUNNAIKE PHILIP TOLUWALOPE

MATRIC. No.: 2004/18400EA

DEPARTMENT OF AGRICULTURAL AND BIORESOURCES ENGINEERING,

FEDERAL UNVERSITY OF TECHNOLOGY, MINNA.

FEBRUARY, 2010.

DEVELOPMENT OF A COMPUTER PROGRAM THAT CALCULATES NET IRRIGATION EFFICIENCIES

BY

OGUNNAIKE PHILIP TOLUWALOPE

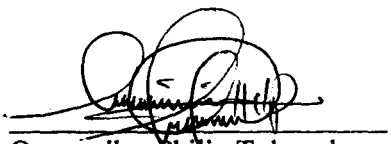
MATRIC. No.: 2004/18400EA

BEING A FINAL YEAR PROJECT REPORT SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE ACHELOR OF ENGINEERING (B. ENG.) DEGREE IN AGRICULTURAL AND BIORESOURCES ENGINEERING, FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA, NIGER STATE.

FEBRUARY, 2010.

DECLARATION

I hereby declare that this project work is a record of a research work that was undertaken and written by me. It has not been presented hence for any degree or diploma or certificate of any university or institution. Information derived from personal communications, published and unpublished work were duly referenced in the text.



Ogunnaike, Philip Toluwalope

16-02-2010.

Date.

CERTIFICATION

This is to certify that "Development of a Computer Program That Calculates Net Irrigation Efficiencies" by Ogunnaike Philip Toluwalope, meets the regulations governing the award of the degree of Bachelor of Engineering (B. Eng) of the Federal University of Technology, Minna, and it is approved for its contribution to scientific knowledge and literary presentation.

Engr. Mrs. H. I. Mustapha
Supervisor

16.02.10
Date

Engr. Dr. A. Balami
Head of Department Agric. & Bio. Engineering

16.02.10
Date

External Examiner

9/2/2010
Date

DEDICATION

With a sense of reference and honor, I humbly dedicate this work to the Lord Jesus whose unfailing grace and mercy has kept me through the huddles of academic pursuit hitherto.

ACKNOWLEDGEMENT

I want to first appreciate the Almighty, Invisible and only wise God whose compassion has kept me through, granting me the wisdom to address the various challenges I faced during the course of this work. To my Supervisor Engr. Mrs. H. I. Mustapha for her wonderful supervision and understanding even during busy times, I say thank you ma.

I do appreciate my lecturers for their expertise in knowledge impartation, selflessness in carrying out their responsibility and the cordiality established with we the students.

To my siblings Samuel, Tope, Emmanuel, Sesan, Seun and kunle thanks for been there. My parent Rev. and Pastor Mrs. E.O. Ogunnaike for their indefatigable concern, care and support to my academic, not considering the cost they've expended thus far, its indeed blessed to be a child of yours. For the 24 main executives of the Fellowship of Christian Students FUT Minna 2008/2009 session I say God bless you for your prayers.

Mention must also be made of the Fellowship of Christian Students, Federal University of Technology, Minna (His Dwelling place); indeed you are a great place . I want to acknowledge the entire 500 level student of Agriculture and Bioresource Engineering, Federal University of Technology Minna 2008/2009 session for their being a team.

To my friends Timothy Ephramu, Kolawole Gbenga, Joab Burak, Emmanuel Adu, Abdulahi kuta and others who since I came to know have been a blessing, Olofu Adams for being a great blessing to me, I know the future hold a great deal, to mark Olofu who was back bone as far as this project is concern, thanks for your technical assistance.

I want to also remember and appreciate the help offered to me towards this work by Bode Osuntunbo; he contributed his expertise into this project by assisting in the kicking off and the major part of the program, I'm indeed very grateful.

ABSTRACT

This project work reports the development of an Irrigation efficiency calculator software. It was written with the aid of the Java programming language and on the Java platform based on the simplicity, dynamism and the portability it provides above other programming languages and also with the analyzing of the various waterways on and within the soil as it interacts with the crops for proper nourishment of the crop. The program makes use of proven formulas and theories that are used to calculate and analyze irrigation efficiencies manually to enable the computer analyze the irrigation data. The program was test run and improved upon by comparing the results gotten with that ought to be gotten with the manual use of the formula and also with the proper use of the flow chart drawn for the program to run. Inputs like the volume of water diverted from source and that to the farm land, the population of the crop on the farm land, the climatic conditions which includes all forms of precipitation, the crop's potential evapotranspiration ET_o, the crop coefficient K_c, and the average depth of the plot are required, and thus enables the program calculate all the types of irrigation efficiency including the net irrigation efficiency and the Irrigation water Requirement of the intended irrigation exercise to ensure accurate irrigation of the farm plot and these are displayed below on the interface of the software.

TABLE OF CONTENTS

CHAPTER ONE

CHAPTER TWO

CHAPTER THREE

CHAPTER FOUR

CHAPTER FIVE

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

CHAPTER 1

1.0 INTRODUCTION

1.1 Background To The Study

Water is a common chemical substance that is essential for the survival of all known forms of life. In typical usage, *water* refers only to its liquid form or state, but the substance also has a solid state, *ice*, and a gaseous state, *water vapor* or *steam*. Water covers 71% of the Earth's surface. On Earth, it is found mostly in oceans and other large water bodies, with 1.6% of water below ground in aquifers and 0.001% in the air as vapor, clouds, and precipitation. Saltwater oceans hold 97% of surface water, glaciers and polar ice caps 2.4%, and other land surface water such as rivers, lakes and ponds 0.6%, the ones useful or potentially useful to humans as water resources, (Wikipedia, 2009).

Irrigation is an artificial application of water to the soil usually for assisting in growing crops. In crop production it is mainly used in dry areas and in periods of rainfall shortfalls, but also to protect plants against frost. Additionally irrigation helps to suppress weed growing in rice fields. In contrast, agriculture that relies only on direct rainfall is referred to as rain-fed farming. Irrigation is often studied together with drainage, which is the natural or artificial removal of surface and sub-surface water from a given area that comes as a result of water wastage during the irrigation process, (Wikipedia, 2009).

Throughout the world, irrigation (water for agriculture, or growing crops) is probably the most important use of water. Large-scale farming would not provide food for the world's large populations without the irrigation of crop fields by water gotten from rivers, lakes, reservoirs, and wells. Without

irrigation, crops could never be grown in the deserts of California, Israel e.t.c. Pouring water on fields is still a common irrigation method today; but other more efficient and mechanized methods are also used.

1.2 Computerization Of Practices

The role of microcomputers has been increasing at a fast rate during the past 10 years or so. Before 1980, computers were used relatively scarcely, and mainly in universities and research institutes in western countries. With the introduction of microcomputers in the early 1980's computer use has not only become much more common in the industrialized world, but has also gained wider acceptance in developing countries, especially over the past five years.

Computer use has also increased in the world of irrigation and drainage and thus also within the International Institute for Land Reclamation and Improvement (ILRI). Because the Institute has a main task in collecting and disseminating knowledge in these two areas, it is only logical that it has a genuine interest in the application of microcomputers. Computers have been used at ILRI for various purposes, such as developing simulation or calculation models, besides administration and word-processing. They have also been used as a training tool in irrigation and drainage courses, particularly in the annual International Course on Land Drainage, organized by the Institute since 1961 (Lenselink and Jurriens, 1993)

1.3 Types And Purposes Of Computer Use

In general terms, various types of computer applications *can* be distinguished, with both hardware and software aspects. Considering the types of computer for scientific use like:

- For performing repetitive calculations quickly and reliably; this function could characterize a computer as a greatly improved calculator;

-Producing graphical images like pictures, diagrams, graphs, etc., some of which may be used for illustrating text in documents, leading to desktop publishing techniques and software;

- As a drawing aid, through digitizing pixel locations, including coloured images; this function would roughly be that of an improved engineering drawing board; the term CAD (computer-aided design) applies and AutoCAD is one of the common software packages for this purpose;

- As a means of instruction, by guiding the user through prepared educational material; terms like CAL (computer-aided learning) or CAI (computer-aided instruction) apply;

- As a means of communication as such (in telecommunications, E-mail) or as connection with large databases (libraries, registers, climatic data, etc.);

- As intelligent knowledge based systems or expert systems, which provide expert knowledge on a subject for consultation;

- As a research tool to venture into such areas as artificial intelligence, chaos theory and the like.

1.3.1 Computer Use In Irrigation

Irrigation entails much more than computers, of course, and many scientists advocate the central position of people rather than technology. This does not mean that the technical and engineering aspects should be neglected or could not benefit from the application of computers. Even social sciences could benefit from an interface with some computer-based tools and techniques.

Considering first in a broad overview, with some points in mind, in which areas computers, and more specifically microcomputers and their software *can* be advantageously applied.

Computers are used in irrigation in the following broad areas and phases of project development:

- In research and education (Verwey, 1985).
- In the planning and design of irrigation systems, either by sophisticated individual farmers, by western extension agencies or by professional designers of consulting engineers.
- During the implementation of major projects, e.g. in levelling work and in supervision and management.
- In the manufacture of irrigation hardware like pumps, sprinklers, drippers, valves, plastic pipes, etc.
- In the operation of field systems, mainly for pressurized irrigation systems like sprinkler and drip irrigation (Karmeli et al., 1985); such computerized operation may include reservoir operation, barrage operation, controlled pumping and fertigation in greenhouses. Systems *can* be fully automated or may be based on simpler technology, with only computerized data processing.
- In monitoring of irrigation systems; this may be automated; a computerized data collection and analysis system may be set up for immediate or later adjustment; reporting could easily be fitted into such a system. (Lenselink and Jurriens, 1993)

1.4 Statement Of The Problem

In this part of the world, many farmers go into farming and irrigation farming without a knowledge of the water requirement of their farms and thus the amount of water with which to

irrigate with, which either brings about a shortage of soil moisture or excess of water (water logging), which is very detrimental to the soil and the crop. This report gives solutions to such problems.

Also, much attention and focus has always being on the usage of the water on the farm and the processing of its data by manual means; thereby bringing about some uncertainty due to human errors and stress on the basis of over working whereas, the computerization of such process (calculations) can be put into place. This report would also seek to give solution to this problem.

1.5 Objective Of The Study

- To develop a program that would predict the net irrigation efficiency
- To compare the predicted value with the calculated efficiencies.
- To quantify irrigation water needed from the source to the field

1.6 Justification Of Study

The study intends to show the essence for the analyzing of irrigation water use for the sake of quantifying it, for effective usage, considering the ideal implication of every drop of water to a crop.

This project program should provide farmers with the best methods for irrigation to ensure effective water use, ease and also place caution on measures to limiting water insufficiency and waste on the farm land by helping to know the actual amount of water needed for irrigation.

1.7 Significance Of The Study

With the present level of development in Science and Technology the use of the computer has spread into virtually all sectors that exist. This is because of the ease it presents as against the tediousness of manually carrying out responsibilities. Besides this, the precision that the use of the computer presents (if well packaged; without fault) is more than a great percentage of humans would present.

So if well enhanced, the capabilities of the computer in the agricultural sector can not be over emphasized even as its expedience is clearly eminent in Nigeria.

1.8 Scope Of The Study

As based on the stated problems and the objective of the report, the neighborhood of the study would cover irrigation techniques, irrigation water efficiencies, programming and measures to make use of it to quantify irrigation water use.

CHAPTER TWO

2.0 LITERATURE REVIEW

2.1 Soil Moisture Regime

This is the relationship between water and soil in relation to the Water required by growing plants, in order to meet its transpirational requirements, and most of this water must come from the soil. Nutrients in soil are dissolved in water in order to make up the solution from which plants absorb essential elements. Also, soil moisture can help to control soil air and soil temperature which are very essential for plant growth. Water is also required for leaching of nutrients from soil profile as well as being an element in the incidence of soil erosion. In addition to its rule in animal and plants, soil water acts as solvent and hastens the weathering of rock and minerals. (Osunde 2000)

2.2 Crop Water Requirement

The first step in a proper design of irrigation is to know the crop water requirements. Datas can be obtained by measuring the off water used by different crops under the field conditions.

The formula for achieving this is;

$$K_c = {}^{ET_c}/_{ET_r} \qquad 2.1$$

Where K_c = Crop coefficient

ETo = Reference evapotranspiration, mm

ET_c = Crop evapotranspiration, mm

The quality or amount of water needed to irrigate a given land area depends on a number of factors; the Nature of crop, the Crop growth cycle, the Climatic condition, land topography, water quality, conveyance efficiency, and field application efficiency

2.3 Irrigation

Irrigation can be defined as the conservation of water and its application to the earth surface for the purpose of supplying the additional moisture needed for proper plant growth (Anthony et al 1986). It is very expedient where there is very little or no precipitation, for the sake of the plants that would need the water to survive.

Some other important reasons for irrigation on crop production are as follows: It provides indemnity to crop as against short duration without water (drought), it helps to reduce the high temperature of the soil; that is cooling the soil sphere with water thereby creating a more favourable environment for plant growth and it soften the soil for better tillage operation to be carried out;

The aim of irrigation is to supply crop with sufficient water so that its evapotranspiration is close to the rate determined by the atmospheric demand for water; this in turn ensures maximum crop growth. The irrigation may be used for many reasons including low total precipitation and provide seasonal or erratic distribution of precipitation. Even when the soil is deep, there are few that can sustain crops from sowing until harvest solely from water stored so that even if permanent irrigation systems are not used, supplementary irrigation is often practiced for a limited period of time.

Furthermore, irrigation schemes or processes can be explain as the;

- Distribution of water in the field after reserving or storing them in a storage tank, and then transporting them to a field or farmland in field scale, where is it needed.

- The Use of input water in the most economical and efficient way in an agricultural system.

This is a function of the highest rate available or distributed between the plants need and the practical portion of water given through the irrigation system by the operator or agricultural engineer (Egharevba, 2002). An Efficient irrigation process would require the knowledge of the crops, the soil and the weather and also; the relationship between them, so that both the amount and timing of irrigation can be interchanged. For example, it is important to know how the available water contents changes as a soil is depleted and the relationship between crop growth and the available water in order to determine when to irrigate. Also it is important to know the depth of the root so that an estimate can be made of the amount of water available in the soil profile. A variety of techniques is used to apply water to crops, but the most common are: gravity or Surface irrigation system (open systems), sprinkle irrigation system and trickle or drip irrigation, where the water table is close to the surface. (Wild, 1988)

2.3.1 Surface Irrigation Systems

Surface irrigation is defined as the group of application techniques where water is applied and distributed over the soil surface by gravity. It is by far the most common form of irrigation throughout the world and has been practiced in many areas virtually unchanged for thousands of years. (Wikipedia 2009.) Surface irrigation is often referred to as flood irrigation, implying that the water distribution is uncontrolled and therefore, inherently inefficient. In reality, some of the irrigation practices grouped under this name involve a significant degree of management (for example surge irrigation). Surface irrigation comes in three major types; level basin, furrow and border strip. It can be described using four phases. As water is applied to the top end of the field it will flow or advance over the field length. The **advance phase** refers to that length of time as water is applied to the top end of the field and flows or

advances over the field length. After the water reaches the end of the field it will either run-off or start to pond. The period of time between the end of the advance phase and the shut-off of the inflow is termed the wetting, ponding or **storage phase**. As the inflow ceases the water will continue to runoff and infiltrate until the entire field is drained. The **depletion phase** is that short period of time after cut-off when the length of the field is still submerged. The **recession phase** describes the time period while the water front is retreating towards the downstream end of the field. The depth of water applied to any point in the field is a function of the **opportunity time**; the length of time for which water is present on the soil surface (Wikipedia, 2009)

2.3.1.1 Basin Irrigation

Level basin irrigation has historically been used in small areas having level surfaces that are surrounded by earth banks. The water is applied rapidly to the entire basin and is allowed to infiltrate. Basins may be linked sequentially so that drainage from one basin is diverted into the next once the desired soil water deficit is satisfied. A "closed" type basin is one where no water is drained from the basin. Basin irrigation is favoured in soils with relatively low infiltration rates (Walker and Skogerboe 1987). Fields are typically set up to follow the natural contours of the land but the introduction of laser levelling and land grading has permitted the construction of large rectangular basins that are more appropriate for mechanised broad acre cropping.

2.3.1.2 Bay/Border Strip Irrigation

Border strip or bay irrigation could be considered as a hybrid of level basin and furrow irrigation. The borders of the irrigated strip are longer and the strips are narrower than for basin irrigation and are orientated to align lengthwise with the slope of the field. The water is applied to the top end of the bay, which is usually constructed to facilitate free-flowing conditions at the downstream end. One common use of this technique includes the irrigation

of pasture for dairy production. A relationship between the discharge through the supply ditch Q, the average depth of water flowing over the strip y, the rate of infiltration through the soil f, the area of land irrigated A, the approximate time required to cover the given area with water t, is given by the empirical equation:

$$t = 2.3\,{}^{y}/_{f}\, log_{10}\left({}^{Q}/_{Q - Af}\right) \qquad 2.2$$

(Michael, 1998)

2.3.1.3 Furrow Irrigation

Furrow irrigation is conducted by creating small parallel channels along the field length in the direction of predominant slope. Water is applied to the top end of each furrow and flows down the field under the influence of gravity. Water may be supplied using gated pipe, siphon and head ditch or bank less systems. The speed of water movement is determined by many factors such as slope, surface roughness and furrow shape but most importantly by the inflow rate and soil infiltration rate. The spacing between adjacent furrows is governed by the crop species, common spacing typically range from 0.75 to 2 metres. The crop is planted on the ridge between furrows which may contain a single row of plants or several rows in the case of a bed type system. Furrows may range anywhere from less than 100 m to 2000 m long depending on the soil type, location and crop type. Shorter furrows are commonly associated with higher uniformity of application but result in increasing potential for runoff losses. Furrow irrigation is particularly suited to broad-acre row crops such as cotton, maize and sugar cane. It is also practiced in various horticultural industries such as citrus, stone-fruit and tomatoes. The water can take a considerable period of time to reach the other end, meaning water has been infiltrating for a longer period of time at the top end of the field. This results in poor uniformity with high application at the top end with lower application at the bottom end. In most cases the performance of furrow irrigation can be improved through increasing the speed at which water moves along the field (the advance rate). This can be achieved through

increasing flow rates or through the practice of surge irrigation. Increasing the advance rate not only improves the uniformity but also reduces the total volume of water required to complete the irrigation. Surge Irrigation is a variant of furrow irrigation where the water supply is pulsed on and off in planned time periods (e.g. on for ½ hour off for ½ hour). The wetting and drying cycles reduce infiltration rates resulting in faster advance rates and higher uniformities than continuous flow. The reduction in infiltration is a result of surface consolidation, filling of cracks and micro pores and the disintegration of soil particles during rapid wetting and consequent surface sealing during each drying phase.(Kemper, et al 1988) The effectiveness of surge irrigation is soil type dependent, for example many clay soils experience a rapid sealing behaviour under continuous flow therefore surge offers little benefit. (Walker and Skogerboe 1987)

2.4 Irrigation Water Requirement (IR)

This is the amount of water that must be supplied to the crop plant to ensure that it receives its full water requirement or a predetermined position of it. (Larry 1988)

The important use of water in the most economical and efficient way is a function of highest rate available between the plants need the practical portion of water that is transferred through irrigation system by the operation. The net amount of water to be applied during an irrigation is given as follows in (Michael, 1998)

$$d = \sum_{i=1} \frac{M_{ei} - M_{bi}}{100} A_i D_i \qquad 2.3$$

Where **d**= net amount of water to be applied during an irrigation. (mm)

M_{ei}= field capacity moisture content in the i^{th} layer of the soil, per cent

M_{bi}= moisture content before irrigation in the i^{th} layer of the soil, per cent.

A_i= bulk density of the soil in the i^{th} layer

D_i= depth of the i^{th} soil layer, (mm), within the root zone, and

n = number of soil layer in the root zone D

The Gross irrigation requirement is given as below in (Michael, 1998)

$$IR = \sum_{i=1}^{n} {}^{d}/_{E_a} \qquad 2.4$$

Where **IR**= Gross irrigation requirement.(mm)

d = Net amount of water to be applied at each irrigation. (mm)

E_a = Irrigation Efficiency.

2.5 Design Daily Irrigation Requirement (DDIR)

DDIR is determined using the equation below:

$$\mathbf{DDIR = AD/II\ min} \qquad 2.5$$

Where **AD** = Allowed depletion of soil water between irrigations (mm, or m)

II mm = minimum irrigation interval during the irrigation season (days)

Generally, **DDIR**'s for crops grown in soil with low soil holding capacities such as sand, are higher than those for crops grown in finer textured soils with higher water holding capacities. This is because the interval between the irrigations increase with water holding capacity and the average daily irrigation requirement is the smallest for longer irrigation intervals.

The DDIR value for a farm is determined from several years of daily irrigation requirement DIR data. (Larry, 1988)

2.6 Water Irrigation Efficiencies

Since irrigation is assumed to be the largest appropriated water user in developed environments, irrigation systems also receive merit based on how efficient they are reported to be. While this might sound straightforward and simple, there is room for confusion because there are different ways to define efficiency. Efficiencies also vary in time and with management. Very "efficient" systems by some definitions can be very poor performers by other definitions, for example, if distribution uniformity and delivery amount are inadequate to fulfil crop need.

Irrigation efficiency indicates how efficiently the available water supply is being used, based on different method of evaluation. The design of the irrigation system, the degree of land preparation, and the skill and care of the irrigator are the principal factors influencing irrigation efficiency. The loss of irrigation water occurs in the conveyance and distribution system, non-uniform distribution of water over the field, percolation below crop root zone, and with and with sprinkler irrigation evaporation from the spray and retention of water on the foliage. In the case of large fields loss may occur by runoff at the end of irrigation border and furrows. The losses can be held to a minimum by adequate planning of irrigation system, proper design of the irrigation method, adequate land preparation and efficient operation of the system. (Michael, 1998)

There are five overall Best Management Practices that have been established for irrigation systems:

1. Assure the overall quality of the irrigation system.
2. Design the irrigation system for the efficient and uniform distribution of water.
3. Install the irrigation system to meet the design criteria.

4. Maintain the irrigation system for optimum performance.

5. Manage the irrigation system to respond to the changing need for water.

(Irrigation Association Water Management Committee. 2002)

In keeping with these overall Best Management Practices, the following guidelines are relevant:

1. Do not over water most established vegetation, it does not require more than one inch per week depending on the season and rainfall. Plants will develop deeper roots, and ultimately require less watering, when not over-watered.

2. Never water if the soil is still wet.

3. Irrigate according to the requirements of the plants, not on a fixed schedule. The duration of irrigation is typically what needs to be modified based on evapotranspiration (ET).

4. Apply only enough irrigation to replace water loss by ET. Match irrigation application to soil type and root depth. Avoid applying more water than can be contained in the root zone. Daily observation is optimal to determine the appropriate changes to make to the irrigation system. If impractical, weekly observation should be conducted at a minimum.

5. Water all plants deeply but infrequently to encourage deeper, healthier rooting. Prolonged intervals between watering (short of drought damage) provide maximum encouragement of plant growth.

6. Until plants have developed deep roots, they may need more frequent watering than older established plants.

7. When determining the watering needs of planted areas, dig down about 4 to 6 inches to determine the moisture content of the soil. Do not worry about the dryness of the top inch of soil. If the soil is too dry to form a ball when squeezed in the hand, it needs water.

8. Water early in the morning or between the hours of 6 P.M. and 10 A.M. when temperatures and winds are at their lowest levels to reduce water loss. Sprinklers are also typically more efficient during these times due to better water pressure.

9. Excessive irrigation after fertilization may cause leaching or surface runoff that pollutes water bodies, while lack of irrigation may result in inefficient utilization of the fertilizer.

10. Water lawns and shrubs/perennial beds separately. (These should be in different irrigation zones.)

11. Water trees and shrubs, which have deeper root systems, longer and less frequently than shallow-rooted plants.

12. When watering plants on slopes, compacted soils, and/or sandy soils, a series of several light applications instead of one continuous application is typically appropriate to account for the lower intake rates of these soils. Consider installing low-angle nozzles on tops of slopes to improve efficiency. Irrigation systems should also apply more water at the top of the slope and less at the base to prevent excess runoff.

13. Watering too frequently may promote some diseases in the landscape.

14. Irrigation efficiency is equally dependent upon a good design, correct installation and proper maintenance. Use only qualified (e.g., licensed, certified as needed) irrigation professionals for all phases of irrigation management.

(Irrigation Association Water Management Committee, May 2004)

2.6.1. Water Conveyance Efficiency (E_c):

This is the percentage of source water that reaches the field, defined by the formulae below in Michael (1998)

$$E_{c=}\ 100(W_f\ /W_s) \qquad 2.6$$

Where, W_f = Water delivered to field (m^3)

W_s = Water diverted from source (m^3)

This term is used to measure the efficiency of water conveyance systems associated with the canal network, water courses and field channels. It is also applicable where the water is conveyed in channels from the well to the individual fields.

Conveyance efficiency is the first of efficiencies a farmer would concern himself about, it deals with the first step in the line of the irrigation process; generally a concern for irrigation districts that supply a group of farmers through a system of canals and open ditches. Conveyance efficiency should be nearly 100 percent, based on the enclosed means of conveying, there is no way for seepage.

2.6.2 Water Application Efficiency (E_a)

Having conveyed the available water to the farm through conveyance structures, the need is apparent to apply the water efficiently.

The following concepts of water application efficiency was developed to measure and focus attention upon the efficiency with which water delivered is being stored within the root zone of the soil where it could be used by plants. The water application efficiency was given in Hansen and stringham, (1980) as;

$$E_a = 100\, W_s / W_f \qquad 2.7$$

Where, E_a = water – Application efficiency.

W_s = water stored in the soil root zone during the irrigation.(m^3)

W_f = water delivered to the farm.(m^3)

Water application efficiency gives a general sense of how well an irrigation system performs its primary task of getting water to the plant roots. However, it is possible to have a high Ea but have the irrigation water so poorly distributed that crop stress exists in areas of the field.

Water application efficiency sometimes is incorrectly used to refer to the amount of water delivered to the surface of the soil in an irrigated field by a sprinkler system. Water losses can occur after reaching the soil surface, leading to overestimation of the application efficiency. Ea is often confused with water storage efficiency (Es), which is the fraction of an irrigation amount stored in the crop root zone.

A water loss in this aspect includes surface runoff and deep percolation. If a centre pivot is equipped with a properly designed nozzle package and operated using best management practices and irrigation scheduling, these losses can be negligible. However, for many systems, these losses can be large and result in poorly distributed or non-uniform irrigation (Rogers, et al, 1997).

Water application efficiencies below 100 per cent are due to seepage loses from the field distribution channels and deep percolation below the crop root zone. Sometimes, in case of very long fields, there may be runoff losses at the tail end of the furrows and borders. In general water application efficiency decreases as the amount of water applied during each irrigation increases. However very small irrigations may not fill the root zone adequately and may reduce crop yields, and in a long run give rise salt problems due to inadequate leaching. (Michael, 1998)

2.6.3 Water Storage Efficiency

It has been stated that small irrigations may lead to high water application efficiencies, yet the irrigation practice may be poor. The concept of water storage efficiency is useful in evaluating

this problem. This concept relates how completely the water needed prior to irrigation has been stored in the root zone during irrigation. It is defined in Michael, (1998) as:

$$E_s = \frac{W_s}{W_n} \times 100 \qquad 2.8$$

Where, E_s= water storage efficiency, per cent.

W_s= Water stored in the root zone during irrigation.(m^3)

W_n = Water needed in the root zone prior to irrigation. (m^3)

Water storage efficiency becomes important when water supplies are limited or when excessive time is required to secure adequate penetration of water into the soil. Also, when salt problems exist, the water storage efficiency should be kept high to maintain a favourable salt balance.

2.6.4 Water Distribution Efficiency (E_d):

The percentage of the average application depth delivered to the east-watered part of the field. It is expressed in Michael (1998) as:

$$E_d = 100[1-(\bar{y}/d)] \qquad 2.9$$

$\bar{y}$ = Average absolute numerical deviation in depth of water stored from average depth stored during the irrigation **d. (mm)**

d = Average depth of water stored during irrigation.(cm)

The water distribution efficiency indicates the degree of uniformity in the amount of the water infiltrated into the soil. It also could be defined as the uniformity in depths applied at the surface based on catch-can measures for sprinkler systems. This concept for uniformity was originally developed by Christiansen in 1942 for sprinkler systems. (Rogers, et al, 1997)

It is not only the right amount of water to the field that is important but also the uniform distribution over the field. Permissible lengths of irrigation runs are controlled to a large extent by the uniformity of water distribution which is possible for a given soil and irrigation management practices. Water distribution efficiency indicates the extent to which water is uniformly distributed along the run.

2.6.5 Water-Use Efficiency (E_u)

Having transported the water to the point of use and having applied the water, the next efficiency concept for concern is the efficiency of the water-use by the crop. What proportion of the water delivered was beneficially used on the farm or field can be calculated using the following formula.

$$E_u = 100\, W_u / W_d \qquad 2.10$$

Where, E_u= water use efficiency

W_u = water beneficially used. (m^3)

W_d= water delivered to the field. (m^3)

Water use efficiency is more broadly defined than water application efficiency in that irrigation water may have more uses than simply satisfying crop water requirements. So therefore we have:

a) **Crop water- use Efficiency:** the ratio of crop yield (y) to the amount of water depleted by the crop in the process of evapotranspiration (ET)

   Crop water-use efficiency = $\frac{Y}{ET}$ 2.11

b) **Field water- use Efficiency:** the ratio of crop yield (y) to the total amount of water used in the field (WR)

Field water- use efficiency $= \frac{Y}{WR}$. 2.12

(Michael, 1998)

Other beneficial uses could include salt leaching, crop cooling, pesticide or fertilizer applications, or frost protection. However, most irrigation systems are single-purposed, that is to supply water for crop use, which allows water application efficiency and irrigation efficiency to be used interchangeably.

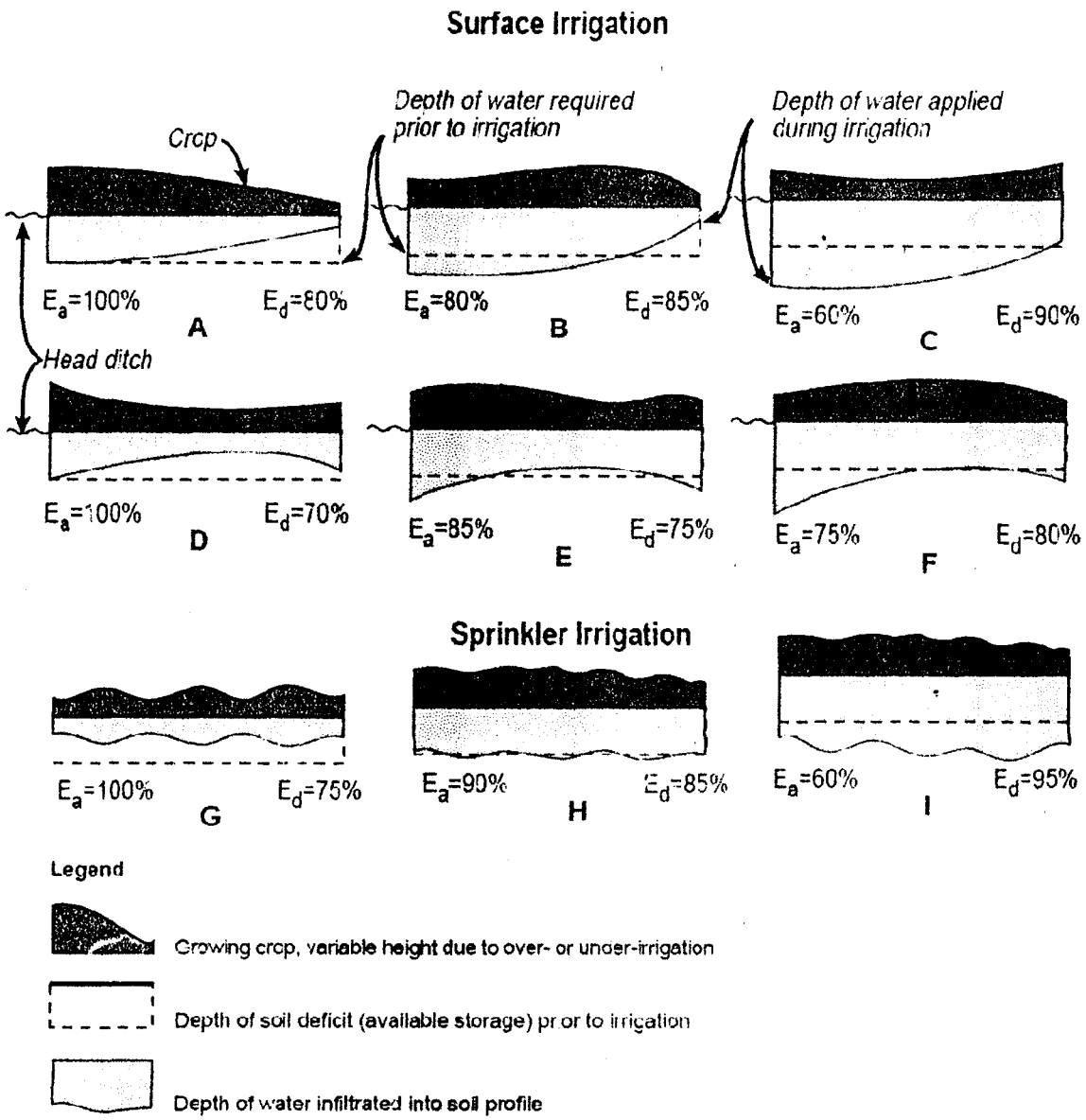


Figure 2.1. Application, Ea, and distribution, Ed, efficiencies and the effect on crop production illustrated by two-dimensional soil profiles. For these examples, Ea, estimates are made assuming no runoff. (Hansen, 1960)

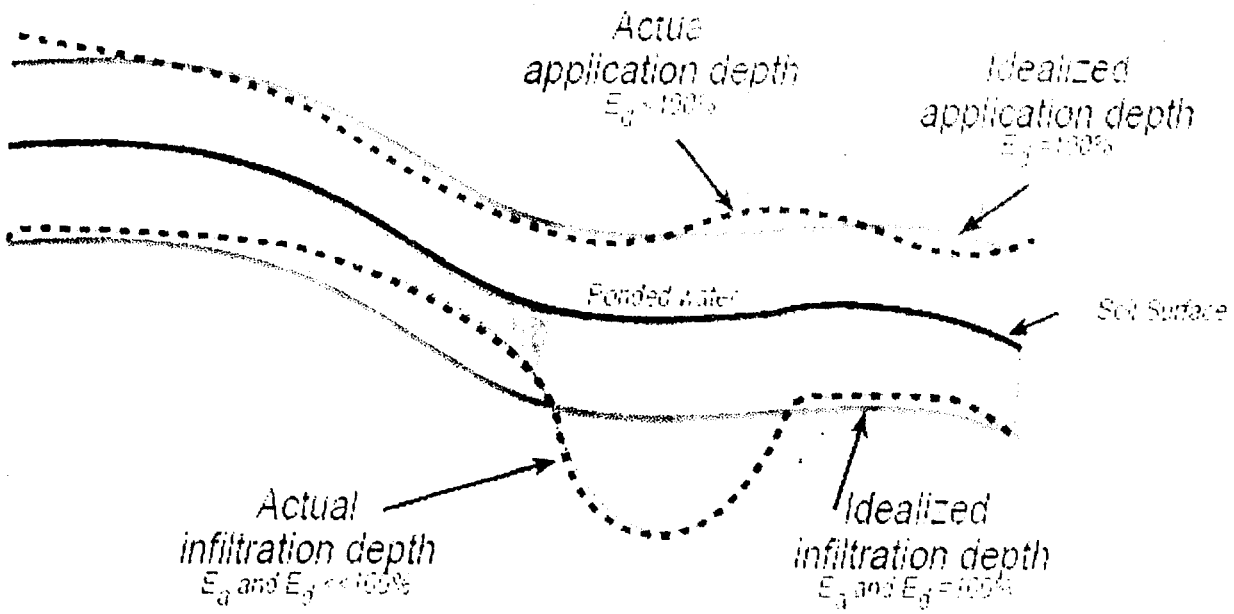


Figure 2.2. Illustration of sprinkler package water distribution uniformity versus infiltrated water distribution uniformity in soil. (Hansen, 1960)

Table 2.1. Range of Application Efficiencies for Various Irrigation Systems

System Type	Application Efficiency Range* (%)
Surface Irrigation	
Basin	60 - 95
Border	60 - 90
Furrow	50 - 90
Surge	60 - 90
Sprinkler Irrigation	
Handmove	65 - 80
Travelling Gun	60 - 70
Centre Pivot & Linear	70 - 95
Solid Set	70 - 85
Micro irrigation	
Point source emitters	75 - 95
Line source emitter	70 - 95

* *Efficiencies can be much lower due to poor design or management. These values are intended for general system type comparisons and should not be used for specific systems.*

2.6.6 Irrigation Efficiency Examples

Irrigation efficiency examples are shown in Figure 1 for surface and sprinkler irrigation. Examples (A), (B) and (C) show a series with increasing application depth for a field with the heaviest application occurring at the top of the field. The dashed line in the profile represents the depth of water needed to meet crop requirements until the next irrigation event. When the shaded application depth does not reach this line, that portion of the field would be under water stress. Example (A) illustrates how a portion of a crop can be under stress with 100 percent application efficiency, while example (C) shows a crop with no stress but a low application efficiency. Notice crop vigour is represented as less than optimum for areas with heavy deep percolation. Excess water can leach needed nutrients or cause waterlogged growing conditions.

The application efficiencies in Figure 1 are made using a no-runoff assumption, although for the surface irrigation example (A), (B), and (C), this might be better represented as complete tailwater capture and reuse. Example (D), (E) and (F) could be thought of as blocked-end or diked surface irrigated fields. Blocking the end of the field generally results in the driest portion of the field being about 2/3 to 4/5 of the length of run with wettest conditions and the potential for deep percolation losses split between the upper and lower portion ends of the field. Sprinkler irrigation illustrations are shown in examples (G), (H), and (I). Example (H) is the desirable situation while (G) illustrates crop stress due to under-irrigation and (I) shows over irrigation. Centre pivot sprinkler packages, even if properly designed, do not have perfect distribution uniformity. Each nozzle outlet progressively has to cover a larger land area (concentric circles) with increasing distance from the centre pivot point. Each outlet has a unique and specific discharge rate requirement. However, nozzle outlets are not manufactured in an infinite number of sizes. For a specific nozzle outlet, the designer will select the nozzle

outlet size that most closely matches the design specification. Sprinkler spacing must also be consistent with the manufacturer's recommendations to avoid distribution problems. Good designs should have distribution uniformities of approximately 90 percent. In Figure 2, the average design application depth is represented by the solid green line above the soil surface. The dotted black line that moves above and below the design depth represents what actual measured results might look like.

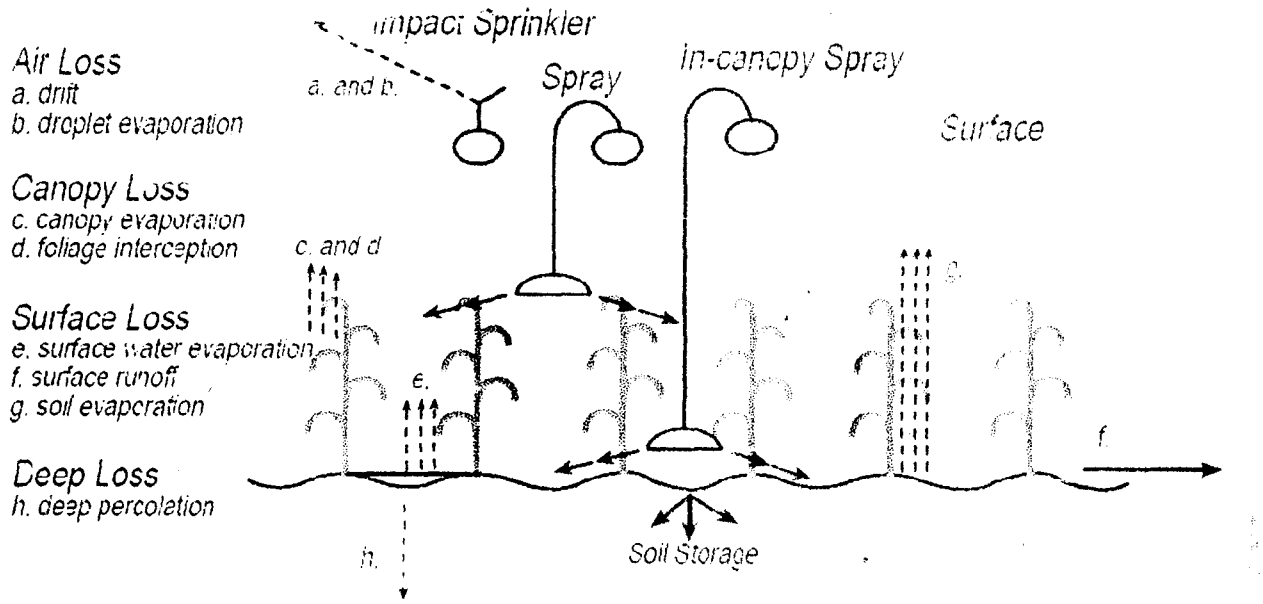


Figure 2.3. Irrigation water loss and storage locations. (Rogers, et al, May 1997)

Table 2.2. Estimated Sprinkler Water Loss Components for a 1-inch Irrigation. Ground evaporation, runoff, and deep percolation were negligible (Schneider and Howell, 1993)

System	Air Loss	Canopy Loss	Surface Loss	Total Loss	Surface Application Efficiency*
Impact Sprinkler	0.03	0.12	-	0.15	85%
Spray Head at Truss	0.01	0.07	-	0.08	92%
LEPA	-	-	0.02	0.02	98%

**Runoff within field, distribution, or deep percolation loss are not considered.*

Table 2.3. Surface Irrigation Loss Estimates*

Loss	Estimate Method	% of 4 Acre-inch Irrigatn Applied
Furrow Water Evaporatn	(0.01 in/hr × 8 hr) = 0.08 inches**	2.00
Runoff Water Evaporatn	(0.3 in/day × 0.6 ac)*** = 0.18 ac-in/day	0.28

Tailwater Pit Evaporation	(0.3 in/day × 2 acres) = 0.60 ac-in/day	0.94	
Tailwater Pit Leakage	(0.25 in/day × 2 acres) = 0.5 ac-in/day	0.78	
Total			4.00

** Distribution or deep percolation loss not considered*

*** Same rate as LEPA. 8 hours represents watered furrow conditions during advance and recession. Every other row irrigation. *** 20 ft. strip, 1320 ft. long.*

2.7 Irrigation Water Losses

Irrigation water losses, illustrated in Figure 3, include air losses, canopy losses, soil and water surface evaporation, runoff, and deep percolation. The magnitude of each loss is dependent on the design and operation of each type of irrigation system. Table 2 shows an estimate of the application efficiency of three sprinkler packages, assuming ground evaporation, runoff and deep percolation are negligible. Ground evaporation may be an important component early in the season, before the crop canopy covers the surface.

2.7.1 Sprinkler Irrigation Losses

Air losses include drift and droplet evaporation. Air losses can be very large if the sprinkler design or excessive pressure produces a high percentage of very fine droplets. Drift is normally considered to be water particles that are removed from the target area, while droplet evaporation would be the loss of water by evaporation directly from the drop of water while in flight. Direct movement and droplet evaporation vary, but the general estimate of droplet evaporation is small, probably less than 1 percent of the output. Total air loss under properly-operating sprinklers and low wind conditions is likely to be in the 1 to 3 percent range, although some older publications have much higher values. Table 2 assumes 3 percent for the impact sprinkler and 1 percent for the spray head at a 5 foot height. Air losses were assumed

to be negligible for the bubble mode LEPA head. Canopy losses include losses due to water held on the plant (foliage interception) and canopy evaporation during the irrigation. Water evaporation from the wetted surface of the plant does reduce transpiration by the plant. However, evaporation from a free water surface is faster than transpiration through plant stomates. Net canopy evaporation loss estimates range from 0.02 to 0.04 inch per hour. Two hours of wetting was assumed for the impact sprinkler and 45 minutes for the spray nozzle. Plant interception loss estimates range from 0.04 to 0.08 inches. The 0.04 inches loss estimate was used in Table 2.2. (Hansen, 1960)

The only loss shown for the bubble mode LEPA nozzle is surface water evaporation. Since the LEPA system uses an application rate in excess of soil intake capabilities, the free water surface must be held on the soil surface until it can be infiltrated. The surface water evaporation loss estimate is 0.01 inch/hour over the two hours estimated for intake to be complete. In all examples of Table 2, water movement as runoff or redistribution of the surface water, deep percolation, and ground evaporation were considered to be negligible. Any runoff from the field or deep percolation would reduce application efficiency by a percentage of the total application amount. Runoff of up to 60 percent of the application amount has been measured for in-canopy sprinkler heads on sloping ground.

2.7.2 Surface Irrigation Losses

Surface irrigation losses include runoff, deep percolation, ground evaporation and surface water evaporation. Runoff losses can be significant if tailwater is not controlled and reused. Although use of tailwater reuse pits could generally increase surface application efficiency, many surface irrigators use a blocked furrow to prevent runoff. Usually the lower portion of the field is levelled to redistribute the tailwater over that portion. While runoff may be reduced to near zero, deep percolation losses may still be high with this practice. Surge

irrigation can accomplish faster furrow advances. To further improve an advance time, large furrow flows may be used. However, care should be taken to avoid furrow erosion. Some chemicals (polymers) have been reported to be useful in reducing erosion. Rapid advance allows better water distribution efficiency and smaller application amounts, which can reduce deep percolation losses and improve overall irrigation efficiency. Evaporation loss percentages from a surface irrigated field are small. The components of the loss are furrow water evaporation (under canopy), tailwater evaporation (where there is no canopy protection) and tailwater pit evaporation, and are dependent on system operation. Loss estimates are shown in Table 3 assuming a 4-inch gross application depth is applied to a 160-acre surface irrigated field using 12-hour sets on a 10-day irrigation interval. Some loss components were estimated on a daily basis, so the percent loss was dictated by the daily application amount (64 acre-inches). Tailwater pit leakage is also a potential loss and is shown in Table 2.3. (Hansen, 1960)

2.8 Programming Language

A programming language is a machine-readable artificial language designed to express computations that can be performed by a machine, particularly a computer. Programming languages can be used to create programs that specify the behaviour of a machine, to express algorithms precisely, or as a mode of human communication. (Wikipedia 2009)

A few examples of programming languages are as follows: A+, A++, ABSYS, ALGOL, Baja, BASIC. C, C--, C-script, C++, FORTRAN, HTML, Java, JavaScript, MATLAB, Maya, Perl, Smalltalk, Scheme, e.t.c. they are used to address various problems as according to the capability of the language

Many programming languages have some form of written specification of their syntax and semantics, since computers require precisely defined instructions. Some (such as C) are

defined by a specification document (for example, an ISO Standard), while others (such as Perl) have a dominant implementation.

The earliest programming languages predate the invention of the computer, and were used to direct the behaviour of machines such as automated looms and player pianos. Thousands of different programming languages have been created, mainly in the computer field, (Murdoch University, 2006) with many more being created every year.

A programming language is a language used to write computer programs, which involve a computer performing some kind of computation or algorithm and possibly control external devices such as printers, robots, and so on. It differs from natural languages in that natural languages are only used for interaction between people, while programming languages also allow humans to communicate instructions to machines. Some programming languages are used by one device to control another. For example PostScript programs are frequently created by another program to control a computer display. Programming languages may contain constructs for defining and manipulating data structures or controlling the flow of execution (Wikipedia 2009).

2.8.1 Program Usage

A programming language provides a structured mechanism for defining pieces of data, and the operations or transformations that may be carried out automatically on that data. A programmer uses the abstractions present in the language to represent the concepts involved in a computation. These concepts are represented as a collection of the simplest elements available (called primitives), (Wikipedia 2009).

Programming languages differ from most other forms of human expression in that they require a greater degree of precision and completeness. When using a natural language to

semantics (either formal or hard-coded in a reference implementation). Since most languages are textual, this article discusses textual syntax. Programming language syntax is usually defined using a combination of regular expressions (for lexical structure) and Backus-Naur Form (for grammatical structure). Not all syntactically correct programs are semantically correct. Many syntactically correct programs are nonetheless ill-formed, per the language's rules; and may (depending on the language specification and the soundness of the implementation) result in an error on translation or execution. In some cases, such programs may exhibit undefined behaviour. Even when a program is well-defined within a language, it may still have a meaning that is not intended by the person who wrote it. (Michael Sipser 1997)

2.8.2.2 Core Library

Most programming languages have an associated core library (sometimes known as the 'Standard library', especially if it is included as part of the published language standard), which is conventionally made available by all implementations of the language. Core libraries typically include definitions for commonly used algorithms, data structures, and mechanisms for input and output.

A language's core library is often treated as part of the language by its users, although the designers may have treated it as a separate entity. Many language specifications define a core that must be made available in all implementations, and in the case of standardized languages this core library may be required. The line between a language and its core library therefore differs from language to language. Indeed, some languages are designed so that the meanings of certain syntactic constructs cannot even be described without referring to the core library. For example, in Java, a string literal is defined as an instance of the java.lang.String class; similarly, in Smalltalk programming language, an anonymous function expression (a "block")

constructs an instance of the library's BlockContext class. Conversely, the language Scheme contains multiple coherent subsets that suffice to construct the rest of the language as library macros, and so the language designers do not even bother to say which portions of the language must be implemented as language constructs, and which must be implemented as parts of a library. (Wikipedia 2009)

So it is well expedient that the various programming languages be understood; the Library, the right semantics and other peculiarities, so as to achieve a run able program.

CHAPTER THREE

3.0 MATERIAL AND METHOD

3.1 Materials

The material used for the development of this program is java. Java Technology is a high level programming language and platform, it is a preferred programming language based on its simplicity, object orientation, dynamism, portability, security, high performance and the many other advantages it has over other programming language.

In the java programming language, all source code is written first in plain text which ends with java extensions. These (source codes) are then compiled into a **class** file by the java compiler. Class file codes are not native to the computer processor, instead it contains byte codes- the machine language of the java virtual machine. The java launcher tool then runs your application with an instance of the java virtual machine.

Java Virtual Machine is available on many different operating systems, the same class files are able to run on operating systems such as; Solaris, Linux, Mac and even the renowned Microsoft Windows. Virtual machines such as the Java hot spot virtual machine perform additional steps at runtime to give your application a performance boost. This includes various tasks such as finding performance bottlenecks and recompiling, to the native code, frequently used sections of your code.

The platform is the hardware or software (the environment) where the program runs, for example as already stated the Microsoft Windows, Linux, Solaris operating system and the Macintosh the operating system of Apple. They are mostly the combination of the operating system and the underlying hardware. Java applications do not interact directly with a computer's central processing unit (CPU) or operating system and are therefore platform

independent, meaning that they can run on any type of personal computer, workstation, or mainframe computer; this is referred to as "write once, run everywhere," which can write applications that will run on otherwise incompatible operating systems such as stated above.

The java platform has two components:

- The Java Virtual Machine
- The Java Application Programming Interface (API)

The Virtual Machine is the base for the Java platform, a special program within the browser software that interprets the bytecode—the code that the applet is written in—for the computer's CPU and is ported onto various hardware- based platforms. The virtual machine is able to translate the platform-independent bytecode into the platform-dependent machine code that a specific computer's CPU understands. The API is a large collection of ready made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. It is grouped into libraries of related classes and interfaces; these libraries are known as packages.

3.2 Method

The method used in developing this software comprises of the java programming codes, using a java compiler on a java development platform which makes functions on any operating system. The software was built upon already existing formulas which bases on the calculations of the rate at which crop evapotranspiration is undergone in relation with the crop water requirement dependent on the crop- coefficient. All these put together with other possible means (the analyzing of the water ways in the soil, outside the soil and even in the plant) so as to be able to achieve the irrigation water efficiency of each stage in the farm.

3.2.1 Mathematical Model Background

The amount of water used by a specific crop can be related to the amount of water used by a reference crop. The reference crop typically is grass that is well irrigated and covers 100% of the ground. The reference crop's water consumption is called "reference evapotranspiration," and abbreviated ETo.

ETo includes the water evaporated from the soil surface and the water used (transpired) by the plant.

To relate the ETo to a specific crop water use, you must multiply the reference ETo y the adjustment coefficients.

The first adjustment is called the "crop coefficient" Kc. The Kc adjust the ETo to account for the difference between the inherent water – using characteristics of the specific crop compared to the reference crop. This adjustment may be upward or downward, because the crop may use more or less water than the reference crop. The second adjustment is called "Kground Cover" abbreviated Kg. Kg accounts for the difference between the portions of ground covered by the crop compared to the reference crop, which covers 100% of the ground. Since the amount of ground covered by the crop will increase as the crop grows, this number will change quite a bit in the early part of the season.

So therefore we get an appropriate formula:

$$ET_c = (K_c)(K_g)(ET_o) \qquad 3.1$$

DU, distribution uniformity; the ratio of the lowest 25% of measurements, known as the low-quarter average and the average of all the measurement, known as the global average, estimates the relationship of the driest quarter of your field to the entire field. In other words, on the basis of the whole field average, if your irrigation was exactly not only the water used

by the crop, but also the water used in evaporation and drainage, the driest quarter of your field would be 25% short of water.

Consequently, the DU is used to guide in correcting the ETc to giving the irrigation water requirement (IR) which is almost equal to the Evapotranspiration. This is achieved by the dividing of the calculated ETc by the Calculated DU.

$$IR = \frac{ETc}{DU} \qquad 3.2$$

Water requirement is a major factor that must be gotten, it can also be computed by adding measured quantities of irrigation water, the effective rainfall received during the season and the contribution of moisture from the soil. This may be expressed by the following relationship:

$$WR = IR + ER + \sum_{i=1}^{n} \frac{M_{bi} - M_{ei}}{100} . A_i . D_i \qquad 3.3$$

In which,

WR = Seasonal water requirement, mm

IR = Total irrigation water applied, mm

ER = Seasonal effective rainfall, mm

M_{bi} =Moisture percentage at the beginning of the season in the i^{th} layer of the soil

M_{ei} = Moisture percentage at the end of the season in the i^{th} layer of the soil.

A_i = apparent specific gravity of the i^{th} layer of the soil

D_i = Depth of the i^{th} layer of the soil within the root zone, mm

n = Number of soil layers in the root zone D

Hence a simplified formula to help calculate the requirement at any needed point in time:

$$W_c = P_n + I + D_i - R_o - D_r - S \qquad 3.4$$

Where:

W_c = Water requirements

P_n = Precipitation applied

I = Irrigation water applied, mm

D_i = Stored soil water depth, mm

R_o = Water runoff, mm

D_r = Drainage / seepage, mm

S = Special losses, mm

3.2.2 Basic Theoretical Concept

The program was developed based on some factors that were put into consideration to guide for computing water irrigation efficiencies. They are:

- The seasons effect on irrigation.
- The soil water requirement
- The seepage capabilities of the soil
- Transpiration of the crop
- Ground slope (affects the water seepage and the distribution efficiency)
- Soil compaction

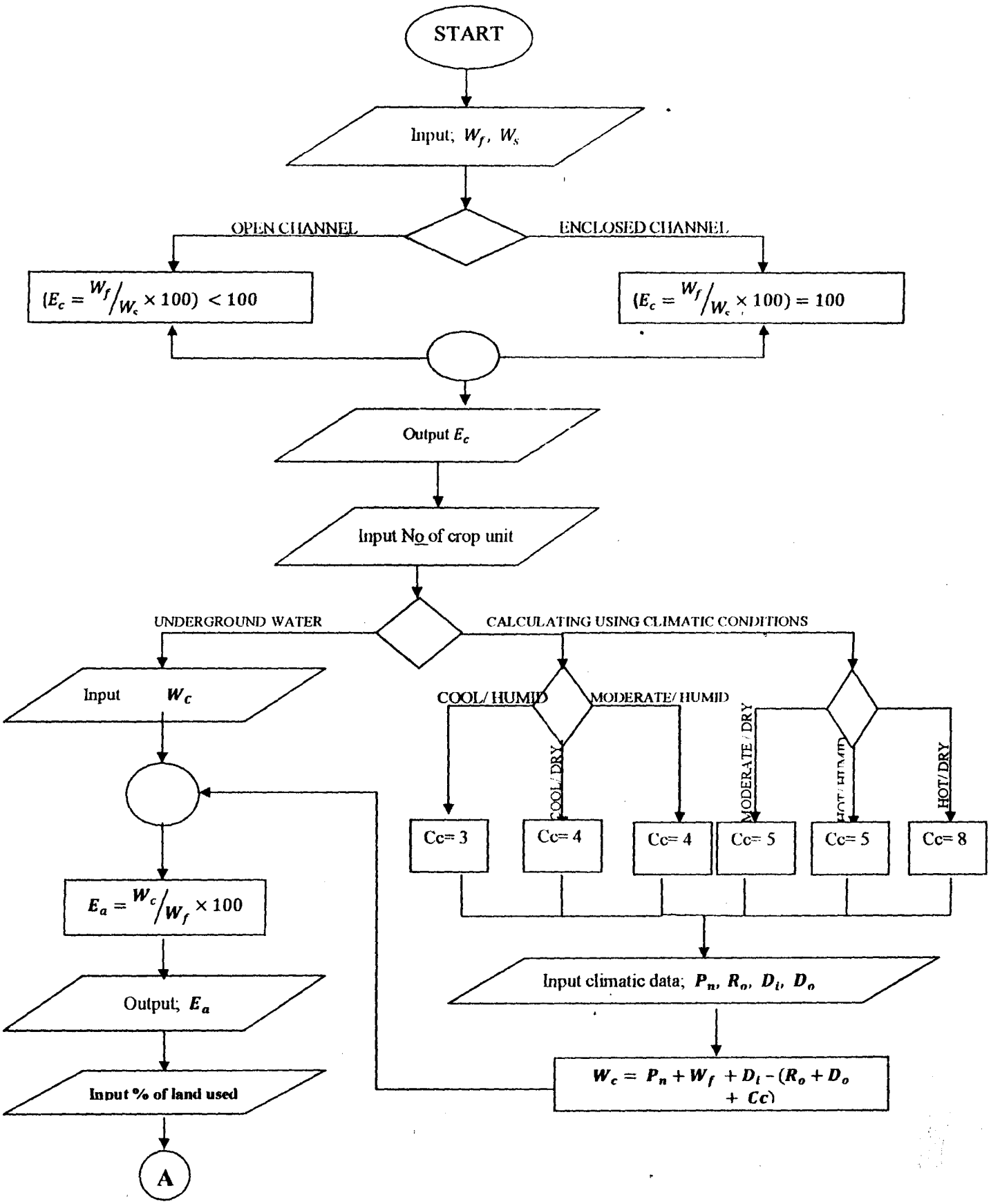
The Consumptive use factors that would affect irrigation and hence the efficiencies, are as follows:

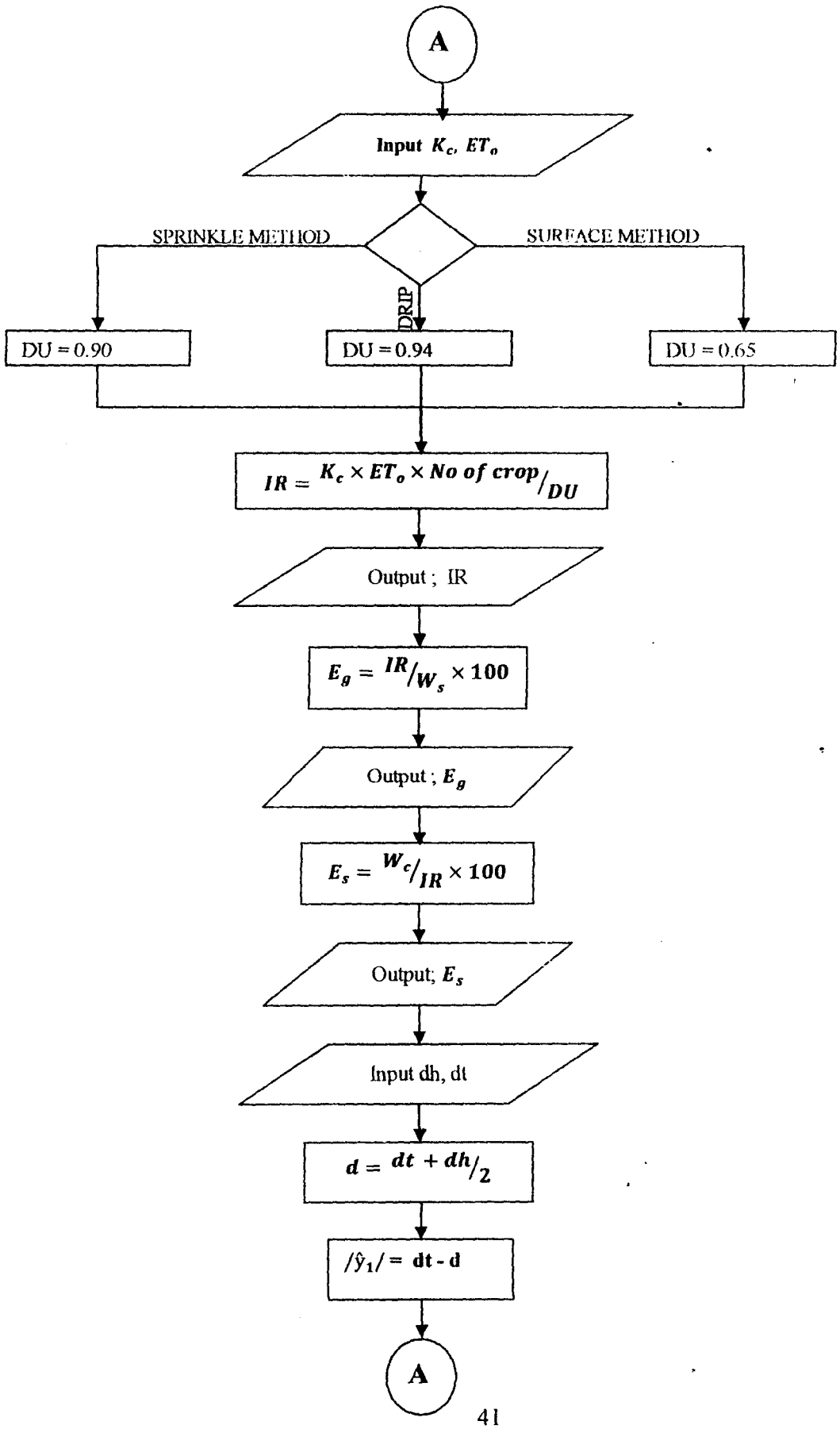
- Crop cover
- Solar radiation(temperature and humidity condition)
- Wind
- Stage of growth of the crop (annual crop)

Table 3.1: Maximum rates of Soil moisture use by crops under different climatic conditions.

Climatic Conditions	**Peak rate of Soil Moisture Removal (mm/day)**
Cool, Humid	3
Cool, Dry	4
Moderate, Humid	4
Moderate, Dry	5
Hot, Humid	5
Hot, Dry	8

Adapted from Molenaar (1960)


START
Input; W_f, W_s
OPEN CHANNEL
ENCLOSED CHANNEL
$(E_c = {}^{W_f}/_{W_s} \times 100) < 100$
$(E_c = {}^{W_f}/_{W_s} \times 100) = 100$
Output E_c
Input No of crop unit
UNDERGROUND WATER
CALCULATING USING CLIMATIC CONDITIONS
Input W_c
COOL/ HUMID
MODERATE/ HUMID
COOL/ DRY
MODERATE / DRY
HOT/ HUMID
HOT/DRY
Cc= 3
Cc= 4
Cc= 4
Cc= 5
Cc= 5
Cc= 8
$E_a = {}^{W_c}/_{W_f} \times 100$
Input climatic data; P_n, R_o, D_i, D_o
Output; E_a
$W_c = P_n + W_f + D_i - (R_o + D_o + Cc)$
Input % of land used
A


A
Input K_c, ET_o
SPRINKLE METHOD
DRIP
SURFACE METHOD
DU = 0.90
DU = 0.94
DU = 0.65
$IR = {K_c \times ET_o \times No\ of\ crop}/{DU}$
Output ; IR
$E_g = {IR}/{W_s} \times 100$
Output ; E_g
$E_s = {W_c}/{IR} \times 100$
Output; E_s
Input dh, dt
$d = {dt + dh}/{2}$
$/\hat{y}_1/ = dt - d$
A

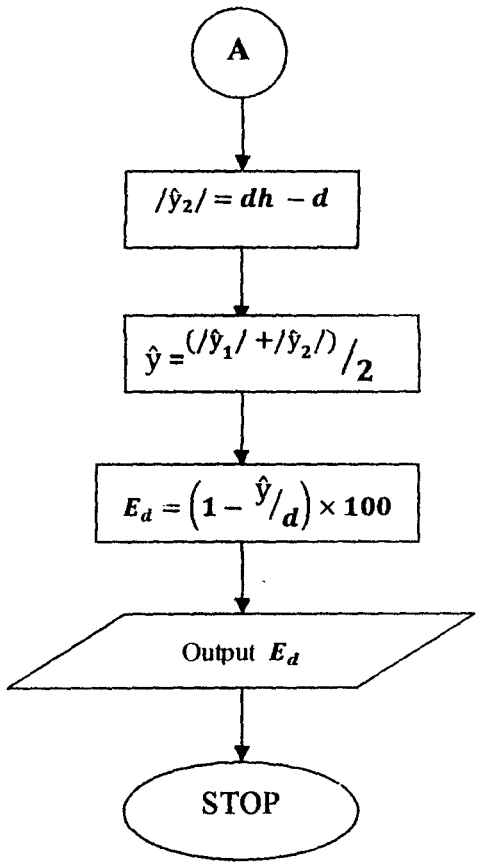


Figure 3.1: The Software Flowchart.

CHAPTER FOUR

4.0 RESULTS AND CALCULATION

4.1 Results

The result achieved clearly shows that the irrigation efficiency calculating program runs effectively and that it complies with data already gotten from calculated irrigation efficiencies when compared as entailed in the objective of this project work.

The screen shots showing the Graphic User Interface (GUI) of the calculator, how to input the value and the resulting efficiencies is displayed below. Also to be noted is the basic unit measurement for all the volume inputs which is (m^3) and the uniformity of the units to be used as stated at the top of the GUI in red to ensure correctness in the answers.

The program calculates the following efficiencies; the conveyance efficiency (E_a), the application efficiency (E_a), the water storage efficiency (E_s), the distribution efficiency (E_d) and the total irrigation efficiency (E_g). The program also calculates the irrigation requirements (IR) of that particular irrigation process, to help the irrigation .

4.2 Calculation

Inorder to ensure that the program is working effectively, it is correct that live scenario questions be solved using the program alongside the manual calculation for comparison sakes. This could also aid the usage of the program, as every unclear step s will be explained. Below is an exercise to be solved.

Exercise 4.1

A stream of 130 liters/s was diverted from a canal and 100 liters/sec were delivered to the field; an area of 1.6 hectares with about 5000 crop units was irrigated in 8 hrs. the effective depth of root zone was 1.7 m. The run-off in the field was 420m^3. The depth of water penetration varies linearly from 1.7 m at head end of the field to 1.1m at the tail end. Available moisture holding capacity of the soil is 20cm/m depth of soil. It is required to determine the conveyance efficiency, water application, water storage efficiency and water distribution efficiency. The irrigation was stated at moisture extraction level of 50% of the available moisture, the crop coefficient (K_c) of the cultivation 0.637, and the potential Evapotranspiration (ET_o) 1.1102.

Solution

i. $E_c = \frac{W_f}{W_s} \times 100 = {}^{100}/_{130} \times 100 = \underline{77\%}$

ii. Water application efficiency $E_a = {}^{W_c}/_{W_f} \times 100$

Volume of water delivered to the plot, $v = qt = 100 litres/sec \times 8 \times 60 \times 60$

$\therefore v = 2880000\ lit. = 2880m^3$ delivered.

but run- off lost = $420m^3$

$$\therefore W_c = 2880m^3 - 420m^3 = 2460m^3$$

$hence, E_a = \frac{2460}{2880} \times 100 = \underline{85.4\%}$

iii. E_s (water storage efficiency) $= \frac{W_c}{W_n} \times 100$

Moisture holding capacity of that zone $= 20\ {}^{cm}/_{m} \times 1.6m = 34\ cm.$

Available moisture at the start of irrigation $= 50/100 \times 34\ cm = 17cm.$

Moisture required in the root zone = $depth \times plot\ area$

$$= (34cm - 17cm) \times 1.6\ ha = \left(\frac{17}{100}\right) m \times (16 \times 10000) m^2 = 2720\ m^3$$

$\therefore E_s = {}^{2460}/_{2720} \times 100 = \underline{90\%}$

iv. Water distribution efficiency $E_d = (1 - {}^{\hat{y}}/_{d}) \times 100$

$$d = \frac{1.7 + 1.1}{2} = 1.4$$

$/\hat{y}_1/$ = deviation from mean (d) at upper end $= (1.7 - 1.4) = 0.3$

$/\hat{y}_2/$ = deviation from mean (d) at lower end $= (1.1 - 1.4) = 0.3$

Hence, overall deviation $\hat{y} = \frac{/\hat{y}_1/ + /\hat{y}_2/}{2} = \frac{0.3+0.3}{2}$, $\hat{y} = 0.3$

$\therefore E_d = \left(1 - \frac{0.3}{1.4}\right) \times 100 = \underline{78.6\%}$

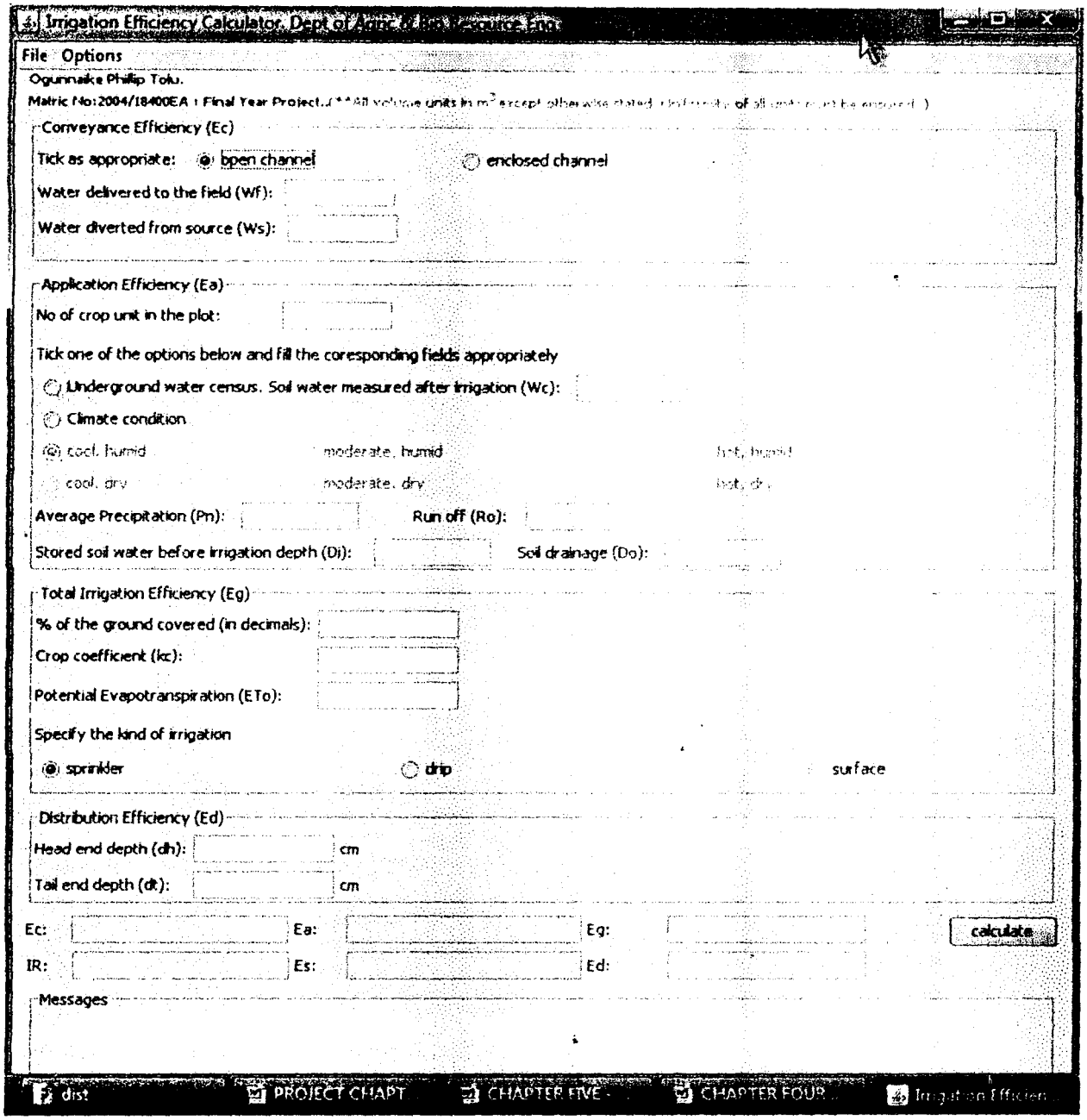


Fig. 4.1. The Graphic User Interface Of the Calculator with no inputs.

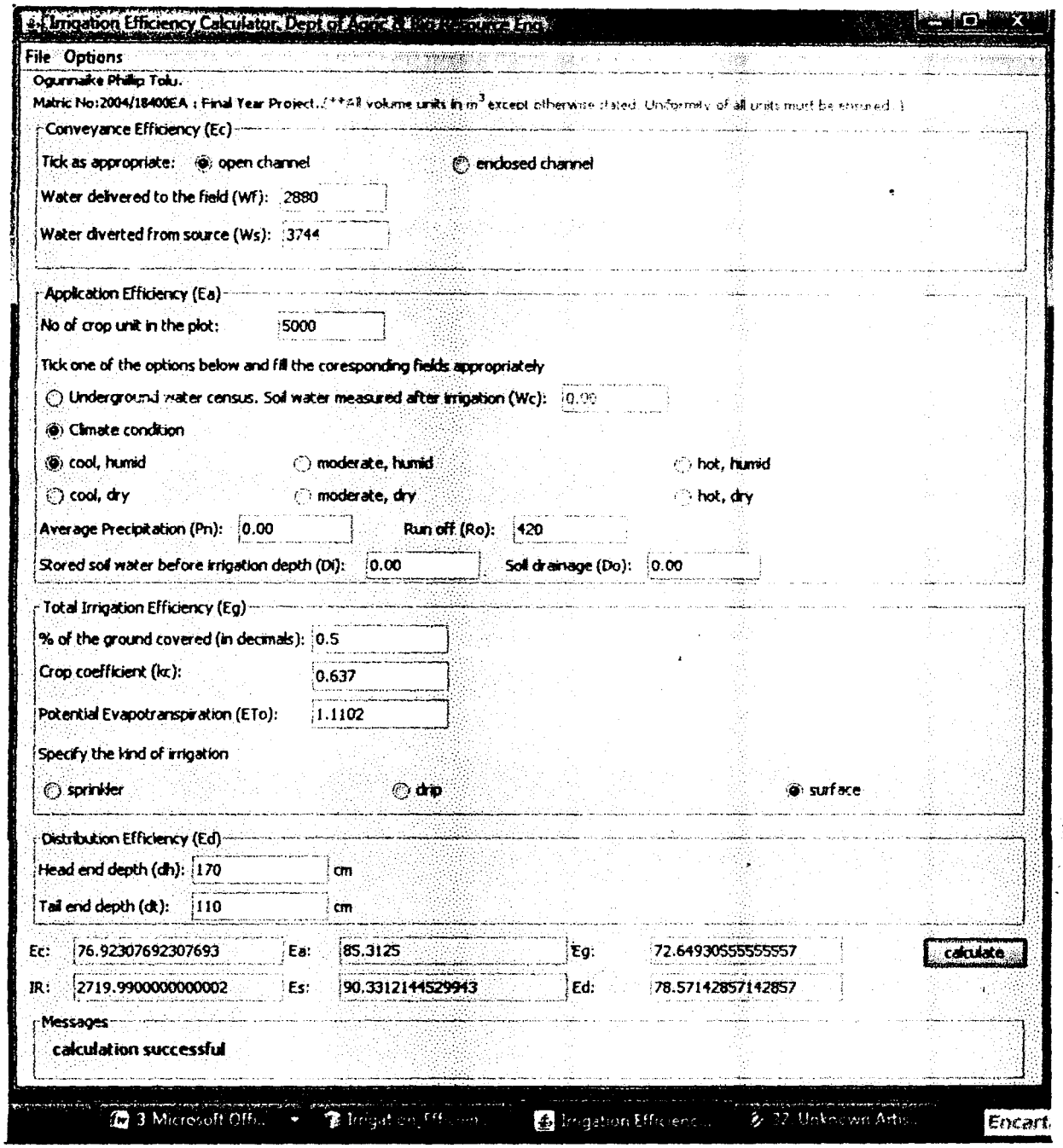


Fig. 4.2. The calculator program calculating the exercise 4.1

CHAPTER FIVE

5.0 CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion.

This project is an irrigation efficiency calculating software written with the Java Language. It helps estimate the crop irrigation water requirement and hence the efficiencies of the irrigation processes: from the application efficiency to the total irrigation efficiency so as to aid the farmer know aside the efficiency, how much water to use as a result of knowing how much of efficiency is gotten with the use of an amount of volume of water. Also, the software gives room for development of recommendation for improvement; the irrigation efficiency calculation under varying water supply conditions. With the ascertaining of the water requirement achieved by the calculator, it therefore serves as a very reliable tool for water- use optimization.

Knowing when crops need water and how much they need are the keys to good water irrigation management. The irrigation efficiency calculator functions based on basic knowledge of soil type and crop water use information; that is the formulas that help analyze the efficiency in the real time. It works with the various parameters fed into it and does not really need any censor to help it calculate the efficiency of the irrigation.

The challenges face that might be faced in using the software would be in the using the program to calculate efficiencies with respect to some other methods of the designing of the software. Also the farmer not knowing how to read and interpret the requirements might not be able to use it; it so therefore entails that the farmer must at least be learned enough to understand the terms to be able to use the software well.

5.2 Recommendations

The priority of Agriculture in Nigeria should be its Science and Technological enhancement in all ramification and hence such project should be more encouraged in higher institutions of learning. Also it obliges to state that this work can be used as a start up basis for other students who intend to work in this line of software design in the future.

REFERENCES

A. M. Michael (1998), Irrigation: Theory And Practice. Vicas publishing house PVT. LTD, New Delhi. Pp. 512-553

As of May 2006 The Encyclopaedia of Computer Languages by Murdoch University, Australia lists 8,512 computer languages.

Anthony Y., Ezedinma F.O.C, Ochapa C.O (1986); Introduction to Tropical Agriculture, Longman, England. Pp. 100-182

Danny H. Rogers, et al. May 1997. Efficiencies and Water losses of Irrigation Systems. Irrigation Management Series. Cooperative Extension Service, Kansas State University, Manhattan.

Egharevba N. A. 2002. Irrigation design and Drainage. Crop water Requirement and Irrigation Frequency. Nma printers and Publishing Co. Ltd. Pp.1-17

Hansen, V. E. 1960. New Concepts in Irrigation Efficiency, Transactions of the ASAE. Vol 3, No. 1, pp. 55-61.

Hansen, V.E, Israelsm O.W and Stringham, G.E (1980); Irrigation Principles and Practices. 4th Edition, John Wiles and Sons. Pp. 112-169.

Irrigation Association Water Management Committee. (2002). *Turf and Landscape Irrigation Best Management Practices.* (www.irrigation.org). Falls Church, VA: Irrigation Association.

Irrigation Association Water Management Committee. (2004). *Irrigation efficiency.* Second release (www.irrigation.org). Falls Church, VA: Irrigation Association.

Karmeli, D., G. Pen & M. Todes, 1985. Irrigation systems: design and operation. Oxford University Press, Cape town, South Africa

Kemper, W. D.; Trout, T. J., Humpherys, A. S., and Bullock, M. S. (1988). "Mechanisms by which surge irrigation reduces furrow infiltration rates in a silty loam soil". *Transactions of the ASAE* **31** (3): 921-829.

Larry G.J. (1988) Principles of Farm Irrigation System Design, USA, Krieger publisher company low land. Pp.40-82

Lenselink K. J., M. Jurriens, 1993. An Inventory Of Irrigation Software For Microcomputers. International Institute for Land Reclamation and Improvement/ ILRI, Wageningen, the Netherlands. Pgs.1, 6-8.

Michael Sipser (1997). *Introduction to the Theory of Computation*. PWS Publishing. ISBN 0-534-94728-X. Section 2.2: Pushdown Automata, pp.101–114.

Osunde A.O (2000); Introduction to Soil Science. Unpublished. Lecture notes, Fderal University of Technology, Minna.

Schneider, A.D. and T.A. Howell. 1993. Reducing Sprinkler Water Losses. Proceedings of the Central Plains Irrigation Short Course. Sterling, CO. Feb. 2 and 3. pp. 43-47.

Verwey, A. *(ed.),* 1985. International colloquium on the role of micro-computers in hydraulic and hydrological research and education. UNESCO, Paris, France.

Wild A. (1988) Russell's soil condition and plant growth 11th Edition U.K. Longman Scientific & Technology. Pp.65-99

Walker, W.R.; Skogerboe, G.V. (1987). *Surface irrigation: Theory and practice*. Prentice-Hall, Englewood Cliffs

APPENDIX

Java Source Codes.

```
/*
 * calcFrame.java
 *
 * Created on September 5, 2009, 11:15 AM
 */
package com.frame;

import com.sun.java.swing.plaf.windows.WindowsLookAndFeel;
import java.awt.Color;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JColorChooser;
import javax.swing.SwingUtilities;
import javax.swing.UIManager;
import javax.swing.UnsupportedLookAndFeelException;

/**
 *
 * @author  philip
 */
public class calcFrame extends javax.swing.JFrame {

    /** Creates new form calcFrame */
    public calcFrame() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        typeOfChannel = new javax.swing.ButtonGroup();
        undergroundOrClimate = new javax.swing.ButtonGroup();
        typeOfClimate = new javax.swing.ButtonGroup();
        kindOfIrrigation = new javax.swing.ButtonGroup();
        messagePanel = new javax.swing.JPanel();
        messageLabel = new javax.swing.JLabel();
        jLabel1 = new javax.swing.JLabel();
        openChannel = new javax.swing.JRadioButton();
        enclosedChannel = new javax.swing.JRadioButton();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        wfTextField = new javax.swing.JTextField();
        wsTextField = new javax.swing.JTextField();
        messagePanel1 = new javax.swing.JPanel();
        messageLabel1 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        noOfCropTextField = new javax.swing.JTextField();
        undergroundWaterCensus = new javax.swing.JRadioButton();
        wcTextField = new javax.swing.JTextField();
        climateCondition = new javax.swing.JRadioButton();
        jLabel11 = new javax.swing.JLabel();
        coolHumid = new javax.swing.JRadioButton();
        moderateHumid = new javax.swing.JRadioButton();
        hotHumid = new javax.swing.JRadioButton();
        coolDry = new javax.swing.JRadioButton();
        moderateDry = new javax.swing.JRadioButton();
```

```
62          hotDry = new javax.swing.JRadioButton();
63          jLabel12 = new javax.swing.JLabel();
64          pnTextField = new javax.swing.JTextField();
65          jLabel13 = new javax.swing.JLabel();
66          diTextField = new javax.swing.JTextField();
67          jLabel14 = new javax.swing.JLabel();
68          roTextField = new javax.swing.JTextField();
69          jLabel15 = new javax.swing.JLabel();
70          doTextField = new javax.swing.JTextField();
71          messagePanel2 = new javax.swing.JPanel();
72          messageLabel2 = new javax.swing.JLabel();
73          jLabel6 = new javax.swing.JLabel();
74          percentageGroundCoveredTextField = new javax.swing.JTextField();
75          jLabel7 = new javax.swing.JLabel();
76          kcTextField = new javax.swing.JTextField();
77          jLabel8 = new javax.swing.JLabel();
78          etoTextField = new javax.swing.JTextField();
79          jLabel10 = new javax.swing.JLabel();
80          sprinkler = new javax.swing.JRadioButton();
81          drip = new javax.swing.JRadioButton();
82          surface = new javax.swing.JRadioButton();
83          calcButton = new javax.swing.JButton();
84          jLabel16 = new javax.swing.JLabel();
85          ecTextField = new javax.swing.JTextField();
86          jLabel17 = new javax.swing.JLabel();
87          eaTextField = new javax.swing.JTextField();
88          egTextField = new javax.swing.JTextField();
89          jLabel18 = new javax.swing.JLabel();
90          jLabel19 = new javax.swing.JLabel();
91          esTextField = new javax.swing.JTextField();
92          jLabel20 = new javax.swing.JLabel();
93          irTextField = new javax.swing.JTextField();
94          messagePanel3 = new javax.swing.JPanel();
95          messageLabel3 = new javax.swing.JLabel();
96          jLabel21 = new javax.swing.JLabel();
97          headEndDepthTextField = new javax.swing.JTextField();
98          jLabel22 = new javax.swing.JLabel();
99          tailEndDepthTextField = new javax.swing.JTextField();
00          jLabel9 = new javax.swing.JLabel();
101         jLabel23 = new javax.swing.JLabel();
02          jLabel26 = new javax.swing.JLabel();
03          edTextField = new javax.swing.JTextField();
04          messagePanel4 = new javax.swing.JPanel();
05          messageLabel4 = new javax.swing.JLabel();
06          messageTextLabel = new javax.swing.JLabel();
07          jLabel4 = new javax.swing.JLabel();
08          jMenuBar5 = new javax.swing.JMenuBar();
09          jMenu9 = new javax.swing.JMenu();
10          jMenuItem1 = new javax.swing.JMenuItem();
11          jMenu10 = new javax.swing.JMenu();
12          backgroundItem = new javax.swing.JMenuItem();
13
14          typeOfChannel.add(openChannel);
15          typeOfChannel.add(enclosedChannel);
16
17          undergroundOrClimate.add(undergroundWaterCensus);
18          undergroundOrClimate.add(climateCondition);
19
20          typeOfClimate.add(coolHumid);
21          typeOfClimate.add(coolDry);
22          typeOfClimate.add(moderateHumid);
23          typeOfClimate.add(moderateDry);
24          typeOfClimate.add(hotHumid);
25          typeOfClimate.add(hotDry);
26
```

```
        kindOfIrrigation.add(sprinkler);
        kindOfIrrigation.add(drip);
        kindOfIrrigation.add(surface);

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Irrigation Efficiency Calculator. Dept of Agric & Bio Resource Eng");
        setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));


ssagePanel.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory.
eateEtchedBorder(), "Conveyance Efficiency (Ec)"));

        jLabel1.setText("Tick as appropriate: ");

        openChannel.setSelected(true);
        openChannel.setText("open channel");
        openChannel.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                openChannelActionPerformed(evt);
            }
        });

        enclosedChannel.setText("enclosed channel");
        enclosedChannel.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                enclosedChannelActionPerformed(evt);
            }
        });

        jLabel2.setText("Water delivered to the field (Wf): ");

        jLabel3.setText("Water diverted from source (Ws): ");

        wfTextField.addCaretListener(new javax.swing.event.CaretListener() {
            public void caretUpdate(javax.swing.event.CaretEvent evt) {
                wfTextFieldCaretUpdate(evt);
            }
        });
        wfTextField.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                wfTextFieldActionPerformed(evt);
            }
        });

        wsTextField.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                wsTextFieldActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout messagePanelLayout = new
vax.swing.GroupLayout(messagePanel);
        messagePanel.setLayout(messagePanelLayout);
        messagePanelLayout.setHorizontalGroup(

sagePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(messagePanelLayout.createSequentialGroup()

dGroup(messagePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(messageLabel)
                    .addGroup(messagePanelLayout.createSequentialGroup()
                        .addComponent(jLabel2)

dPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
85                          .addComponent(wfTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, 75, javax.swing.GroupLayout.PREFERRED_SIZE))
86                      .addGroup(messagePanelLayout.createSequentialGroup()
87                          .addComponent(jLabel3)
88
ddPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
89                          .addComponent(wsTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, 75, javax.swing.GroupLayout.PREFERRED_SIZE))
90                      .addGroup(messagePanelLayout.createSequentialGroup()
91                          .addComponent(jLabel1)
92
ddPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
93                          .addComponent(openChannel)
94                          .addGap(92, 92, 92)
95                          .addComponent(enclosedChannel)))
96                  .addGap(238, 238, 238))
97          );
98          messagePanelLayout.setVerticalGroup(
99
ssagePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00              .addGroup(messagePanelLayout.createSequentialGroup()
201
ddGroup(messagePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
02                      .addComponent(messageLabel)
03
dGroup(messagePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
04                          .addComponent(jLabel1)
05                          .addComponent(openChannel)
06                          .addComponent(enclosedChannel)))
07                  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
08
ddGroup(messagePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
09                      .addComponent(jLabel2)
10                      .addComponent(wfTextField, javax.swing.GroupLayout.PREFERRED_SIZE,
vax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
11                  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
12
ddGroup(messagePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
13                      .addComponent(jLabel3)
14                      .addComponent(wsTextField, javax.swing.GroupLayout.PREFERRED_SIZE,
avax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
15                  .addContainerGap())
16          );
17
18
ssagePanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory
reateEtchedBorder(), "Application Efficiency (Ea)"));
19
20          jLabel5.setText("No of crop unit in the plot:");
21
22          undergroundWaterCensus.setText("Underground water census. Soil water measured
er irrigation (Wc):");
23          undergroundWaterCensus.addActionListener(new java.awt.event.ActionListener() {
24              public void actionPerformed(java.awt.event.ActionEvent evt) {
25                  undergroundWaterCensusActionPerformed(evt);
26              }
27          });
28
29          wcTextField.setEnabled(false);
30          wcTextField.addActionListener(new java.awt.event.ActionListener() {
31              public void actionPerformed(java.awt.event.ActionEvent evt) {
32                  wcTextFieldActionPerformed(evt);
33              }
34          });
35
```

```
        climateCondition.setText("Climate condition");
        climateCondition.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                climateConditionActionPerformed(evt);
            }
        });

        jLabel11.setText("Tick one of the options below and fill the corresponding fields
propriately");

        coolHumid.setSelected(true);
        coolHumid.setText("cool, humid");
        coolHumid.setEnabled(false);

        moderateHumid.setText("moderate, humid");
        moderateHumid.setEnabled(false);

        hotHumid.setText("hot, humid");
        hotHumid.setEnabled(false);

        coolDry.setText("cool, dry");
        coolDry.setEnabled(false);

        moderateDry.setText("moderate, dry");
        moderateDry.setEnabled(false);

        hotDry.setText("hot, dry");
        hotDry.setEnabled(false);

        jLabel12.setText("Average Precipitation (Pn):");

        pnTextField.setEnabled(false);

        jLabel13.setText("Stored soil water before irrigation depth (Di):");

        diTextField.setEnabled(false);

        jLabel14.setText("Run off (Ro):");

        roTextField.setEnabled(false);

        jLabel15.setText("Soil drainage (Do):");

        doTextField.setEnabled(false);

        javax.swing.GroupLayout messagePanel1Layout = new
vax.swing.GroupLayout(messagePanel1);
        messagePanel1.setLayout(messagePanel1Layout);
        messagePanel1Layout.setHorizontalGroup(

sagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(messagePanel1Layout.createSequentialGroup()

dGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(messagePanel1Layout.createSequentialGroup()

dGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(messageLabel1)
                            .addComponent(jLabel5))
                        .addGap(41, 41, 41)
                        .addComponent(noOfCropTextField,
ax.swing.GroupLayout.PREFERRED_SIZE, 75, javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(messagePanel1Layout.createSequentialGroup()
                        .addComponent(undergroundWaterCensus)
```

```
294
addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
295                         .addComponent(wcTextField,
avax.swing.GroupLayout.PREFERRED_SIZE, 75, javax.swing.GroupLayout.PREFERRED_SIZE))
296                     .addComponent(climateCondition)
297                     .addComponent(jLabel11, javax.swing.GroupLayout.PREFERRED_SIZE, 395,
avax.swing.GroupLayout.PREFERRED_SIZE)
298                     .addGroup(messagePanel1Layout.createSequentialGroup()
299
addGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
300                             .addComponent(coolHumid)
301                             .addComponent(coolDry))
302                         .addGap(96, 96, 96)
303
ddGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
304                             .addComponent(moderateDry)
305                             .addComponent(moderateHumid))
306                         .addGap(160, 160, 160)
307
ddGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
308                             .addComponent(hotDry)
309                             .addComponent(hotHumid)))
310                     .addGroup(messagePanel1Layout.createSequentialGroup()
311                         .addComponent(jLabel12)
312
ddPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
313                         .addComponent(pnTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, 80, javax.swing.GroupLayout.PREFERRED_SIZE)
314                         .addGap(37, 37, 37)
315                         .addComponent(jLabel14, javax.swing.GroupLayout.PREFERRED_SIZE,
, javax.swing.GroupLayout.PREFERRED_SIZE)
316
ddPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
17                          .addComponent(roTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, 80, javax.swing.GroupLayout.PREFERRED_SIZE))
18                      .addGroup(messagePanel1Layout.createSequentialGroup()
19                          .addComponent(jLabel13, javax.swing.GroupLayout.PREFERRED_SIZE,
6, javax.swing.GroupLayout.PREFERRED_SIZE)
320
addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
321                         .addComponent(diTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, 60, javax.swing.GroupLayout.PREFERRED_SIZE)
22                          .addGap(18, 18, 18)
23                          .addComponent(jLabel15)
24
ddPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
25                          .addComponent(doTextField,
avax.swing.GroupLayout.PREFERRED_SIZE, 80, javax.swing.GroupLayout.PREFERRED_SIZE)))
26                  .addContainerGap(111, Short.MAX_VALUE))
27          );
28          messagePanel1Layout.setVerticalGroup(
29
ssagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
30              .addGroup(messagePanel1Layout.createSequentialGroup()
331
ddGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
32                      .addComponent(messageLabel1)
33
ldGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
34                          .addComponent(jLabel5)
35                          .addComponent(noOfCropTextField,
avax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
avax.swing.GroupLayout.PREFERRED_SIZE)))
36                  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
37                  .addComponent(jLabel11)
```

```
338                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
339
addGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
340                         .addComponent(undergroundWaterCensus,
avax.swing.GroupLayout.PREFERRED_SIZE, 14, javax.swing.GroupLayout.PREFERRED_SIZE)
341                         .addComponent(wcTextField, javax.swing.GroupLayout.PREFERRED_SIZE,
avax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
342                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
343                     .addComponent(climateCondition)
344                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
345
ddGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
346                         .addComponent(coolHumid)
347                         .addComponent(hotHumid)
348                         .addComponent(moderateHumid))
349                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
350
ddGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
351                         .addComponent(coolDry)
352                         .addComponent(hotDry)
353                         .addComponent(moderateDry))
354                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
355
addGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
356                         .addComponent(jLabel12)
357                         .addComponent(pnTextField, javax.swing.GroupLayout.PREFERRED_SIZE,
vax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
358                         .addComponent(jLabel14)
359                         .addComponent(roTextField, javax.swing.GroupLayout.PREFERRED_SIZE,
vax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
360                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
361
ddGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
362
addGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
363                             .addComponent(jLabel13)
364                             .addComponent(diTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.PREFERRED_SIZE))
365
ddGroup(messagePanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
366                             .addComponent(jLabel15)
367                             .addComponent(doTextField,
avax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.PREFERRED_SIZE))))
368         );
369
370
ssagePanel2.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory
reateEtchedBorder(), "Total Irrigation Efficiency (Eg)"));
371
372         jLabel6.setText("% of the ground covered (in decimals):");
373
374         percentageGroundCoveredTextField.addActionListener(new
a.awt.event.ActionListener() {
375             public void actionPerformed(java.awt.event.ActionEvent evt) {
376                 percentageGroundCoveredTextFieldActionPerformed(evt);
377             }
378         });
379
380         jLabel7.setText("Crop coefficient (kc):");
381
382         kcTextField.addActionListener(new java.awt.event.ActionListener() {
383             public void actionPerformed(java.awt.event.ActionEvent evt) {
384                 kcTextFieldActionPerformed(evt);
```

```
85              }
86          });
87
88          jLabel8.setText("Potential Evapotranspiration (ETo):");
89
90          etoTextField.addActionListener(new java.awt.event.ActionListener() {
91              public void actionPerformed(java.awt.event.ActionEvent evt) {
92                  etoTextFieldActionPerformed(evt);
93              }
94          });
95
96          jLabel10.setText("Specify the kind of irrigation");
97
98          sprinkler.setSelected(true);
99          sprinkler.setText("sprinkler");
00
01          drip.setText("drip");
02
03          surface.setText("surface");
04
05          javax.swing.GroupLayout messagePanel2Layout = new
vax.swing.GroupLayout(messagePanel2);
06          messagePanel2.setLayout(messagePanel2Layout);
07          messagePanel2Layout.setHorizontalGroup(
08
sagePanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
09              .addGroup(messagePanel2Layout.createSequentialGroup()
10                  .addComponent(messageLabel2)
11                  .addGap(630, 630, 630))
12              .addGroup(messagePanel2Layout.createSequentialGroup()
13
ldGroup(messagePanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING,
se)
14                      .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
sagePanel2Layout.createSequentialGroup()
15                          .addComponent(sprinkler)
16
dPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
ax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
17                          .addComponent(drip))
18                      .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
ssagePanel2Layout.createSequentialGroup()
19                          .addComponent(jLabel7)
20
dPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
ax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
21                          .addComponent(kcTextField,
ax.swing.GroupLayout.PREFERRED_SIZE, 96, javax.swing.GroupLayout.PREFERRED_SIZE))
22                      .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
ssagePanel2Layout.createSequentialGroup()
23                          .addComponent(jLabel6)
24
dPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
25                          .addComponent(percentageGroundCoveredTextField,
ax.swing.GroupLayout.PREFERRED_SIZE, 96, javax.swing.GroupLayout.PREFERRED_SIZE))
26                      .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
sagePanel2Layout.createSequentialGroup()
27
ddGroup(messagePanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING,
se)
33                              .addComponent(jLabel10,
ax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
ax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
429                                 .addComponent(jLabel8,
avax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
avax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
430
addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
avax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
431                             .addComponent(etoTextField,
avax.swing.GroupLayout.PREFERRED_SIZE, 96, javax.swing.GroupLayout.PREFERRED_SIZE))
432                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 173,
hort.MAX_VALUE)
433                     .addComponent(surface)
434                     .addGap(108, 108, 108))
435         );
436         messagePanel2Layout.setVerticalGroup(
437
essagePanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
438             .addGroup(messagePanel2Layout.createSequentialGroup()
439
addGroup(messagePanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
440                         .addComponent(messageLabel2)
441
addGroup(messagePanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
442                             .addComponent(jLabel6)
443                             .addComponent(percentageGroundCoveredTextField,
avax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
avax.swing.GroupLayout.PREFERRED_SIZE)))
444                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
445
addGroup(messagePanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
446                         .addComponent(jLabel7)
447                         .addComponent(kcTextField, javax.swing.GroupLayout.PREFERRED_SIZE,
avax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
448                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
449
addGroup(messagePanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
450                         .addComponent(jLabel8)
451                         .addComponent(etoTextField, javax.swing.GroupLayout.PREFERRED_SIZE,
avax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
452                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
453                     .addComponent(jLabel10)
454                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
455
addGroup(messagePanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
456                         .addComponent(sprinkler)
457                         .addComponent(surface)
458                         .addComponent(drip))
459                     .addContainerGap(3, Short.MAX_VALUE))
460         );
461
462         calcButton.setText("calculate");
463         calcButton.addActionListener(new java.awt.event.ActionListener() {
464             public void actionPerformed(java.awt.event.ActionEvent evt) {
465                 calcButtonActionPerformed(evt);
466             }
467         });
468
469         jLabel16.setText("Ec:");
470
471         ecTextField.setEditable(false);
472
473         jLabel17.setText("Ea:");
474
475         eaTextField.setEditable(false);
476
477         egTextField.setEditable(false);
```

```
478
479         jLabel18.setText("Eq:");
480
481         jLabel19.setText("Es:");
482
483         esTextField.setEditable(false);
484
485         jLabel20.setText("IR:");
486
487         irTextField.setEditable(false);
488         irTextField.addActionListener(new java.awt.event.ActionListener() {
489             public void actionPerformed(java.awt.event.ActionEvent evt) {
490                 irTextFieldActionPerformed(evt);
491             }
492         });
493
494
ssagePanel3.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory
createEtchedBorder(), "Distribution Efficiency (Ed)"));
495
496         jLabel21.setText("Head end depth (dh):");
497
498         headEndDepthTextField.addActionListener(new java.awt.event.ActionListener() {
499             public void actionPerformed(java.awt.event.ActionEvent evt) {
500                 headEndDepthTextFieldActionPerformed(evt);
501             }
502         });
503
504         jLabel22.setText("Tail end depth (dt):");
505
506         tailEndDepthTextField.addActionListener(new java.awt.event.ActionListener() {
507             public void actionPerformed(java.awt.event.ActionEvent evt) {
508                 tailEndDepthTextFieldActionPerformed(evt);
509             }
510         });
511
512         jLabel9.setText("cm");
513
514         jLabel23.setText("cm");
515
516         javax.swing.GroupLayout messagePanel3Layout = new
avax.swing.GroupLayout(messagePanel3);
517         messagePanel3.setLayout(messagePanel3Layout);
518         messagePanel3Layout.setHorizontalGroup(
519
sagePanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
520             .addGroup(messagePanel3Layout.createSequentialGroup()
521
ddGroup(messagePanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
522                     .addComponent(messageLabel3)
523                     .addGroup(messagePanel3Layout.createSequentialGroup()
524
dGroup(messagePanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
lse)
525                                 .addComponent(jLabel22, javax.swing.GroupLayout.DEFAULT_SIZE,
a.ax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
526                                 .addComponent(jLabel21, javax.swing.GroupLayout.DEFAULT_SIZE,
a ax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
527
dPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
528
ddGroup(messagePanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
529                                 .addGroup(messagePanel3Layout.createSequentialGroup()
530                                     .addComponent(tailEndDepthTextField,
avax.swing.GroupLayout.PREFERRED_SIZE, 96, javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
31
ddPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
32                                      .addComponent(jLabel23))
33                                  .addGroup(messagePanel3Layout.createSequentialGroup()
34                                      .addComponent(headEndDepthTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, 96, javax.swing.GroupLayout.PREFERRED_SIZE)
35
ddPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
36                                      .addComponent(jLabel9))))
37                  .addContainerGap(409, Short.MAX_VALUE))
38          );
39          messagePanel3Layout.setVerticalGroup(
40
ssagePanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
41              .addGroup(messagePanel3Layout.createSequentialGroup()
42
ddGroup(messagePanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
43                      .addComponent(messageLabel3)
44
ddGroup(messagePanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
45                          .addComponent(jLabel21)
46                          .addComponent(headEndDepthTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.PREFERRED_SIZE)
47                          .addComponent(jLabel9)))
48                  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
49
ddGroup(messagePanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
50                      .addComponent(jLabel22)
51                      .addComponent(tailEndDepthTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.PREFERRED_SIZE)
52                      .addComponent(jLabel23)))
53          );
54
55          jLabel26.setText("Ed:");
56
57          edTextField.setEditable(false);
58
59
essagePanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory
reateEtchedBorder(), "Messages"));
60
61          messageTextLabel.setFont(new java.awt.Font("Tahoma", 1, 11));
62          messageTextLabel.setForeground(new java.awt.Color(51, 204, 0));
63          messageTextLabel.setOpaque(true);
64
65          javax.swing.GroupLayout messagePanel4Layout = new
vax.swing.GroupLayout(messagePanel4);
66          messagePanel4.setLayout(messagePanel4Layout);
67          messagePanel4Layout.setHorizontalGroup(
68
sagePanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
69              .addGroup(messagePanel4Layout.createSequentialGroup()
70                  .addComponent(messageLabel4)
571                 .addGap(551, 551, 551))
572             .addGroup(messagePanel4Layout.createSequentialGroup()
573                 .addGap(10, 10, 10)
574                 .addComponent(messageTextLabel, javax.swing.GroupLayout.DEFAULT_SIZE,
, Short.MAX_VALUE)
75                  .addContainerGap())
76          );
77          messagePanel4Layout.setVerticalGroup(
78
essagePanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
79              .addGroup(messagePanel4Layout.createSequentialGroup()
80
ddGroup(messagePanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
81                      .addComponent(messageLabel4)
82                      .addComponent(messageTextLabel, javax.swing.GroupLayout.DEFAULT_SIZE,
, Short.MAX_VALUE))
83                  .addContainerGap())
84          );
85
86          jLabel4.setFont(new java.awt.Font("Tahoma", 0, 10)); // NOI18N
87          jLabel4.setText("<html>Ogunnaike Phillip Tolu.<br>Matric No:2004/18400EA  : Final
ar Project..<font color=\"red\">(**All volume units in m<sup>3</sup> except otherwise
ated. Uniformity of all units must be  ensured. )</font></html>");
88
89          jMenu9.setText("File");
90
91          jMenuItem1.setText("Exit");
92          jMenuItem1.addActionListener(new java.awt.event.ActionListener() {
93              public void actionPerformed(java.awt.event.ActionEvent evt) {
94                  jMenuItem1ActionPerformed(evt);
95              }
96          });
97          jMenu9.add(jMenuItem1);
98
99          jMenuBar5.add(jMenu9);
00
01          jMenu10.setText("Options");
02
03          backgroundItem.setText("Change Background Color");
04          backgroundItem.addActionListener(new java.awt.event.ActionListener() {
05              public void actionPerformed(java.awt.event.ActionEvent evt) {
06                  backgroundItemActionPerformed(evt);
07              }
08          });
09          jMenu10.add(backgroundItem);
10
11          jMenuBar5.add(jMenu10);
12
13          setJMenuBar(jMenuBar5);
14
15          javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
16          getContentPane().setLayout(layout);
17          layout.setHorizontalGroup(
18              layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
19              .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
yout.createSequentialGroup()
20                  .addContainerGap()
21
ddGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
22                      .addComponent(messagePanel4,
vax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
rax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
23                      .addComponent(messagePanel1,
vax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
24                      .addComponent(messagePanel2,
vax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
25                      .addComponent(messagePanel3, javax.swing.GroupLayout.DEFAULT_SIZE,
avax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
26                      .addGroup(layout.createSequentialGroup()
27
addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
28                              .addComponent(jLabel16,
avax.swing.GroupLayout.PREFERRED_SIZE, 25, javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
9                                   .addComponent(jLabel20,
ax.swing.GroupLayout.PREFERRED_SIZE, 27, javax.swing.GroupLayout.PREFERRED_SIZE))
0
dPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
1
dGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)
2                                   .addComponent(irTextField,
ax.swing.GroupLayout.Alignment.LEADING)
3                                   .addComponent(ecTextField,
ax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE, 448,
rt.MAX_VALUE))
4                               .addGap(4, 4, 4)
5
dGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
6                                   .addComponent(jLabel17,
ax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT_SIZE,
ax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
7                                   .addComponent(jLabel19,
ax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED_SIZE, 25,
vax.swing.GroupLayout.PREFERRED_SIZE))
8
ddPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
9
dGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
0                                   .addComponent(eaTextField,
ax.swing.GroupLayout.DEFAULT_SIZE, 130, Short.MAX_VALUE)
1                                   .addComponent(esTextField,
ax.swing.GroupLayout.DEFAULT_SIZE, 130, Short.MAX_VALUE))
2
dPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
3
dGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
4                                   .addComponent(jLabel18,
ax.swing.GroupLayout.PREFERRED_SIZE, 25, javax.swing.GroupLayout.PREFERRED_SIZE)
5                                   .addComponent(jLabel26,
vax.swing.GroupLayout.PREFERRED_SIZE, 25, javax.swing.GroupLayout.PREFERRED_SIZE))
6                               .addGap(29, 29, 29)
7
dGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
8                                   .addGroup(layout.createSequentialGroup()
9                                       .addComponent(egTextField,
ax.swing.GroupLayout.PREFERRED_SIZE, 135, javax.swing.GroupLayout.PREFERRED_SIZE)
0
dPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 26, Short.MAX_VALUE)
1                                       .addComponent(calcButton))
2                                   .addComponent(edTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, 135, javax.swing.GroupLayout.PREFERRED_SIZE)))
3                           .addComponent(messagePanel,
vax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE, 642,
t.MAX_VALUE)
4                           .addComponent(jLabel4, javax.swing.GroupLayout.Alignment.LEADING,
ax.swing.GroupLayout.DEFAULT_SIZE, 642, Short.MAX_VALUE))
5                       .addContainerGap())
6           );
7           layout.setVerticalGroup(
8               layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
9               .addGroup(layout.createSequentialGroup()
0                   .addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED_SIZE, 26,
ax.swing.GroupLayout.PREFERRED_SIZE)
1                   .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
2                   .addComponent(messagePanel, javax.swing.GroupLayout.PREFERRED_SIZE,
vax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
3                   .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
4                   .addComponent(messagePanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
vax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(messagePanel2, javax.swing.GroupLayout.PREFERRED_SIZE, 156,
vax.swing.GroupLayout.PREFERRED_SIZE)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(messagePanel3, javax.swing.GroupLayout.PREFERRED_SIZE,
vax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

ddGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(calcButton)
                        .addGroup(layout.createSequentialGroup()

ddGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                                .addComponent(jLabel16)
 .addComponent(ecTextField, javax.swing.GroupLayout.PREFERRED_SIZE,
vax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(eaTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(egTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(jLabel18)
                                .addComponent(jLabel17))

dPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

dGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                                .addComponent(jLabel20)
                                .addComponent(irTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(esTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(edTextField,
vax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(jLabel26)
                                .addComponent(jLabel19))))
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(messagePanel4, javax.swing.GroupLayout.DEFAULT_SIZE,
vax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addContainerGap())
            );

            pack();
        }// </editor-fold>

        private void wsTextFieldActionPerformed(java.awt.event.ActionEvent evt) {
            // TODO add your handling code here:
    }

        private void undergroundWaterCensusActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
            if (this.undergroundWaterCensus.isSelected()) {

                this.wcTextField.setEnabled(true);
                this.coolHumid.setEnabled(false);
                this.coolDry.setEnabled(false);
                this.moderateHumid.setEnabled(false);
                this.moderateDry.setEnabled(false);
                this.hotHumid.setEnabled(false);
                this.hotDry.setEnabled(false);
                pnTextField.setEnabled(false);
```

```
712             roTextField.setEnabled(false);
713             diTextField.setEnabled(false);
714             doTextField.setEnabled(false);
715
716         } else if (this.climateCondition.isSelected()) {
717             this.coolHumid.setEnabled(true);
718             this.coolDry.setEnabled(true);
719             this.moderateHumid.setEnabled(true);
720             this.moderateDry.setEnabled(true);
721             this.hotHumid.setEnabled(true);
722             this.hotDry.setEnabled(true);
723             this.wcTextField.setEnabled(false);
724         }
725
726 }
727
728     private void wcTextFieldActionPerformed(java.awt.event.ActionEvent evt) {
729         // TODO add your handling code here:
730 }
731
732     private void openChannelActionPerformed(java.awt.event.ActionEvent evt) {
733         // TODO add your handling code here:
734 }
735
736     private void enclosedChannelActionPerformed(java.awt.event.ActionEvent evt) {
737         // TODO add your handling code here:
738 }
739
740     private void
rcentageGroundCoveredTextFieldActionPerformed(java.awt.event.ActionEvent evt) {
741         // TODO add your handling code here:
742 }
743
744     private void kcTextFieldActionPerformed(java.awt.event.ActionEvent evt) {
745         // TODO add your handling code here:
746 }
747
748     private void etoTextFieldActionPerformed(java.awt.event.ActionEvent evt) {
749         // TODO add your handling code here:
750 }
751
752     private void climateConditionActionPerformed(java.awt.event.ActionEvent evt) {
753         // TODO add your handling code here:
754         if (this.undergroundWaterCensus.isSelected()) {
755             this.coolHumid.setEnabled(false);
756             this.coolDry.setEnabled(false);
757             this.moderateHumid.setEnabled(false);
758             this.moderateDry.setEnabled(false);
759             this.hotHumid.setEnabled(false);
760             this.hotDry.setEnabled(false);
761             this.wcTextField.setEnabled(true);
762
763         } else if (this.climateCondition.isSelected()) {
764             this.coolHumid.setEnabled(true);
765             this.coolDry.setEnabled(true);
766             this.moderateHumid.setEnabled(true);
767             this.moderateDry.setEnabled(true);
768             this.hotHumid.setEnabled(true);
769             this.hotDry.setEnabled(true);
770             this.wcTextField.setEnabled(false);
771             this.wcTextField.setText("");
772             pnTextField.setEnabled(true);
773             roTextField.setEnabled(true);
774             diTextField.setEnabled(true);
775             doTextField.setEnabled(true);
```

```
76          }
77
78  }
79
80      private void calcButtonActionPerformed(java.awt.event.ActionEvent evt) {
81          // TODO add your handling code here:
82              this.esTextField.setText(String.valueOf(calculateEs()));
83              this.irTextField.setText(String.valueOf(calculateIr()));
84              this.egTextField.setText(String.valueOf(calculateEg()));
85              this.ecTextField.setText(String.valueOf(calculateEc()));
86              this.eaTextField.setText(String.valueOf(calculateEa()));
87              this.edTextField.setText(String.valueOf(calculateEd()));
88              messageTextLabel.setForeground(new java.awt.Color(0, 0, 0));
789             this.messageTextLabel.setText("calculation successful");
790
791         /* this.esTextField.setText("");
92          this.irTextField.setText("");
93          this.egTextField.setText("");
94          this.ecTextField.setText("");
95          this.eaTextField.setText("");
796         ////////////////////////////////////////
797         double wc = 0;
798         int climateNo = 0;
799         double ea;
800         double eg;
801         double ir;
802         double es;
803         double wf;
804         double ws;
805         if (this.wfTextField.getText().equals(null) ||
is.wfTextField.getText().equals("")) {
 06             messageTextLabel.setForeground(new java.awt.Color(255, 51, 51));
 07             this.messageTextLabel.setText("Ensure that fields are correctly filled");
 08         }
809         try {
 10             wf = Double.parseDouble(this.wfTextField.getText());
 11             ws = Double.parseDouble(this.wsTextField.getText());
 12
 13             double ec = (wf / ws) * 100;
814             double noOfCrop = Double.parseDouble(this.noOfCropTextField.getText());
815             if (this.undergroundWaterCensus.isSelected()) {
816                 wc = Double.parseDouble(this.wcTextField.getText());
 17
 18             } else if (this.climateCondition.isSelected()) {
 19                 if (this.coolHumid.isSelected()) {
 20                     climateNo = 3;
 21                 }
 22                 if (this.coolDry.isSelected()) {
823                     climateNo = 4;
824                 }
825                 if (this.moderateHumid.isSelected()) {
826                     climateNo = 4;
 27                 }
 28                 if (this.moderateDry.isSelected()) {
 29                     climateNo = 5;
 30                 }
831                 if (this.hotHumid.isSelected()) {
832                     climateNo = 5;
833                 }
 34                 if (this.hotDry.isSelected()) {
 35                     climateNo = 8;
 36                 }
 37                 double Pn = Double.parseDouble(this.pnTextField.getText());
 38                 double Di = Double.parseDouble(this.diTextField.getText());
 39                 double Ro = Double.parseDouble(this.roTextField.getText());
```

```
40                  double Do = Double.parseDouble(this.doTextField.getText());
41                  wc = (Pn + wf + Di) - (Ro + Do + climateNo);
42              }
43
44              ea = (wc / wf) * 100;
45
46              Double percentageGroundCovered =
uble.parseDouble(this.percentageGroundCoveredTextField.getText());
47              Double kc = Double.parseDouble(this.kcTextField.getText());
48              Double eto = Double.parseDouble(this.etoTextField.getText());
49              // Double du = Double.parseDouble(this.duTextField.getText());
50              double Du = 0;
51
52              if (this.sprinkler.isSelected()) {
53                  Du = 0.90;
54              } else if (this.drip.isSelected()) {
55                  Du = 0.94;
56              } else if (this.surface.isSelected()) {
57                  Du = 0.65;
58              }
59
60              double headEndDepth =
uble.parseDouble(this.headEndDepthTextField.getText());
61              double tailEndDepth =
uble.parseDouble(this.tailEndDepthTextField.getText());
62
63              double d = (headEndDepth + tailEndDepth) / 2;
64              double y1 = (headEndDepth - d) / 2;
65              if (y1 < 0) {
66                  y1 = y1 * (-1);
67              }
68              double y2 = (tailEndDepth - d) / 2;
69              if (y2 < 0) {
70                  y2 = y2 * (-1);
71              }
72              double ybar = (y1 + y2) / 2;
73
74              double ed = (1 - (ybar / d)) * 100;
75
76              ir = (kc * eto * noOfCrop * percentageGroundCovered) / Du;
77              eg = 100 * (ir / ws);
78              es = 100 * (wc / ir);
79
80
81              this.esTextField.setText(String.valueOf(es));
82              this.irTextField.setText(String.valueOf(ir));
83              this.egTextField.setText(String.valueOf(eg));
84              this.ecTextField.setText(String.valueOf(ec));
85              this.eaTextField.setText(String.valueOf(ea));
86              this.edTextField.setText(String.valueOf(ed));
87              messageTextLabel.setForeground(new java.awt.Color(0, 0, 0));
88              this.messageTextLabel.setText("calculation successful");
89
90          } catch (NumberFormatException nfe) {
91              messageTextLabel.setForeground(new java.awt.Color(255, 51, 51));
92              this.messageTextLabel.setText("Ensure that fields are correctly filled");
93          }
94           * */
95  }
96
97      private void wfTextFieldActionPerformed(java.awt.event.ActionEvent evt) {
98          // TODO add your handling code here:
99  }
00
01      private void irTextFieldActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        // TODO add your handling code here:
    }

    private void headEndDepthTextFieldActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
}

    private void tailEndDepthTextFieldActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
}

    private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        System.exit(0);
    }

    private void backgroundItemActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        @SuppressWarnings("static-access")
        Color newColor  = chooser.showDialog(this, "pick a Color!", defaultcolor);
        messagePanel.setBackground(newColor);
        messagePanel1.setBackground(newColor);
        messagePanel2.setBackground(newColor);
        messagePanel3.setBackground(newColor);
        messagePanel4.setBackground(newColor);
        setBackground(newColor);

    }

    private void wfTextFieldCaretUpdate(javax.swing.event.CaretEvent evt) {
        // TODO add your handling code here:
       /* Object object = (Object)String.valueOf(wfTextField.getText());
        if( (object instanceof Integer == false) || (object instanceof Double == false)
 (object instanceof Float == false) ){
            messageTextLabel.setText("Enter Only Integers or Decimal");
        }else{
           messageTextLabel.setText("");
        }
        */
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {

            public void run() {
                calcFrame calc = new calcFrame();
                calc.setVisible(true);
                try {
                    UIManager.setLookAndFeel(new WindowsLookAndFeel());
                    SwingUtilities.updateComponentTreeUI(calc);
                } catch (UnsupportedLookAndFeelException ex) {
                    Logger.getLogger(calcFrame.class.getName()).log(Level.SEVERE, null,
;
                }

            }
        });

    }

    // Variables declaration - do not modify
    private javax.swing.JMenuItem backgroundItem;
```

```
    private javax.swing.JButton calcButton;
    private javax.swing.JRadioButton climateCondition;
    private javax.swing.JRadioButton coolDry;
    private javax.swing.JRadioButton coolHumid;
    private javax.swing.JTextField diTextField;
    private javax.swing.JTextField doTextField;
    private javax.swing.JRadioButton drip;
    private javax.swing.JTextField eaTextField;
    private javax.swing.JTextField ecTextField;
    private javax.swing.JTextField edTextField;
    private javax.swing.JTextField egTextField;
    private javax.swing.JRadioButton enclosedChannel;
    private javax.swing.JTextField esTextField;
    private javax.swing.JTextField etoTextField;
    private javax.swing.JTextField headEndDepthTextField;
    private javax.swing.JRadioButton hotDry;
    private javax.swing.JRadioButton hotHumid;
    private javax.swing.JTextField irTextField;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel14;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel16;
    private javax.swing.JLabel jLabel17;
    private javax.swing.JLabel jLabel18;
    private javax.swing.JLabel jLabel19;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel20;
    private javax.swing.JLabel jLabel21;
    private javax.swing.JLabel jLabel22;
    private javax.swing.JLabel jLabel23;
    private javax.swing.JLabel jLabel26;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JMenu jMenu10;
    private javax.swing.JMenu jMenu9;
    private javax.swing.JMenuBar jMenuBar5;
    private javax.swing.JMenuItem jMenuItem1;
    private javax.swing.JTextField kcTextField;
    private javax.swing.ButtonGroup kindOfIrrigation;
    private javax.swing.JLabel messageLabel;
    private javax.swing.JLabel messageLabel1;
    private javax.swing.JLabel messageLabel2;
    private javax.swing.JLabel messageLabel3;
    private javax.swing.JLabel messageLabel4;
    private javax.swing.JPanel messagePanel;
    private javax.swing.JPanel messagePanel1;
    private javax.swing.JPanel messagePanel2;
    private javax.swing.JPanel messagePanel3;
    private javax.swing.JPanel messagePanel4;
    private javax.swing.JLabel messageTextLabel;
    private javax.swing.JRadioButton moderateDry;
    private javax.swing.JRadioButton moderateHumid;
    private javax.swing.JTextField noOfCropTextField;
    private javax.swing.JRadioButton openChannel;
    private javax.swing.JTextField percentageGroundCoveredTextField;
    private javax.swing.JTextField pnTextField;
```

```
    private javax.swing.JTextField roTextField;
    private javax.swing.JRadioButton sprinkler;
    private javax.swing.JRadioButton surface;
    private javax.swing.JTextField tailEndDepthTextField;
    private javax.swing.ButtonGroup typeOfChannel;
    private javax.swing.ButtonGroup typeOfClimate;
    private javax.swing.ButtonGroup undergroundOrClimate;
    private javax.swing.JRadioButton undergroundWaterCensus;
    private javax.swing.JTextField wcTextField;
    private javax.swing.JTextField wfTextField;
    private javax.swing.JTextField wsTextField;
    // End of variables declaration

    private Color defaultcolor = Color.gray;
    private JColorChooser chooser = new JColorChooser(defaultcolor);



    private double getWf() {
        if (wfTextField.getText().trim().isEmpty()) {
            wfTextField.setText("0.00");
        }
        return Double.parseDouble(wfTextField.getText().trim());
    }

    private double getWs() {
        if (wsTextField.getText().trim().isEmpty()) {
            wsTextField.setText("0.00");
        }
        return Double.parseDouble(wsTextField.getText().trim());
    }

    private double getNoOfCropUnit() {
        if (noOfCropTextField.getText().trim().isEmpty()) {
            noOfCropTextField.setText("0.00");
        }
        return Double.parseDouble(noOfCropTextField.getText().trim());
    }

    private double getWc() {

        if (wcTextField.getText().trim().isEmpty()) {
            wcTextField.setText("0.00");
        }else{
            if (this.undergroundWaterCensus.isSelected()) {
                return Double.parseDouble(wcTextField.getText().trim());

            } else if (this.climateCondition.isSelected()) {
                double climateNo = 0;
                if (this.coolHumid.isSelected()) {
                    climateNo = 3;
                }
                if (this.coolDry.isSelected()) {
                    climateNo = 4;
                }
                if (this.moderateHumid.isSelected()) {
                    climateNo = 4;
                }
                if (this.moderateDry.isSelected()) {
                    climateNo = 5;
                }
                if (this.hotHumid.isSelected()) {
                    climateNo = 5;
                }
                if (this.hotDry.isSelected()) {
```

```
095                     climateNo = 8;
096                 }
097
098                 return ( (getPn()+getWf()+getDi())- (getRo()+getDo()+climateNo) );
099             }
100         }
101        return 0;
102     }
103
104     private double getPn() {
105         if (pnTextField.getText().trim().isEmpty()) {
106             pnTextField.setText("0.00");
107         }
108         return Double.parseDouble(pnTextField.getText().trim());
109     }
110
111     private double getRo() {
112         if (roTextField.getText().trim().isEmpty()) {
113             roTextField.setText("0.00");
114         }
115         return Double.parseDouble(roTextField.getText().trim());
116     }
117
118     private double getDi() {
119         if (diTextField.getText().trim().isEmpty()) {
120             diTextField.setText("0.00");
121         }
122         return Double.parseDouble(diTextField.getText().trim());
123     }
124
125     private double getDo() {
126         if (doTextField.getText().trim().isEmpty()) {
127             doTextField.setText("0.00");
128         }
129         return Double.parseDouble(doTextField.getText().trim());
130     }
131
132     private double getGroundCovered() {
133         if (percentageGroundCoveredTextField.getText().trim().isEmpty()) {
134             percentageGroundCoveredTextField.setText("0.00");
135         }
136         return Double.parseDouble(percentageGroundCoveredTextField.getText().trim());
137     }
138
139     private double getKc() {
140         if (kcTextField.getText().trim().isEmpty()) {
141             kcTextField.setText("0.00");
142         }
143         return Double.parseDouble(kcTextField.getText().trim());
144     }
145
146     private double getETo() {
147         if (etoTextField.getText().trim().isEmpty()) {
148             etoTextField.setText("0.00");
149         }
150         return Double.parseDouble(etoTextField.getText().trim());
151     }
152
153     private double getdh() {
154         if (headEndDepthTextField.getText().trim().isEmpty()) {
155             headEndDepthTextField.setText("0.00");
156         }
157         return Double.parseDouble(headEndDepthTextField.getText().trim());
158     }
159
```

```
60      private double getdt() {
61          if (tailEndDepthTextField.getText().trim().isEmpty()) {
62              tailEndDepthTextField.setText("0.00");
63          }
64          return Double.parseDouble(tailEndDepthTextField.getText().trim());
65      }
66
67      private double calculateEc(){
68        if(openChannel.isSelected()){
69            return (getWf()/getWs())*100;
70        }else if(enclosedChannel.isSelected()){
71           return (getWf()/getWs())*100;
72        }
73        return 0;
74      }
75
76      private double calculateIr(){
77          double Du = 0;
78          if (this.sprinkler.isSelected()) {
79                  Du = 0.90;
80              } else if (this.drip.isSelected()) {
81                  Du = 0.94;
82              } else if (this.surface.isSelected()) {
83                  Du = 0.65;
84              }
85          return (getKc()*getETo()*getNoOfCropUnit()*getGroundCovered())/Du;
86      }
87
88      private double calculateEa(){
89          return (getWc()/getWf())*100;
90      }
91
92      private double calculateEs(){
93          return (getWc()/calculateIr())*100;
94      }
95
96      private double calculateEg(){
97          return (calculateIr()/getWs())*100;
98      }
99
00      private double calculateEd(){
01              double d = (getdh() + getdt()) / 2;
02              double y1 = (getdh() - d) / 2;
03              if (y1 < 0) {
04                  y1 = y1 * (-1);
05              }
06              double y2 = (getdt() - d) / 2;
07              if (y2 < 0) {
08                  y2 = y2 * (-1);
09              }
10              double ybar = (y1 + y2) / 2;
11              double ed = (1 - (ybar / d)) * 100;
12
13          return ed;
14      }
15  }
16
17
```