EXPERIMENT NO: 03

Aim: Design Software Requirement Specification (SRS) document for the project. Consider any project to be developed in any technology as a software Architect or project Manager.

Theory:

- What is a Software Requirements Specification?

  A software requirements specification is a document which is used as a communication medium between the customer and the supplier. When the software requirement specification is completed and is accepted by all parties, the end of the requirements engineering phase has been reached. This is not to say, that after the acceptance phase, any of the requirements cannot be changed, but the changes must be tightly controlled. The software requirement specification should be edited by both the customer and the supplier, as initially neither has both the knowledge of what is required (the supplier) and what is feasible (the customer).

- Why is a Software Requirement Specification Required?

  A software requirements specification has a number of purposes and contexts in which it is used. This can range from a company publishing a software requirement specification to companies for competitive tendering, or a company writing their own software requirement specification in response to a user requirement document. In the first case, the author of the document has to write the document in such a way that it is general enough as to allow a number of different suppliers to propose solutions, but at the same time containing any constraints which must be applied. In the second instance, the software requirement specification is used to capture the user's requirements and if any, highlight any inconsistencies and conflicting requirements and define system and acceptance testing activities.

A software requirement specification in its most basic form is a formal document used in communicating the software requirements between the customer and the developer. With this in

mind then the minimum amount of information that the software requirement specification should contain is a list of requirements which has been agreed by both parties. The requirements, to fully satisfy the user should have the However the requirements will only give a narrow view of the system, so more information is required to place the system into a context which defines the purpose of the system, an overview of the systems functions and the type of user that the system will have. This additional information will aid the developer in creating a software system which will be aimed at the user's ability and the client's function.

- Types of Requirements

Whilst requirements are being collated and analysed, they are segregated into type categories. The European Space Agency defined possible categories as

- Functional requirements,
- Performance requirements,
- Interface requirements,
- Operational requirements,
- Resource requirements,
- Verification requirements,
- Acceptance testing requirements,
- Documentation requirements,
- Quality requirements,
- Safety requirements,
- Reliability requirements and
- Maintainability requirements

- Functional Requirements

Functional or behavioural requirements are a sub-set of the overall system requirements. These requirements are used to consider trade-offs, system behaviour, redundancy and human aspects. Tradeoffs may be between hardware and software issues, weighing up the benefits of each. Behavioural requirements, as well as describing how the system will operate under normal operation should also consider the consequences and response due to software failure or invalid inputs to the system.

 Performance Requirements

All performance requirements must have a value which is measurable and quantitative, not a value which is perceptive. Performance requirements are stated in measurable values, such as rate, frequency, speeds and levels. The values specified must also be in some recognised unit, for example metres, centimetre square, BAR, kilometres per hour, etc. The performance values are based either on values extracted from the system specification, or on an estimated value.

 Interface Requirements

Interface requirements, at this stage are handled separately, with hardware requirements being derived separately from the software requirements. Software interfaces include dealing with an existing software system, or any interface standard that has been requested. Hardware requirements, unlike software give room for trade-offs if they are not fully defined, however all assumptions should be defined and carefully documented.

 Operational Requirements

Operational requirements give an "in the field" view to the specification, detailing such things as:

- how the system will operate,
- what is the operator syntax?
- how the system will communicate with the operators,
- how many operators are required and their qualification?
- what tasks will each operator be required to perform?

- what assistance/help is provided by the system, any error messages and how they are displayed, and
- what the screen layout looks like.

 Resource Requirements

Resource requirements divulge the design constraints relating to the utilisation of the system hardware. Software restrictions may be placed on only using specific, certified, standard compilers and databases. Hardware restrictions include amount, percentage or mean use of the available memory and the amount of memory available. The definition of available hardware is especially important when the extension of the hardware, late in the development life cycle is impossible or expensive.

 Verification Requirements

Verification requirements take into account how customer acceptance will be conducted at the completion of the project. Here a reference should be made to the verification plan document. Verification requirements specify how the functional and the performance requirements are to be measured and verified. The measurements taken may include simulation, emulation and live tests with real or simulated inputs. The requirements should also state whether the measurement tests are to be staged or completed on conclusion of the project, and whether a representative from the client's company should be present.

 Acceptance Testing Requirements

Acceptance test requirements detail the types of tests which are to be performed prior to customer acceptance. These tests should be formalised in an acceptance test document.

 Documentation Requirements

Documentation requirements specify what documentation is to be supplied to the client, either through or at the end of the project.

The documentation supplied to the client may include project specific documentation as well as user guides and any other relevant documentation.

 Quality Requirements

Quality requirements will specify any international as well as local standards which should be adhered to. The quality requirements should be addressed in the quality assurance plan, which is a core part of the quality assurance document. Typical quality requirements include following ISO9000-3 procedures. The National Aeronautics and Space Administration's software requirement specification - SFW-DID-08 goes to the extent of having subsections detailing relevant quality criteria and how they will be met. These sections are Quality Factors:

- Correctness
- Reliability Efficiency
- Integrity
- Usability
- Maintainability
- Testability
- Flexibility
- Portability
- Reusability
- Interoperability
- Additional Factors

Some of these factors can be addressed directly by requirements, for example, reliability can be stated as an average period of operation before failure. However most of the factors detailed above are subjective and may only be realised during operation or post-delivery maintenance. For example, the system may be vigorously tested, but it is not always possible to test all permutations of possible inputs and operating conditions. For this reason, errors may be found in the delivered system. With correctness the subjectifies of how correct the system is, is still open to interpretation and needs to be put into context with the overall system and its intended usage. An example of this can be taken from the recently publicised 15th point rounding error found in Pentium (processors. In the whole most users of the processor will not be interested in values of that order, so as far as

they are concerned, the processor meets their correctness quality criteria, however a laboratory assistant performing minute calculations for an experiment this level or error may mean that the processor does not have the required quality of correctness.

 Safety Requirements

Safety requirements cover not only human safety, but also equipment and data safety. Human safety considerations include protecting the operator from moving parts, electrical circuitry and other physical dangers. There may be special operating procedures, which if ignored my lead to a hazardous or dangerous condition occurring. Equipment safety includes safeguarding the software system from unauthorised access either electronically or physically. An example of a safety requirement may be that a monitor used in the system will conform to certain screen emission standards or that the system will be installed in a Faraday Cage with a combination door lock.

 Reliability Requirements

Reliability requirements are those which the software must meet in order to perform a specific function under certain stated conditions, for a given period of time. The level of reliability requirement can be dependent on the type of system, i.e. the more critical or life threatening the system, the higher the level of reliability required. Reliability can be measured in a number of ways including number of bugs per x lines of code, mean time to failure and as a percentage of the time the system will be operational before crashing or an error occurring. Davis states however that the mean time to failure and percent reliability should not be an issue as if the software is fully tested, the error will either show itself during the initial period of use, if the system is asked to perform a function it was not designed to do or the hardware/software configuration of the software host has been changed. Davis suggests the following hierarchy when considering the detail of reliability in a software requirement specification.

- Destroy all humankind
- Destroy large numbers of human beings
- Kill a few people
- Injure people

- Cause major financial loss
- Cause major embarrassment
- Cause minor financial loss
- Cause mild inconvenience
- Naming conventions
- Component headers
- In-line document style
- Control constructs
- Use of global/common variables

 Maintainability Requirements

Maintainability requirements look at the long-term lift of the proposed system. Requirements should take into consideration any expected changes in the software system; any changes of the computer hardware configuration and special consideration should be given to software operating at sites where software support is not available. Davis suggests defining or setting a minimum standard for requirements which will aid maintainability i.e.

 Characteristics of a Good Software Requirements Specification

A software requirements specification should be clear, concise, consistent and unambiguous. It must correctly specify all of the software requirements, but no more. However, the software requirement specification should not describe any of the design or verification aspects, except where constrained by any of the stakeholder's requirements.

 Complete

For a software requirements specification to be complete, it must have the following properties:

Description of all major requirements relating to functionality, performance, design constraints and external interfaces.

Definition of the response of the software system to all reasonable situations.

Conformity to any software standards, detailing any sections which are not appropriate

Have full labelling and references of all tables and references, definitions of all terms and units of measure.

Be fully defined, if there are sections in the software requirements specification still to be defined, the software requirements specification is not complete.

 Consistent

A software requirement specification is consistent if none of the requirements conflict. There are a number of different types of confliction:

Multiple descriptors - This is where two or more words are used to reference the same item, i.e. where the term cue and prompt are used interchangeably.

Opposing physical requirements - This is where the description of real-world objects clash, e.g. one requirement states that the warning indicator is orange, and another states that the indicator is red.

Opposing functional requirements - This is where functional characteristics conflict, e.g. perform function X after both A and B has occurred, or perform function X after A or B has occurred.

 Traceable

A software requirement specification is traceable if both the origins and the references of the requirements are available. Traceability of the origin or a requirement can help understand who asked for the requirement and also what modifications have been made to the requirement to bring the requirement to its current state. Traceability of references are used to aid the modification of future documents by stating where a requirement has been referenced. By having foreword traceability, consistency can be more easily contained.

 Unambiguous

As the Oxford English dictionary states the word unambiguous means "not having two or more possible meanings". This means that each requirement can have one and only one interpretation. If it is

unavoidable to use an ambiguous term in the requirements specification, then there should be clarification text describing the context of the term. One way of removing ambiguity is to use a formal requirements specification language. The advantage to using a formal language is the relative ease of detecting errors by using lexical syntactic analysers to detect ambiguity. The disadvantage of using a formal requirements specification language is the learning time and loss of understanding of the system by the client.

- Verifiable

A software requirement specification is verifiable if all of the requirements contained within the specification are verifiable. A requirement is verifiable if there exists a finite cost-effective method by which a person or machine can check that the software product meets the requirement. Non-verifiable requirements include "The system should have a good user interface" or "the software must work well under most conditions" because the performance words of good, well and most are subjective and open to interpretation. If a method cannot be devised to determine whether the software meets a requirement, then the requirement should be removed or revised.

- Scenario: Industrial Employees Attendance Management System.

- INTRODUCTION:

An Industrial Employees Attendance Management System, allows the users (employees) to log in to their account and mark their respective attendance from anywhere through this software. This software helps the users by making the process of marking their attendance easily from their home due to work from home policies.

Purpose:

The software allows the user to access the attendance system and mark their attendance from anywhere by online accessing through the software.

Scope:

- An online attendance management system which provides attendance record, login-logout functions and other facilities.
- It has cut down the time-consuming and manual process of marking the attendance in the register by making online attendance marking facility available.

Definitions, Acronyms, and Abbreviations:

- It is a web application which provides attendance facilities to employees and admin access to the administrators.
- Few of the acronyms used in the software are o AB – Absent o P – Present o ID – Identity Card o Dt – Date

References:

- Software Engineering A practitioner's approach – Roger. S. Pressman
- Clean Code: A handbook of agile software craftsmanship. – Robert. C. Martin

- OVERALL DESCRIPTION

Product Perspective:

- This software product, i.e. an Industrial Employee Attendance Management System is a standalone product/software system.
- As hardware this software can be accessed through any web-browser.
- The software also stores the user's data in a secure database which contains user information, login details, etc.

Product Function:

- Employee information editing system.
- Login and logout system.
- Employee Attendance checking (Admin)

User Characteristics:

- Little to minimal education about a few technical terms.
- Little knowledge about compute/android device.
- Basic knowledge about web-browsing.

Constraints:

- A mobile phone with android version 7.0 or above, or a computer system with windows XP or above.
- An active internet connection for accessing the web site.
- Compatible web browser, e.g. chrome, brave, etc.

Assumptions and Dependencies:

- Users can log in through their respective username or user ID no.
- Admins will be able to print the attendance of the employees in an excel sheet format.

- SPECIFICATION REQUIREMENTS

External interfaces:

- It uses resources from hardware components which include RAM, Hard-disk, network card, etc.
- The web browser should also be the latest version for the software to run well.

Functions:

- Logging in and logging out by the employee at the time of starting, ending work respectively.
- Creating account for new employee in the attendance system
- Accessing employee attendance details - Editing the employee details.

Performance Requirements:

For Computer Users:

- Intel Pentium 4 processor or above
- 2 GB or more than that of RAM
- A minimum screen resolution of the display/monitor should be 800*600, whereas a display of 1600*900 or 1920*1080 is recommended.
- A compatible browser

For Android users:

- 512MB or more than that of RAM
- A compatible browser
- Android version 7.0 or above

Logical Database Requirement:

- As the software will be storing all the data on cloud storage, it does not need any write permissions.
- The admins should have access to the data in the database and should be able to fetch it whenever required.

Design constraints:

- As this software is a web-application, there are no specific constraints apart from the hardware requirements stated beforehand.
- The website can be accessed directly through any web-browser and a stable internet connection.

Software System Quality:

- Maintainability :
  - The client will have some time after we deliver this app to them, once the software application is delivered.
  - If anything happens during the period, the software will be fixed and updated without any extra charges.

- Correctness:
  - The tests done by the testing team should be done manually by the testing team members, so that the accuracy of the checks can be maintained, and errors are debugged before the delivery of the product.

- Flexibility :
  - The software system should be flexible and any changes that the user needs after the software system is delivered shall be made to the software at the decided cost by the customer and firm in the contract.

Object Oriented Models:

- The online Industrial Employees Attendance Management System can be described by the use of class diagram, flow state diagram and other such models to show the relationship between the various classes that compose the system.

Appendices:

- RAM – Random Access Memory
- MB – Mega Byte GB – Giga Byte

Index:

 Conclusion:

Hence, by performing this practical I learnt about the Software Requirement Specification (SRS) document, its requirement and how to design a SRS document. I also designed a Software Requirement Specification (SRS) document for the 'Industrial Employees Attendance Management System' considering the software to be developed in any technology as a software Architect or a project Manager.

(10)	(20)	(10)	(10)	TOTAL (50)