REVISED EDITION

Introduction to Systems Development

NQF Level 2

R. Jonker, M. Smit

STUDENT'S BOOK

FET FIRST

FET FIRST

INTRODUCTION TO SYSTEMS DEVELOPMENT

Revised Edition

NQF Level 2

Student's Book

with CD

R. Jonker, M. Smit

FET FIRST Introduction to Systems Development Revised Edition
NQF Level 2 Student's Book with CD

Second edition, first impression 2012

First published 2007 by
Troupant Publishers (Pty) Ltd
Suite 10 Private Bag X12
Cresta
2118

Authors: R. Jonker and M.W.H. Smit
Copy-editor: Jeannie van den Heever
Proofreader: Michael Sangster
Cover design by Brand Talk, a subsidiary of African Access
Typeset by Golden Pear Desktop Publishing

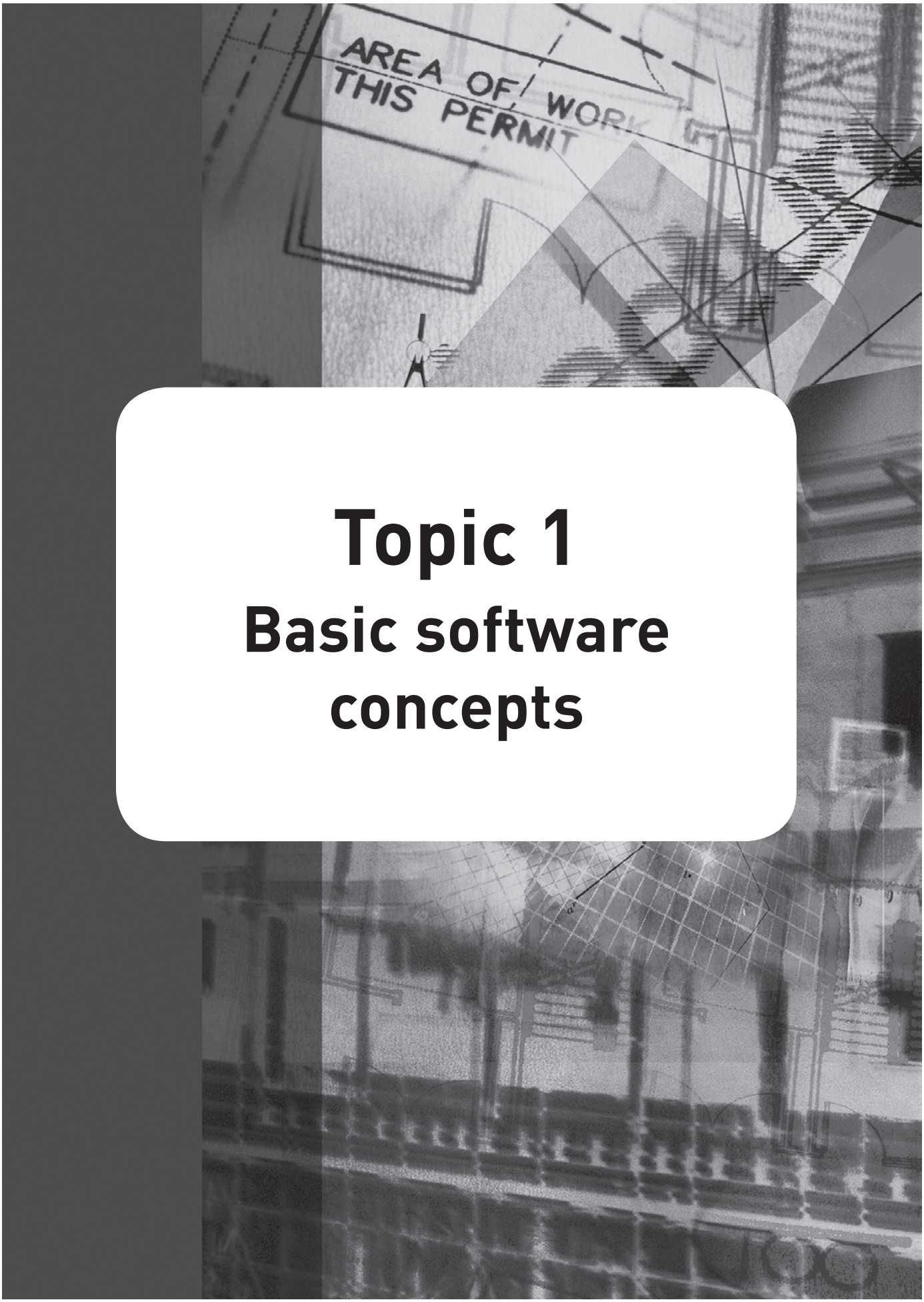
Distributed by Macmillan South Africa (Pty) Ltd

ISBN: 978-1-920334-87-1

To order any of these books contact Macmillan Customer Services at:

Tel: (011) 731 3337
Fax: (011) 731 3535
e-mail: skhosanag@macmillan.co.za

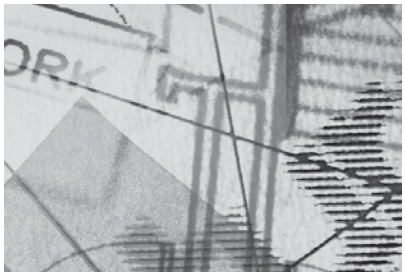
Contents



Topic 1
Basic software concepts

Module 1

Types of software



Overview

At the end of this module, you should be able to:

- explain the term 'software'
- distinguish between various types of software and their purposes (applications and systems software)
- distinguish between open-source and proprietary software
- explain why the same software has different versions.

Unit 1.1 What is software?

Computer **software,** or simply software, consists of programs and other operating information used by a computer. Instructions and data are stored electronically in programs and enable your computer to carry out specific actions. The physical components of the system, such as the hard drive, printer, monitor, keyboard and mouse, are called **hardware**.

Software consists of:

- machine-readable code that enables the computer to perform a specific task or instruction
- all documentation that forms a vital component of each project, such as the specifications, the design, legal and accounting documents, as well as all the user manuals.

Software is generally written in a **high-level language** that is easier and more efficient for the programmer to use. A high-level language is a programming language that has instructions resembling a human language, such as English. A **low-level language** is a programming language that is close to machine code in form. The term 'high-level language' does not imply that the language is superior to low-level programming languages. In fact, in terms of depth of knowledge of how computers operate, the opposite may be true.

High-level programming languages are more abstract, easier to use and more portable across platforms than low-level programming languages. In general, high-level languages make complex programming simpler, while low-level languages tend to produce more efficient code.

With a high-level language, complex elements can be broken up into simpler, although still fairly complex elements. This allows programmers to avoid having to continually recreate the same programming code. However, the cost of this convenience is often less efficient code overall. For this reason, code that needs to run particularly quickly and efficiently may be written in a low-level language, even if a high-level language would make the coding easier.

A high-level programming language **translates** (converts) the program into a low-level machine language that consists of **binary values** (0, 1).

Did you know?

John W. Tukey first used the term 'software' in 1957 to describe the concept of reading different sequences of instructions into the memory of a device to control computations.

Words & Terms

software: programs and other operating information used by a computer

hardware: the physical components of a computer, such as the hard drive, printer, monitor, keyboard and mouse

high-level language: a programming language that has instructions resembling a human language, such as English

low-level language: a language that is close to machine code in form

translate: to convert from one computer language into another

binary values: having two values or states; this describes a number system that has a base of two and uses 1 and 0 as its digits

These binary values are then loaded into the computer's **random access memory** (RAM) before being sent to the **central processing unit** (CPU) to perform the specific instruction.

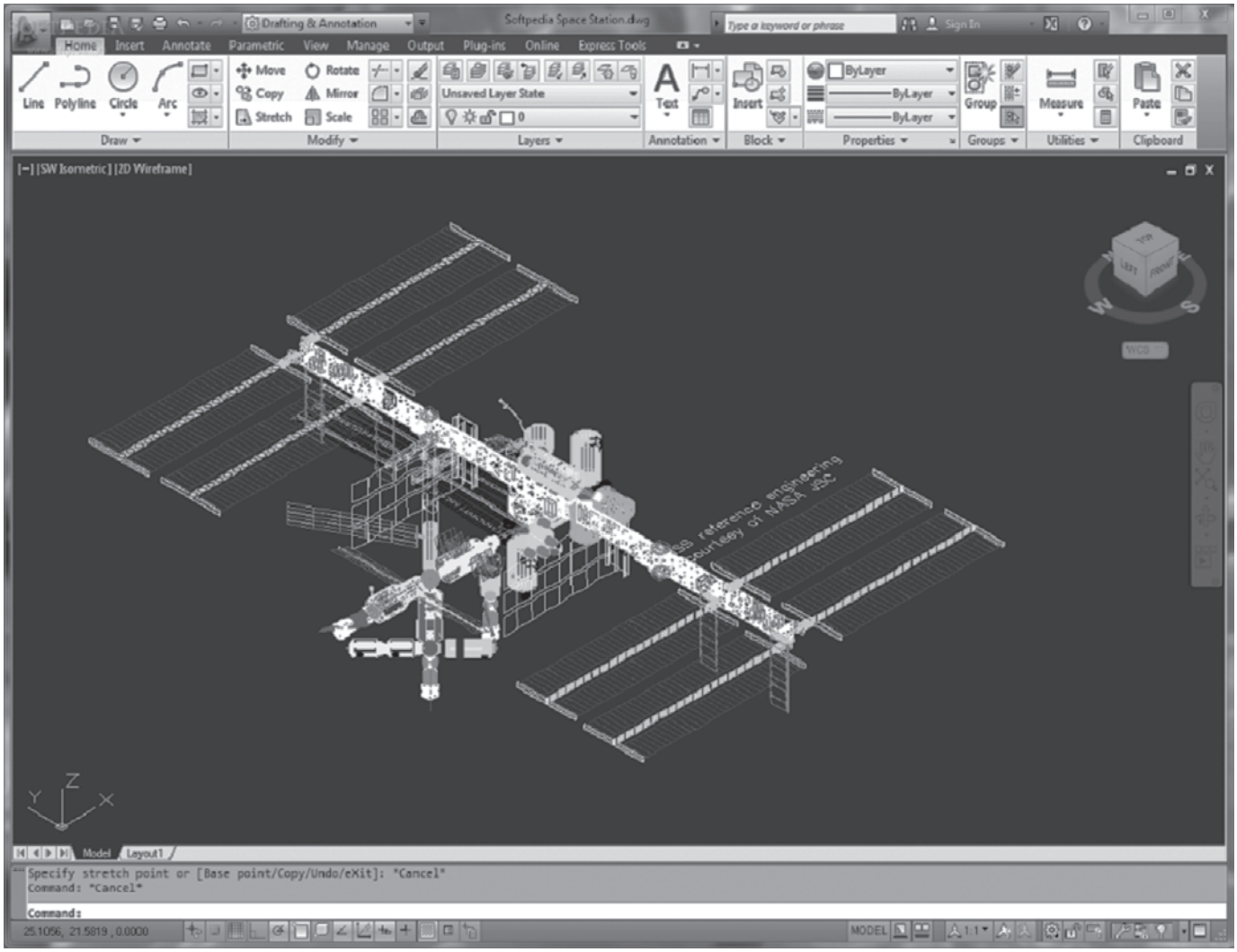


Figure 1.1 An example of software in use – AutoCAD 2011©

Unit 1.2 Different types of software and their functions

Software can be divided into subclasses depending on the use and purpose of the program. Some software programs perform tasks that the user wishes them to perform, for example word processing, while other programs (such as antivirus software) perform tasks on different programs to achieve a desired result.

In the recent past, sales of all types of software have increased, providing the user with a wide variety of both general software and software designed for specific tasks. These include the following:

- Software tools are either basic single programs created to perform one task, such as word processing, or advanced software tools, such as a software bundle.

Words & Terms

random access memory (RAM): computer memory used by programs to perform necessary tasks while the computer is on; it holds the program code and data during computation

central processing unit (CPU): interprets instructions and processes data contained in computer programs

- Application software (also called end-user programs) performs a specific task for the end user. Application software can be divided into numerous categories, for example media players, games, spreadsheets and word processors.
- System software refers to programs that manage a computer's resources, scheduling and the monitoring of computer events. Operating systems are a form of system software.
- Utility software is designed to help manage an operating system. It performs a single task or a range of small tasks to fine-tune the operating system and hardware.
- Language translators translate a high-level language into a low-level language (assembly language or machine language).

Unit 1.3 Software tools

Introduction

Most people would agree that the software market has changed dramatically in the last few years. In the past, users would buy what is called **proprietary software** from a **software vendor**. The software vendor would then bring out updated versions to replace older versions on a regular basis. If users wanted the updated version, they had to buy it and the software vendor thus made more money. This system provided some competition among the software companies, which helped to improve the quality of the software.

Today, however, users may also obtain free software, called **open-source software**. Users pay only for maintenance of the software. In this way, users have the advantage of having the latest version of the software for free, while the software company still makes money by charging for the maintenance of their software.

Open-source software is becoming more popular and there is now a wider range available. This software has become more reliable and **compatible** with other software applications. It has also become more easily available as a result of the internet. Open-source software has a big following on the internet where users worldwide use online forums to improve it.

The success of open-source software has proved to be a challenge for software vendors. The software products produced by these companies now have to be of an even higher standard, as they are not only competing against other software vendors, but against a global software community looking for alternative options to proprietary software.

Good examples of open-source versus proprietary software are Linux versus Microsoft Windows and Mozilla Firefox versus Internet Explorer.

Words & Terms

proprietary software: software programs that users must buy

software vendor: a software company that develops and sell proprietary software

open-source software: software programs that users can obtain for free

software compatibility: compatible software can interact with different versions of the same software, as well as with other software products

Proprietary software

Software vendors develop proprietary software for specific tasks. The source code is hidden from users who are not allowed to make any changes to it. If changes are needed, the software vendor will make the

necessary changes. Proprietary software comes with obligatory, reliable and professional technical support that is normally provided via online chat rooms, forums or call centres staffed by software experts. Software vendors provide regular updates and bug fixes, and are constantly improving their software products. Microsoft Excel is an example of proprietary software (see Figure 1.2).

Advantages of proprietary software include the following:

- Customer support is obligatory – vendors help users with any software problems by providing reliable and professional support.
- There are regular updates to the software.

Disadvantages of proprietary software include the following:

- Closed standards may hinder compatibility with other software or between different operating systems.
- It is expensive to buy the licence for the software.
- The software may not be changed or customised without the vendor's permission.

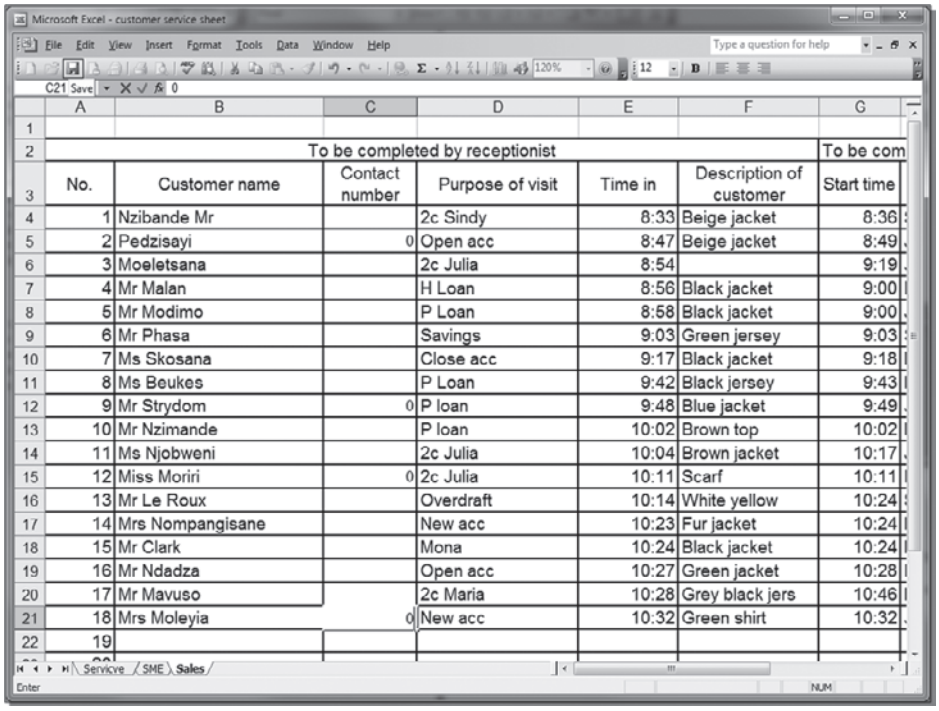


Figure 1.2 Microsoft Excel 2007 © Spreadsheet

Open-source software

Open-source software, as implied by the name, gives users access to the software's source code. This allows users to see the source code and change the code to suit themselves. These software programs are therefore flexible and can be customised to any specification using current programming standards that ensure compatibility across platforms and other software applications. Figures 1.3 and 1.4 show examples of open-source software.

Advantages of open-source software include the following:

- Open standards allow the software to be more compatible with other software or between operating systems.
- Users can download open-source software for free.
- Users can also customise the software to suit their purposes.

Disadvantages of open-source software include the following:

- There is no obligatory support, but users can find online support forums where other users may be able to help them solve any problems.
- Updates may be erratic.

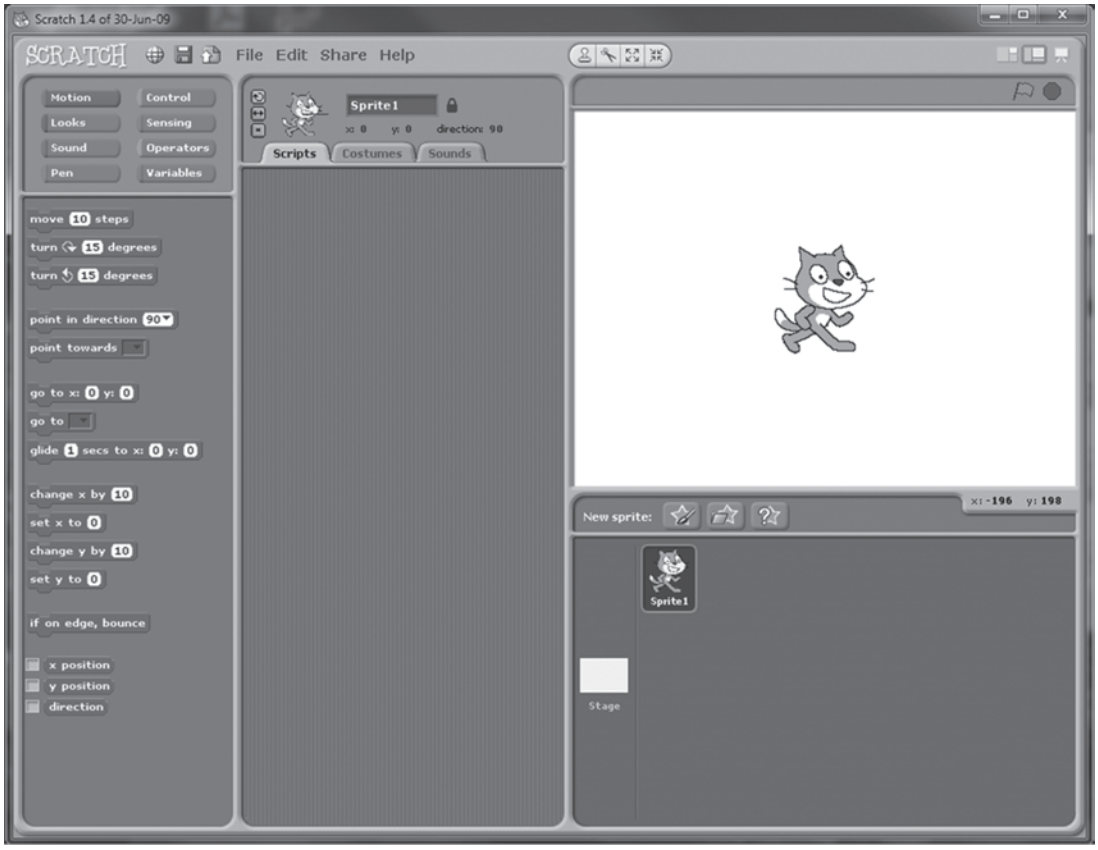


Figure 1.3 Open-source software development tool – Scratch©

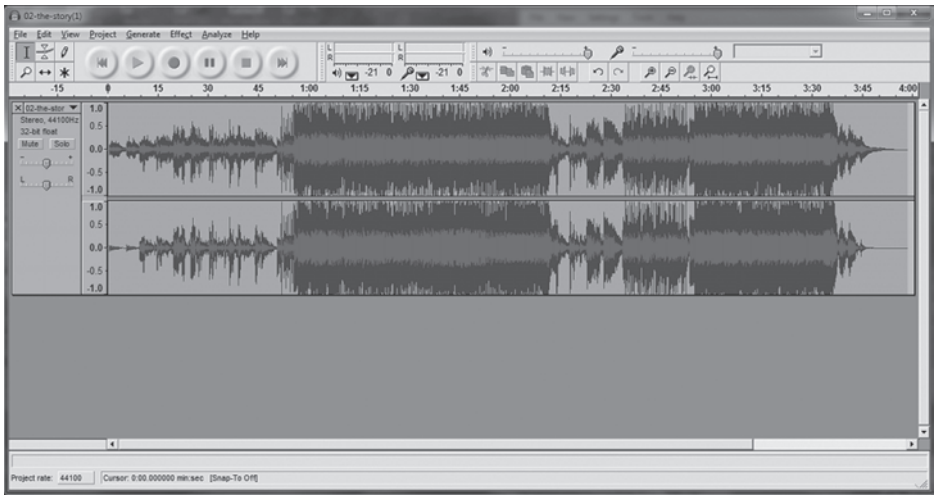


Figure 1.4 Open-source sound editing program©

Assessment activity 1.1

Discussion. Work in groups of at least four.

1. Discuss the differences between open-source and proprietary software-using the following as guidelines:
   a) Open-source versus closed-source programming code
   b) Customer support
   c) Security issues with regard to open-source and closed-source code.
2. Use the internet to make a list of proprietary software programs and their open-source equivalents.
3. Divide your group in two. Download two of the open-source programs you named in question 2. Make sure that you have the propriety equivalents of these programs. Now list the advantages and disadvantages of the programs. One group can evaluate the open-source programs, while the other group evaluates the proprietary software. Try to be objective and do not just look at the price of the software.

Unit 1.4 Software versions

Before we go into detail regarding different types of software, let us first examine the use of **versions** in software. A version of something is a little different from others of the same type (a new model of a car is a version of the same car).

Few software programs are perfect from the start. Software vendors therefore have to update and improve their software programs continually. This results in different versions of the same software program. The process followed is described below:

Reasons for version changes:

- **Pre-release stage:** When software developers develop a software program, they give each new phase of the project a new version number. They also document the changes made to the version number to keep track of the modifications made during the development.
- **Testing phase:** During the testing phase, the version of the program is referred to as the **beta version**. A beta version of software is supposed to be very close to the final product, but, in practice, it is a way of having users to test the software under real conditions.
- **Post-release phase:** After testing, an **alpha version** of the software program is released with a new version number. When problems or bugs appear, the software developer makes the necessary modifications and then labels the software with a new version number. This process goes on until the end of the software program's life cycle.

Software developers therefore use version numbers to:

- identify different phases in the development of a program and the modifications made during each phase
- identify and document modifications made to a program once it has been released.

Words & Terms

version: a new version of a software program is an updated form of the same program; differences can be major or minor

beta version: a pre-release version of software that has gone through some tests, but is not yet ready to be sold

alpha version: software that is ready for the market

In the workplace

The speed with which operating systems and applications software have been developed and keep on changing has been mind-boggling. However, the changes are often not as far-reaching as the companies who market them would have us believe. For instance, a word processing program you use today will carry out much the same functions as the program you used 15 years ago, although there will have been many 'improved' versions since then.

However, users cannot afford to ignore the changes. If a business does not update its software regularly, it will find that, after a few years, the software is completely out of date and people in other companies can no longer read their documents.

When software developers give different version numbers to the software to keep track of all the changes made to programs, they use different ways of numbering their developments. Here is a list of some version schemes that developers use:

- **Year of release:** Use of the year in which the software was released, for example Macromedia Dreamweaver 2004
- **Alphanumeric codes:** Use of alphabetic and/or numeric values, for example Adobe Photoshop CS2
- **Roman numerals:** Use of roman numerals, for example Apple's Mac OS X
- **Numeric:** Use of numeric values, for example Winamp 5.0.8.

The most popular scheme, and one we will discuss in more detail here, is the numeric version scheme. The advantage of a numeric scheme is that it is flexible and it gives information about the software's history. The development in a numeric version scheme will typically be as follows:

- A software program must first be tested before it can be released. The first version will be 0.1.0, also known as the beta version. As testing progresses, so the versioning continues up to the point of release.
- When the software is released, the version becomes 1.0.0. This is the first alpha version. When a few bugs are fixed during maintenance work, the version becomes 1.0.1. When a minor improvement is then made to the version, it becomes 1.1.1. Finally, when a major update is done, it becomes version 2.0.0.

As the developers continue to improve a software program with minor or major updates, the version will keep us up-to-date on how it developed over time. An easy way to remember numeric versions is by remembering this simple equation: *Major.Minor.Maintenance.*

Did you know?

Microsoft has used different version schemes to denote the releases of their operating system software, Windows. The first Windows had a numeric scheme – Windows 1.0 to Windows 3.11. Later, this changed to Windows 95, Windows 98 and Windows 2000, using the year of release scheme. Finally, it changed to the alphanumeric scheme – Windows ME and Windows XP.

Assessment activity 1.2

Work on your own.

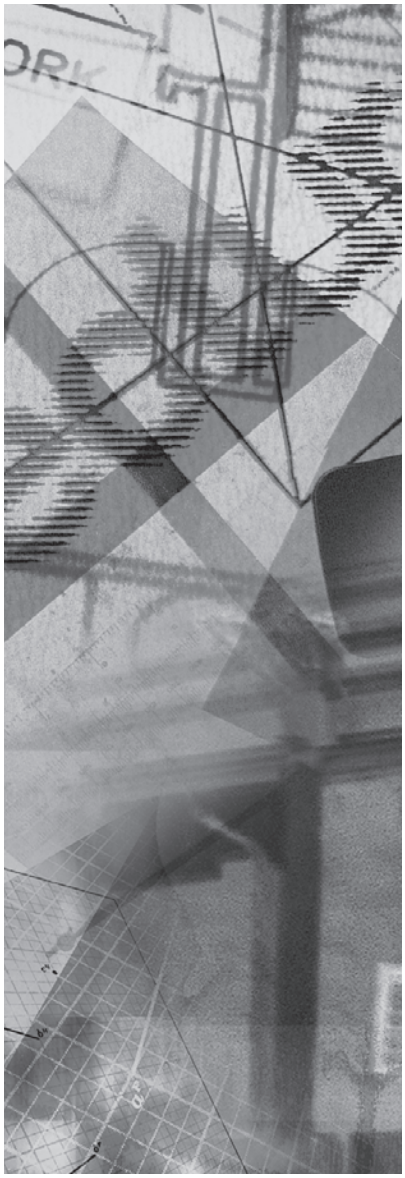
1. a) What is computer software? (3)
   b) What is computer hardware? (2)
2. Explain the difference between high-level and low-level programming languages. (3)
3. Name three types of software and briefly explain what each does. (6)
4. a) Explain the differences between open-source software and proprietary software. (4)
   b) List the advantages and disadvantages of each of these two different types of software. (10)
5. Create your own software program with a unique name and complete the following:
   a) Name four beta versions.
   b) Name four maintenance releases.
   c) Name four minor releases.
   d) Name three major releases using the numeric versioning scheme. (15)
6. Explain in one paragraph why software developers use version schemes for software. (4)
7. List four version schemes that developers use and explain them each in one sentence. (4)
8. Name two advantages and two disadvantages of using versioning schemes. (4)

Five marks for the following:
Neatness: whether written by hand or typed, the print should be neat
Creativity: any use of colour, pictures and individual thought
Elements of planning: due date met, headings in a different style or colour, logical order of answers, relationship of parts to each other.

Total marks: 60

Module 2

Application software



Overview

At the end of this module, you should be able to:

- identify and demonstrate the different features common to all types of application software
- list and describe different types of application software and their use
- explain the purpose and use of each of the types of features common to all types of application software
- outline the processes for installing application software.

Unit 2.1 Features common to all types of application software

What is application software?

Application software, also called end-user programs, performs a specific task for the end user. By way of contrast, **system software** is computer software designed to operate the computer hardware and to provide a platform for running application software. There are many types of application software, such as media players, games, spreadsheets and word processors. Each type of application software performs a certain task for the user, whether for recreational use or for use in business applications.

Multiple applications bundled together is referred to as an **application suite**. Microsoft Office is a good example as it includes a word processing program, MS Word, a spreadsheet program, MS Excel and other programs that are bundled together into one application suite. Each program is used in a similar way, making it easier to learn and use the application software. An advantage of a suite is that the programs in the suite can interact with each other, for example a spreadsheet from MS Excel can be embedded in an MS Word document. Application software cannot run without an operating system (discussed in Module 4). Figure 2.1 shows an example of application software (Corel PaintShop photo Pro X3).

Words & Terms

application software: performs a specific task for the end user; also called an end-user program

system software: operates the computer hardware and provides a platform for application software

application suite: multiple applications bundled together

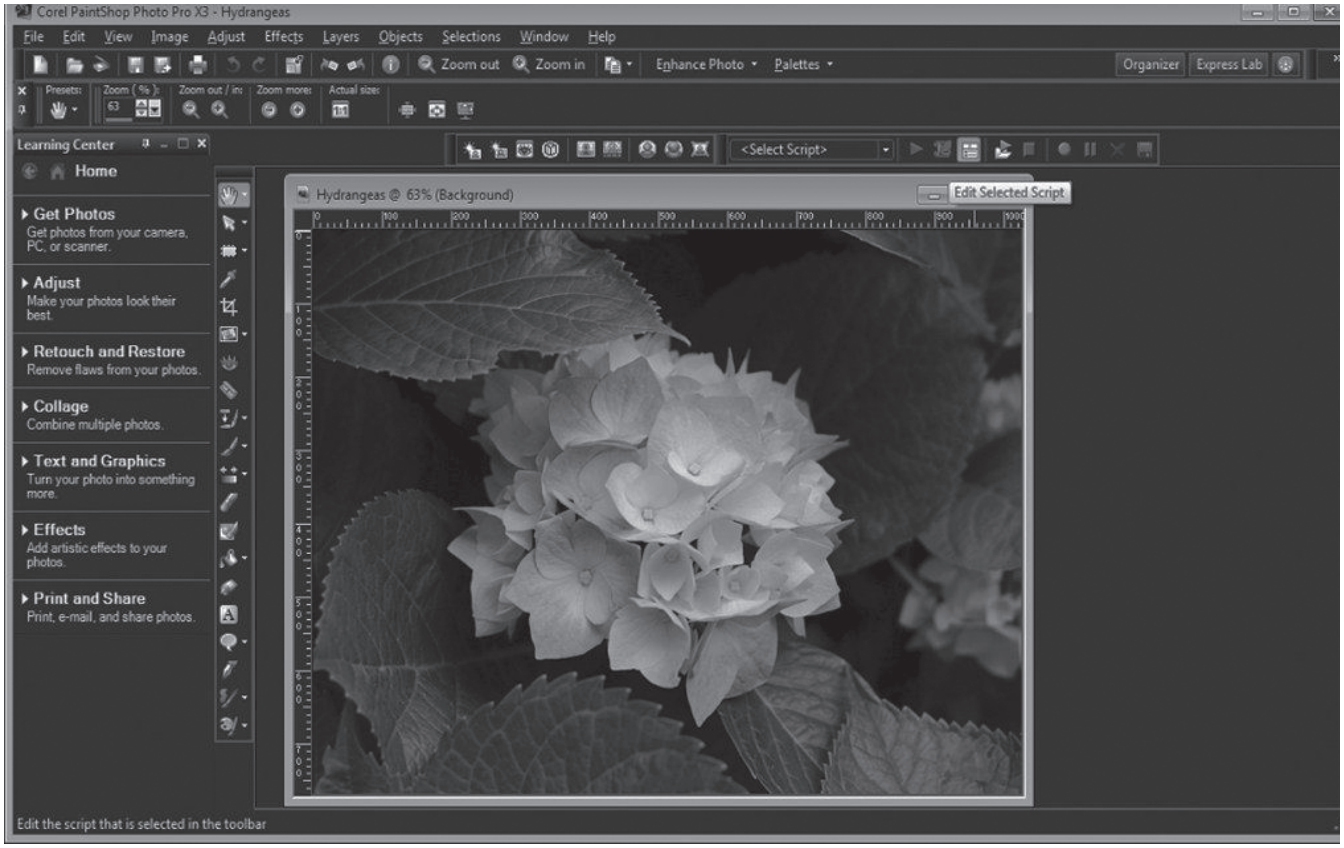


Figure 2.1 Example of application software

Features of application software

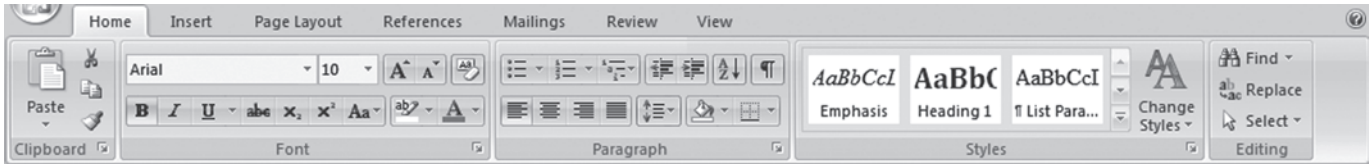
Figure 2.2 shows the toolbars of four programs from the Microsoft Office 2007 suite. Just by looking at them, the user can see that they share common features. A **feature** is a distinctive attribute or aspect of a program. These programs were developed by the same company and are designed to look the same. This not only makes it easier for users to learn a new program, but it is also easier to take a document from one program and use it in another program.

Other software development companies also use the same style of menus as shown in Figure 2.2. The reason for this is that these companies know that most of their users probably have one of these programs. By using a menu with which users are familiar, they make it easier for new users to learn their own programs. Section 2.3 explores features common to application software in more detail.

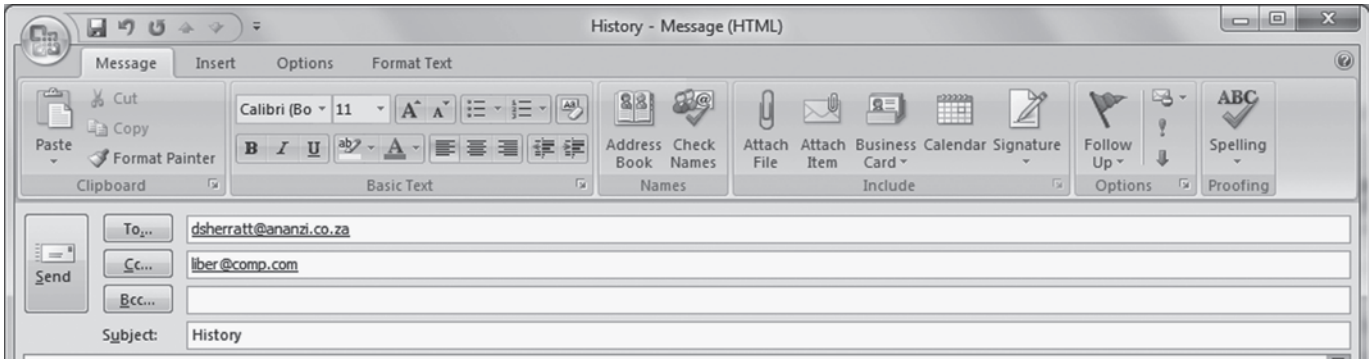
Words & Terms

feature: a distinctive attribute or aspect of a program

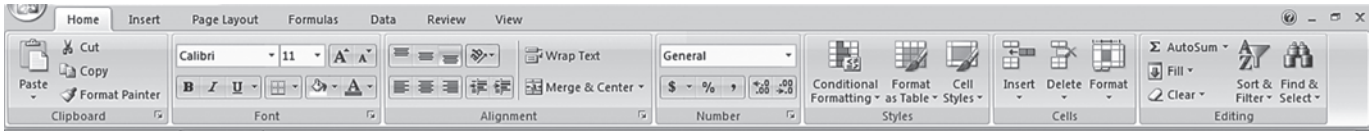
MS WORD 2007



OUTLOOK 2007



EXCEL 2007



POWERPOINT 2007

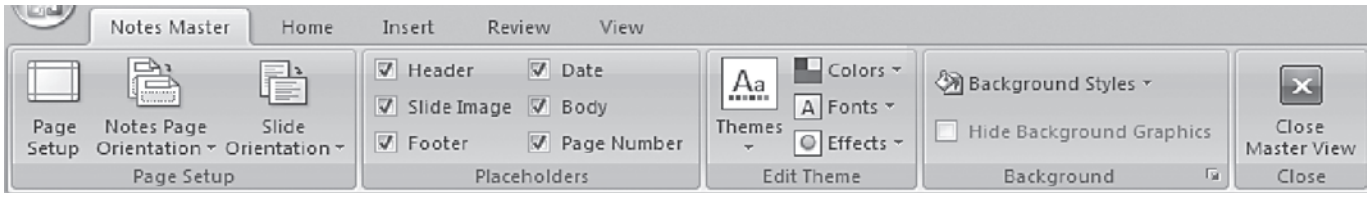


Figure 2.2 Menus of programs in the Microsoft Office 2007 suite

Unit 2.2 Different types of application software and their use

Word processing software

Word processing software enables users to create, edit and store a document electronically. Users can save their documents on a computer hard drive, a CD or DVD, or a flash drive. The advantages of a word processing program are that you can edit your work, delete mistakes, move sections of text around, and insert new words and sentences. When you have completed your document, you can save it for future reference, print it on a printer to obtain a hard copy or send it to a recipient via email.

Word processing programs, or word processors, vary considerably, but they all support the following basic features:

- **Insert text:** Allows you to insert text anywhere in the document.
- **Delete text:** Allows you to erase characters, words, lines or pages by pressing the backspace or delete keys.

- **Cut and Paste:** Allows you to remove or cut a section of text from one place in a document and insert or paste it somewhere else.
- **Copy:** Allows you to duplicate a section of text or copy a section of text from another document.
- **Page size and margins:** Allows you to define various page sizes and margins. The program will automatically readjust the text so that it fits onto the page if your margins are too big or small. This helps when working with different sizes of paper.
- **Search and replace:** Allows you to direct the word processor to search for a particular word or phrase. You can also replace one group of characters with another wherever that the first group appears.
- **Word wrap:** The word processor automatically moves to the next line when you have filled one line with text, and it will readjust text if you change the margins.
- **Print:** Allows you to send a document to a printer to obtain a hard copy.

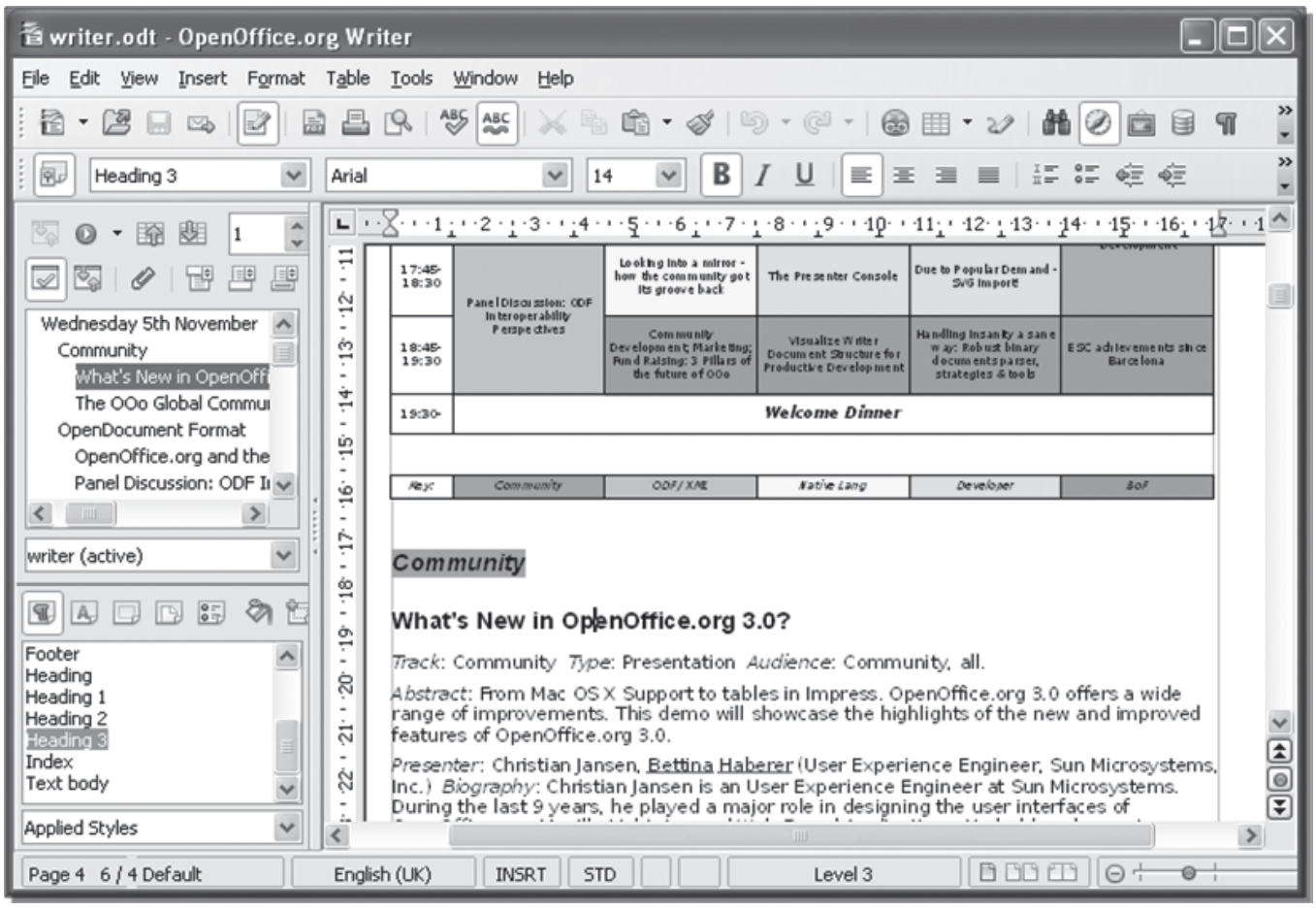


Figure 2.3 A word processing application – OpenOffice.org

Word processors that support only these basic features listed above are called text editors. Most word processors, however, support additional features that enable you to manipulate and format documents in more sophisticated ways. These more advanced word processors are sometimes called full-feature word processors. Full-feature word processors usually support the following added features:

- **File management:** Word processors have file management capabilities that allow you to create, delete, move and search for files.
- **Font specification:** Allows you to change fonts in a document. For example, you can specify bold, italics and underlining. Most word processors also let you change the font size and the typeface.
- **Footnotes and cross-references:** Automates the numbering and placement of footnotes and enables you to cross-reference other sections of the document.
- **Graphics:** Allows you to embed illustrations and graphs into a document. Some word processors let you create the illustrations within the word processor; others let you insert an illustration produced by a different program.
- **Headers, footers and page numbering:** Allows you to specify customised headers and footers that the word processor will put at the top and bottom of every page. The word processor automatically keeps track of page numbers so that the correct number appears on each page.
- **Layout:** Allows you to specify different margins in a single document and to specify various methods for indenting paragraphs.
- **Merges:** Allows you to merge text from one file into another file. This is particularly useful for generating many files that have the same format but different data. Generating mailing labels is a classic example of using the merge feature.
- **Spell checker:** This allows you to check the spelling of words. It will highlight any words that it does not recognise.
- **Windows:** This feature allows you to edit two or more documents at the same time. Each document appears in a separate window. This is particularly valuable when working on a large project that consists of several different files.
- **WYSIWYG (what you see is what you get):** With WYSIWYG, a document appears on the screen exactly as it will look when printed.

Different types of word processors include WordPad and MS Word. An open-source option is Open Office's word processor.

Words & Terms

alphanumeric: using both letters and numbers: A-Z, a-z, 0–9

Spreadsheet software

Spreadsheet application software enables the user to create a worksheet in a two-dimensional matrix or grid containing rows and columns. Each cell in the grid can contain **alphanumeric** text, numeric values or formulas. The formula defines how the content or value in each cell can be calculated or defined. For example, you can insert values in two cells and use a formula to multiply the two values and insert the answer in a third cell.

Most spreadsheets also allow for the creation of graphs and charts from the values in the cells. As the values in the cells change, they will automatically change the graph or chart as well. Spreadsheets are mostly used for accounting purposes because of their ability to recalculate the entire sheet automatically after changing one or more values. They are also used for a wide range of other applications where calculations are required.

Most spreadsheet applications also allow users to link data from different spreadsheets. For example, you can link all the monthly income/expenditure spreadsheets for a 12-month period to create one spreadsheet containing all the totals. You can then use graphs to give an overview of the year's income and expenditure.

As with word processing software, you can also save spreadsheets electronically to a hard drive, CD or DVD, or a flash drive. You can print and email spreadsheets. Many of the features listed above for word processing software are also applicable to spreadsheets.

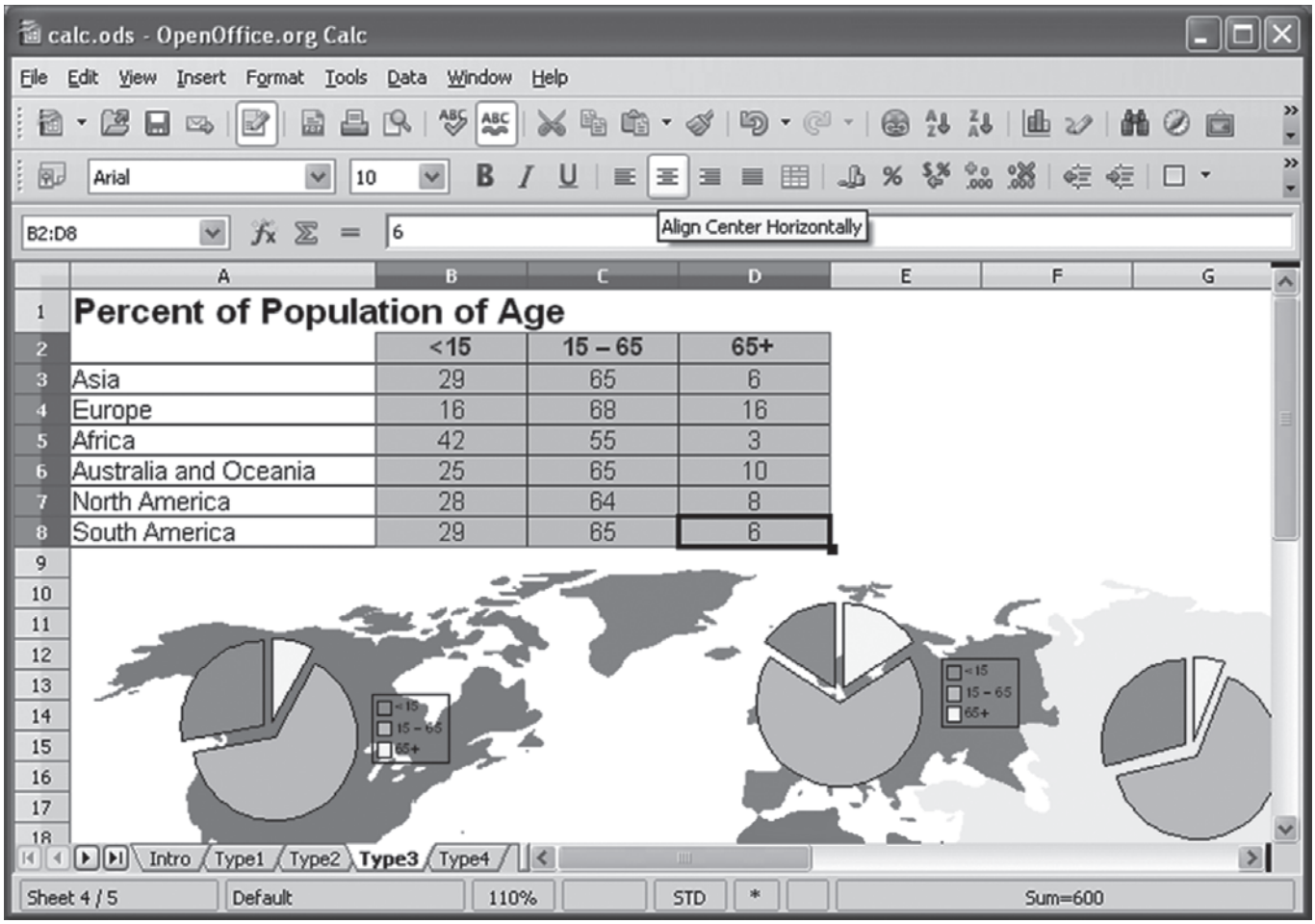


Figure 2.4 A spreadsheet application – OpenOffice.org

Different types of spreadsheets include MS Excel and Lotus 123. OpenOffice Calc is an example of an open-source application.

Presentation software

Presentation application software is used to create and display information in a slideshow format. The program can create slides combining text, images, graphs and audio/video clips. The program can then manipulate the transition between slides to create a slideshow (see Figure 2.5).

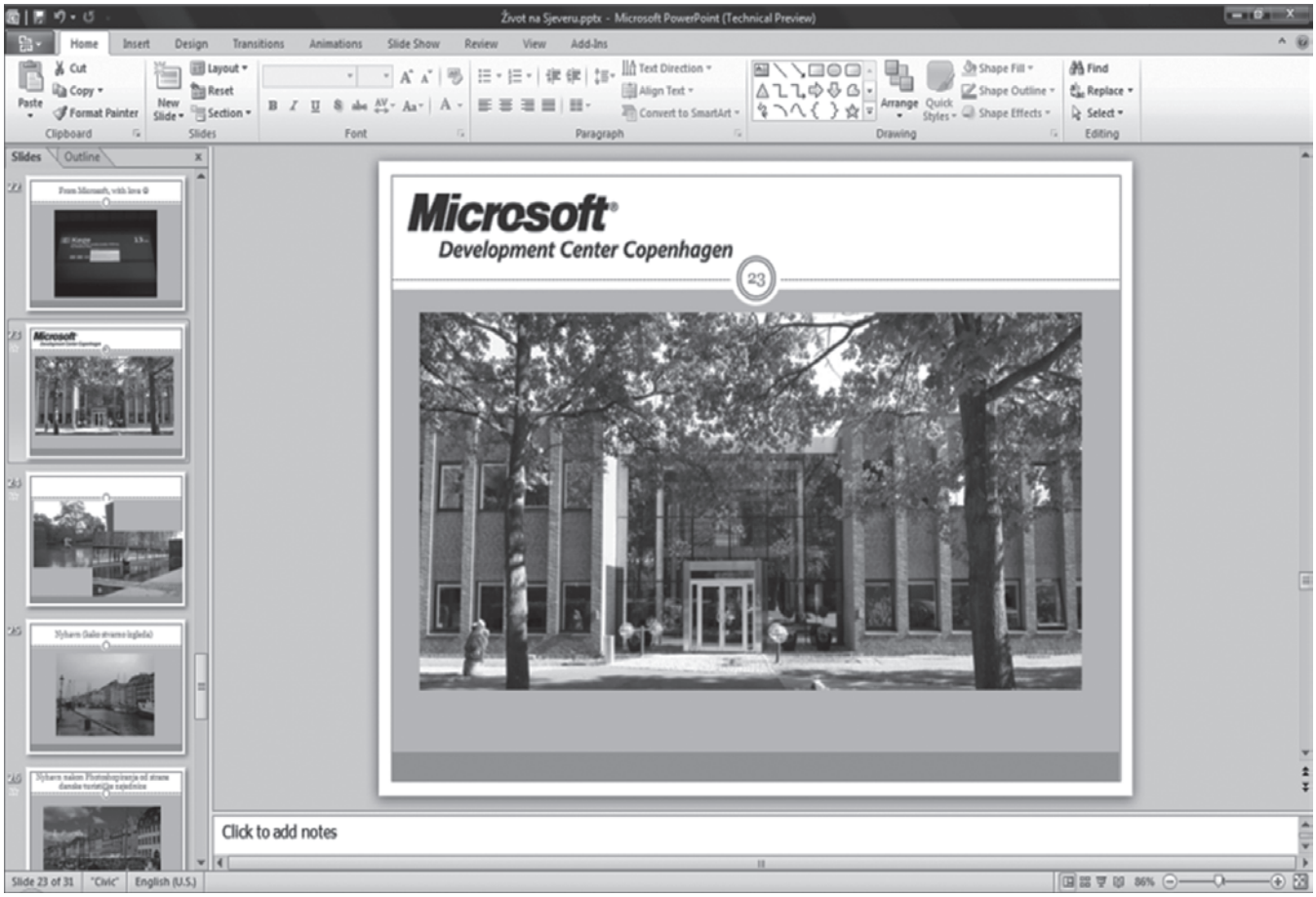


Figure 2.5 Microsoft PowerPoint 2010©

People use presentation software to help with visual aspects when delivering a presentation, lecture or speech. Business people use it to show graphs and statistics. Today, the most common way of displaying presentations is by means of a video projector. An example of presentation software is MS PowerPoint. An open-source version is Open Office Impress.

Database software

Database application software, often abbreviated to DB, is an application designed to store, organise and manage large amounts of data easily. Traditional databases are organised by fields, records and files. A **field** is a single piece of information, for example your student number. A **record** is a collection of fields. For example, the details on your college registration form, such as your name, surname and ID number, would be separate fields. Together, these fields form a record of your personal details. A **file** is a collection of records. We can describe it as a book containing all the details of all the students.

To manage or access the information in a database, you need a program that enables you to enter data, organise the data and retrieve information from the database easily and effectively. To do this, a database needs a database management system (DBMS). The DBMS

Words & Terms

field: a single piece of information

record: a collection of fields

file: a collection of records

can retrieve information from the database as required by the user through queries or reports. This allows the user to retrieve only that information required to perform a certain task. For example, if a student wants to find out about only his/her final results from the previous semester, the student counsellor just has to run a query for that specific information and not the student's entire academic record. Examples of database software include MS Access, Oracle and SQL (see Figure 2.6).

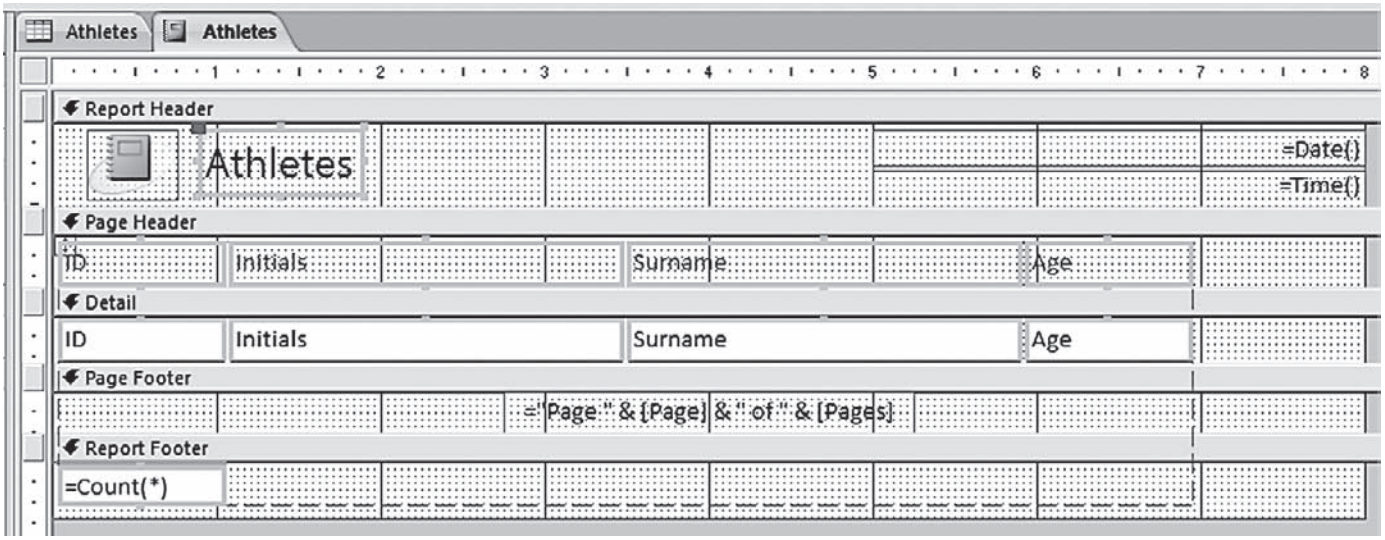


Figure 2.6 A database application – Microsoft Office 2007 Access©

Multimedia software

The term 'media' traditionally refers to forms of printed material. Multimedia, however, is a combination of text, audio, video, animation and still images. If you take a piece of text and add music and a still image next to the text, you have created a basic multimedia file. Multimedia applications allow you to create, edit, record and play various forms of multimedia depending on which multimedia application you use.

Different types of multimedia applications are MS Media Player, Real Video, Win-Amp, Adobe Flash and Apple iTunes.

Think about it

Gone are the days when web pages just had text. Today, most web pages contain various forms of multimedia. These include audio files, video clips, animations and **Adobe Flash**, commonly known as Flash. Some Flash animations allow for interaction – this is called interactive media.

Words & Terms

Adobe Flash: multimedia software that adds animation, video and interactivity to web pages.

Assessment activity 2.1

Work on your own.

1. Define application software and an application suite in your own words. (4)
2. Give two examples of where you would use a word processing application in your daily life. (4)
3. Pastel/Lotus 123/AutoCad/Norton AntiVirus/ MediaPlayer/Visual Studio 2005/Power Point
   From the list above, choose the program you would use to do each of the following. You could possibly use one program for more than application:
   a) Draw a plan of your dream home.
   b) Plan the budget for building your dream home.
   c) Protect your computer from viruses.
   d) Listen to your music on the computer.
   e) Create a new program for your company.
   f) Create a materials list with prices for your dream home.
   g) Give a presentation of a new venture at your company. (7)

Total marks: 15

Unit 2.3 Purpose and use of features common to types of application software

There are many different types of application software available for the end user, but all share some common features. In this section, we shall discuss some of the more general features common to all application software.

In the workplace

In practice, most people in business do not create their own programs, but work with Microsoft Word for word-processing purposes (for example when writing a document), and with Microsoft Excel for spreadsheet purposes (for example when drawing up a budget). Other applications that are widely used are Microsoft Outlook for emailing and Microsoft Explorer for internet browsing. Microsoft PowerPoint, a presentation program, is also widely used. For anything beyond these standard applications, a business must either invest in more expensive applications or commission a software development company to write a customised program.

The fundamental reasons why various software applications share the same features are:

- usability
- functionality
- productivity
- compatibility.

Usability

Usability refers to how user-friendly or easy a program is to use. When end users buy software, they want a program that is easy to use with a simple interface to perform the tasks they want to do. Usability thus denotes the ease with which people can perform a certain task by using a particular tool.

Words & Terms

usability: the ease with which people can perform a certain task by using a particular tool

functionality: the set of product features and functions of a software application and how well the product performs its functions

graphical user interface (GUI): a type of user interface that allows users to interact with the computer with images rather than text commands

Functionality

Functionality is the set of product features and functions of a software application and its functions. However, functionality does not stop at what the software application is capable of doing. If a product has all the required capabilities, but takes hours to perform a simple instruction, is it still functional? If the user wants to perform a task, but is not sure how to do so, is there support to help the user? Software functionality is more than just the application's ability to perform a given set of instructions; it also refers to how easy it is to perform that given task.

How does one improve functionality in an application? The **graphical user interface (GUI)** is a type of user interface that allows users to interact with the computer with images rather than text commands. A GUI uses images, icons and text to represent the information and actions available to the user. The user interacts with the computer by manipulating the images and icons to perform the desired actions. Most users are not interested in the source code of the software application – they want an interface with a good layout and easy-to-use functions. The interface should not be cluttered with unnecessary information. The user interface is responsible for the efficient input of data and clear and understandable ouput from the system.

Help tools and **online support** are the backbone of all modern software packages:

- A **help tool** is an onscreen instructional program that provides further information about how a function works. It tells you how and when to use a specific instruction. Modern help tools are sophisticated, with an index and search function, making it easy for the user to understand and apply any function that they do not understand. You can press *F1* on the keyboard in Microsoft Windows operating systems and applications to call up the help function.
- **Online support** and **live support** give extra information when the help tool is inadequate. Online support is support for software supplied over the internet. It normally has a page listing frequently asked questions (FAQ) and extra information on the product's functions. When this is still inadequate, users can call the live support call centre where qualified staff help users with their problems by explaining what to do in a step-by-step manner.

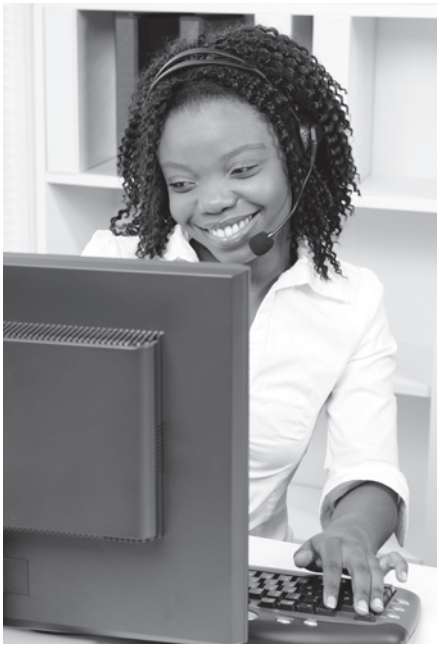


Figure 2.7 A live support call centre operator

Productivity

Equipment is productive when it achieves a significant result. In software terms, we say software is productive when it creates a large amount of output for every unit of input – simply put, this occurs when you get more out than you put in. A good software product that is easy to use and has functions that increase the user's **productivity** results in satisfied clients.

What types of functions create better productivity? Software packages have different functions, but they do share some general features. We shall point out a few of them to give you a perspective on how they simplify the end user's life:

- A well-designed interface with easy-to-use functions will increase a user's productivity immensely.
- A **toolbar** is a bar with tools or functions that enable the user to be more productive. A toolbar contains a cluster of functions in groups. When the user clicks on a tool, it performs a certain task.
- **Shortcut keys**, like toolbars, are an aid to help the user perform certain tasks more quickly by using a combination of keys. For example, by pressing *Ctrl + S* in a Windows-based application, the user saves the document he or she is currently working on.

Compatibility

Compatibility is the ability of a software package to work on different types of system software, to interact with different software packages and, most important of all, to run on different types of hardware configurations. When designing a software program, a software developer must take compatibility into account – not just compatibility with hardware, but with other software too (system software as well as other applications).

In software development there are two important types of compatibility, namely backward compatibility and forward compatibility:

- Backward compatibility (or downward compatibility) is the ability of a software package to interact with older versions of the same product. It refers to the relationship between two components rather than being an attribute of just one of them.
- Forward compatibility is the ability of a software package to accept instructions and data formats in later versions of itself.

Words & Terms

help tool: an onscreen program that provides further information about how any functions in an application work; it tells you how and when to use a specific instruction

online support: support for software supplied over the internet

live support: a qualified person who gives advice directly over the phone

productivity: the amount of output created per unit of input

toolbar: a bar with tools or functions that enable the user to be more productive

shortcut keys: an aid to help the user perform certain tasks more quickly by using a combination of keys

compatibility: the ability of a software package to work on different types of system software, to interact with different software packages and to run on different types of hardware configurations

In the workplace

Samantha works at a design company specialising in graphic design and image manipulation. The company uses different application software for various projects. A client comes to Samantha and asks her if she could help him manipulate a photo of his father into a drawing, which the client wants to frame as a birthday present for him on his 80th birthday.

Samantha tells the client about a new application that takes a raster image and converts it into a vector image, thus creating the illusion that it is a drawing. The client is excited and gives Samantha the photo of his daughter.

After Samantha scanned the photo, she opened the program. After loading the client's image, she used the software to create a vectorised model of the client's photo. You can see the result in Figure 2.8.

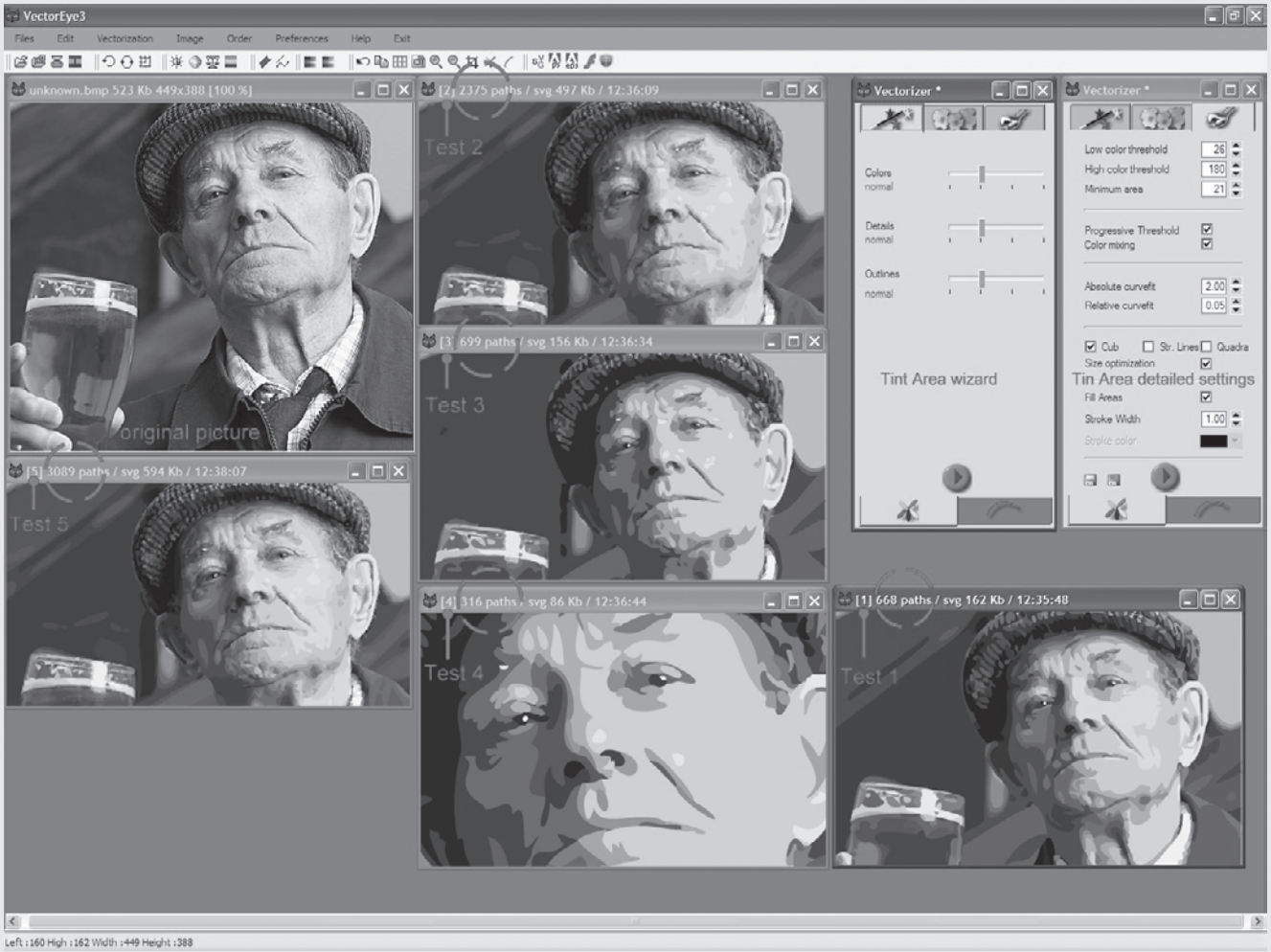


Figure 2.8 The original scanned photo and geometric model using vector graphics

Assessment activity 2.2

1. Explain the concepts application software and application suite. (4)
2. Choose an item from Column B to match a description in Column A. Write only the letter (A – L) next to the question number (2.1 – 2.10). (10)

	Column A		Column B
2.1	Microsoft Word	A	Application programs
2.2	Microsoft Excel	B	For browsing the World Wide Web
2.2	Microsoft 7/Vista/XP	C	For making a list of CDs in a collection
2.4	Norton, AVG, Panda	D	For sending/receiving emails
2.5	Microsoft Outlook	E	For typing a letter or CV
2.6	Microsoft Access	F	Operating systems
2.7	Internet Explorer	G	Utility programs
2.8	Calculator, System Restore, Disk Defragmenter	H	For creating a presentation
		I	For preventing viruses, spam
2.9	Microsoft Powerpoint	J	For doing calculations and drawing graphs
2.10	Smartdraw, Adobe Reader, Quicktime	K	For making drawings
		L	Sound editing

Total marks: 14

Unit 2.4 Installing application software

What is software installation?

Software installation is an automated process in which the user answers questions asked by an **installation wizard**. This program installs the software application by responding to input from the user. The user manual normally has detailed instructions on how to execute the installation process.

The installation process

A program consists of many files. When the program is installed, these files are stored on the computer's hard drive. The program does not use all these files while running, or at **run time**. Some files will be used for a short period only and then they will be released from memory. These files are called **run-time modules** or **dynamic linked library (DLL) files**. The printer driver is an example of a DLL file. Some of these files may need to be located in different directories because they are shared by different programs.

Words & Terms

software installation: an automated process in which the user answers questions asked by an installation wizard

installation wizard: a utility program that installs the software application by responding to input from the user

run time: the period during which a file is being used

run-time modules or **dynamic linked library (DLL) files:** files that are used for a short period only and are then released from memory, for example the printer driver

During the installation process, a program directory is created to store the main files of the program. The registry may need to be updated with file references. Computers that run on the Windows operating system have a **Windows registry**. This is a database that controls virtually every aspect of your computer's operation. Database files also need to be created. Because users do not know exactly where each file should be stored, the installation process is therefore automated by means of an installation wizard.

Words & Terms

Windows registry: a database that controls virtually every aspect of your computer's operation and that is found on all Microsoft Windows operating systems

Installing a software program

a) Welcome screen

Figure 2.9 shows the Welcome screen. This screen allows the user to exit the installation process before it starts. If the user selects Cancel, the installation process is aborted. To continue, click on Next.

Figure 2.9 Welcome screen

Did you know?

The installation process of a new program is controlled by one file, the SETUP.EXE file. The purpose of this file is to do most of the thinking during the installation process for you. When the user double-clicks on this file, the program installation starts and the wizard asks the user a series of questions.

b) Licence agreement

Figure 2.10 shows the **licence agreement**. The user uses the scroll bar on the right-hand side of the screen to view the agreement. The licence agreement offers the user the option of either accepting the terms and conditions of the software company or rejecting them. If the user decides to reject the agreement, the installation process will be aborted. If the user accepts the agreement, the Next button becomes active.

Words & Terms

licence agreement: a contract between the user and the software development company

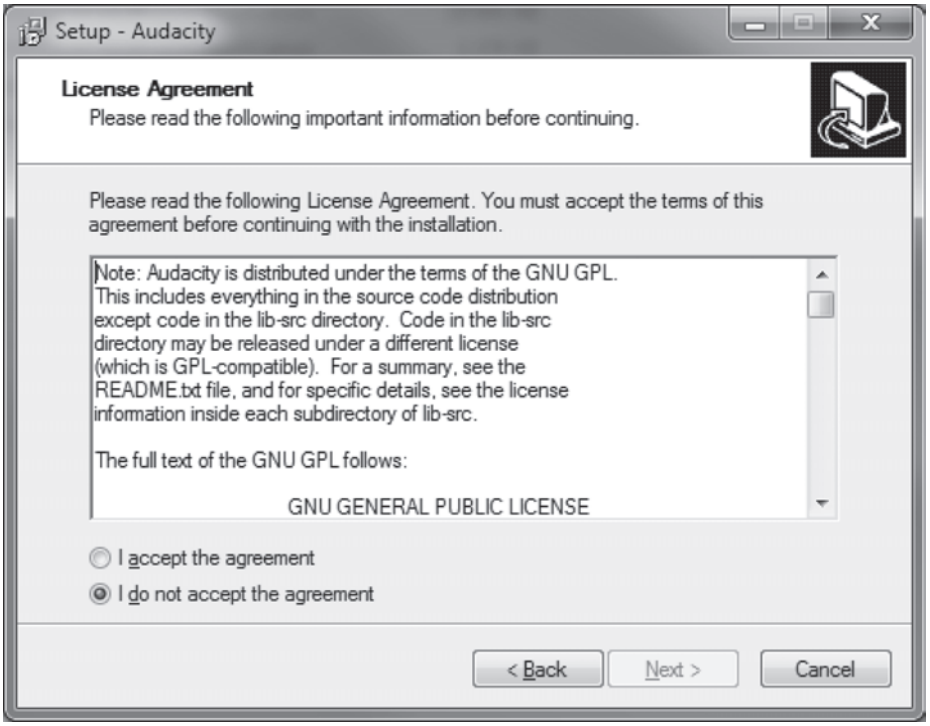


Figure 2.10 Licence agreement

c) Information screen

Figure 2.11 shows the information screen. In some cases, the installation process offers more information, such as the email address of the developer of the program. Click Next to continue with the installation.

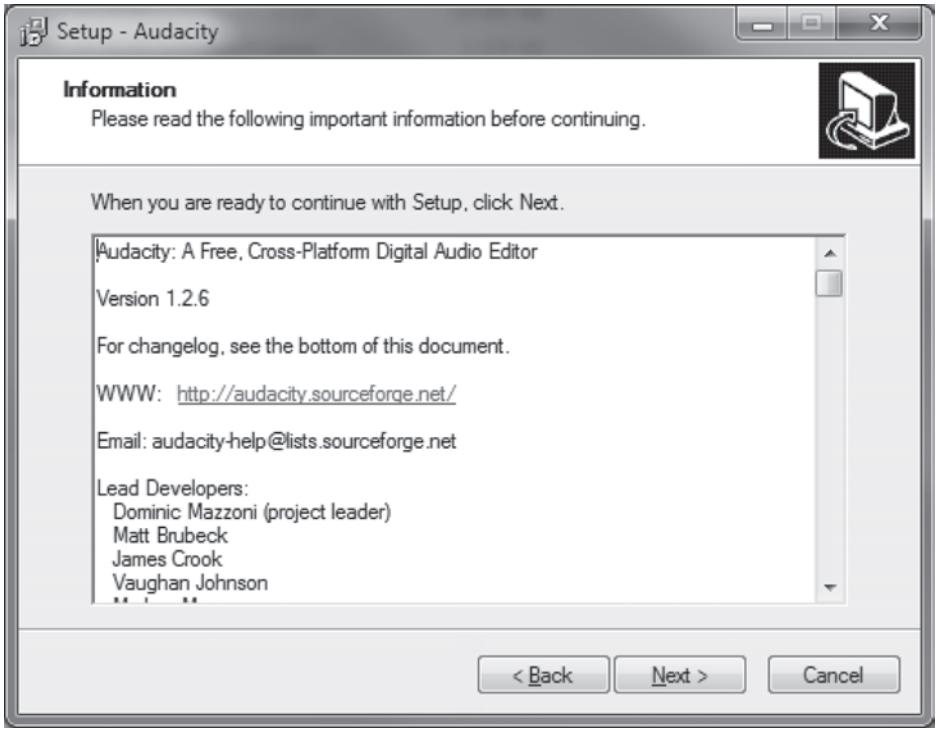


Figure 2.11 Information screen

d) Setup destination

Here, you need first to understand the difference between 32-bit and 64-bit machines. To explain this, think of a single-lane road on which cars may travel at 120 km/h. However, if there are many cars, then they will all have to slow down to, say, 100 km/h. If there is another lane, the traffic flow will be better and many cars will be able to travel at 120 km. A 64-bit machine means that there are 64 lanes for cars to travel in. The newer machines are all 64-bit machines. Unfortunately, some software does not know how to take advantage of the increased number of lanes. A 64-bit machine with Windows 7 helps programs to use all the lanes to full advantage and run faster.

Figure 2.12 shows the Setup destination screen. The directory where installation will take place is now determined. The user can click on the Browse button to find an alternative directory in which to install the program. If a 32-bit program is being installed on a 64-bit machine with Windows 7, the installation will take place in the Program Files (x86) directory.

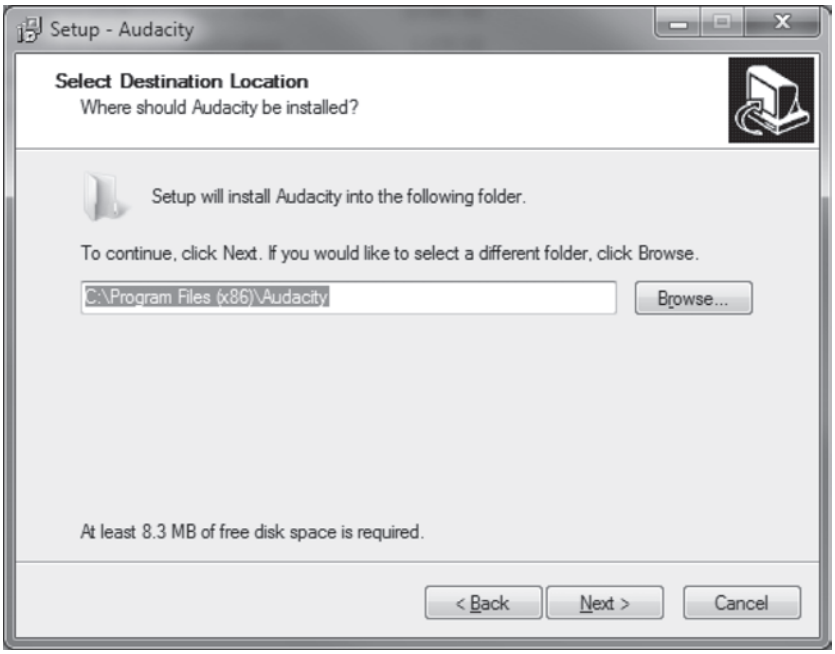


Figure 2.12 Setup destination screen

e) Folder exists warning

Figure 2.13 shows the Folder Exists warning. If the program is being reinstalled, the wizard will warn you that the directory already exists.

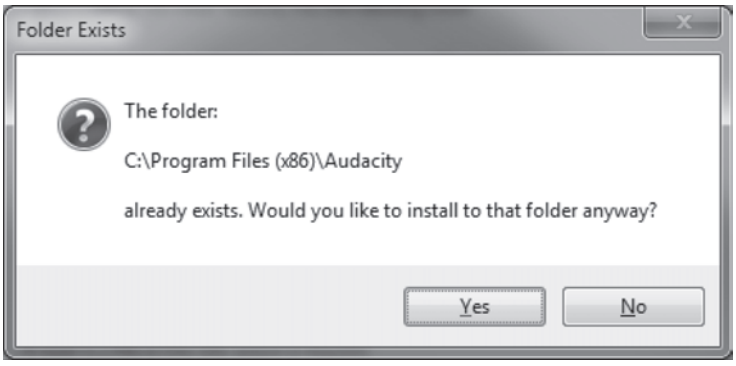


Figure 2.13 Information screen

f) Additional tasks

The user can perform additional task such as:

- creating desktop icons
- creating a program group in the Windows startup folder
- creating an uninstall program
- associating project files
- creating a quick-launch button on the taskbar.