International Journal For Technological Research In Engineering
Volume 8, Issue 5, January-2021 *(Online): 2347 - 4718*

SOFTWARE REQUIREMENT ANALYSIS

[1]Hrithik Sharma, [2]Varun, [3]Mrs. Indu Khatri
[1,2]Student, [3]Assistant Professor
Department of Computer Science and Information Technology
BMCEM, Sonipat, India

Abstract: ***we describe Software Requirement analysis as an important role in taking a step towards designing software. Software Requirement Analysis means that the developer analysis the requirements of the client carefully and with full concentration so that the developer don't face the problem after the development of the software like the client denies some of the functions that were given by the developer which he thought was good for the software bur the client doesn't understand and wants them to be removed from the application. This kind of work may make the developer bad for the work that he did was of no use. We argue the cases that are to be remembered before the developer starts the development of the software. So, that the developer might have to face less consequences after remembering these factors. Here, we have tried to discuss every possible factor that might affect the developer in different circumstances. He models that the developer can use to show the idea of designing the software to the client.***

Keywords : ***Entity relationship model, Data flow diagram, State-transition diagram, Cardinality, Data store, leveling.***

1. INTRODUCTION

In speculation to the development of the software, the first question the developer should ask is whether after getting the work from the client should the work began the same day or there are some factors that developer should examine before beginning the project. These are the factors that the developer should always analyses. We are going to discuss these factors here and we will clarify every factor that is important for the developer to always remember.

We think that as the current society is stepping towards the software infrastructure rapidly. This rapid growth is due to which many developers are coming forward to begin their own business and so these developers are hard working on the project they get from the client. There is some time that they may make tough decisions and may face future consequences. So to not to just take decisions we have worked on some factors that the developer should focus on during taking the responsibility to start the project.

Analysis is the process of breaking down something into parts to learn what they do and how they relate to one another. Hence, keeping that in mind, we have come up with some factors that might help the developer not to make rash decisions.

Our article is divided into two parts. In first part we discuss the analysis of developer i.e., the developer should see that how he/she is efficient for the project and also other factors like time, team etc. In the second part, we have discussed different models that can be used to show how developer is going to develop the software.

I. Analysis

Researchers who work on software analysis only focus on the different factors which make the development of the project easy i.e., using different models that can make the project development in time. We don't say, that is not the wrong way to focus on, rather we argue that there are other factors too, which also depend whether the responsibility taken to develop the project has to be fulfilled. The factors that affect the development of project are discussed here.

A. On Ourselves

Analysis on us means that the developer should not only analysis the conditions which are useful for the fulfillment of the software that is being developed but also on him/her. Hereby we mean that, the develop is suitable for the requirements for the develop the software.

B. Time

Analysis of time, this is taught to us at every aspect of our life, is also very important key which is to be decided initially by the developer before the client hands over the project. The client needs to know by how much time he has to wait before the developer hands over the project. So, here is the race of time, developer has to give appropriate time in which he is capable of finishing the project with proper satisfaction and without getting rush to the date, which may make the software to be handed over without getting perfectly getting examine. Due to this the software the developer builds could cause problem in future which would make bad impressions on further customers.

This time analysis has to be taken a precisely so that the developer should not face the rush to the development of the project due to the submission date getting closer. If such type of mistake is made this will surely make the team to work hard then the usual and can cause conflicts between the team. The time analysis is essential that has to be made by the developer before taking the project as he/she has to also compete against other developers. So, the time that is chosen for the developer precisely such that the project is developed within the time given to the client.

C. Team

Team is the core of the project development. So, the developer should choose team wisely which can analyses the situation and take correct measures towards the situation to handle the problem. Team is a collective of the members who believe in each other and help the other team members so that they don't have conflicts the work that they have to do.

The developer sometimes chooses team members who are not well trained/experienced to keep more money to him. When the software crashes then they have some difficulty verifying what actually caused the problem. When they are given to check the code on different platforms, they might lose some of the platforms because of the less experience. Such type of mistake may cause problems for the developer when the client checks the software after the submission.

We do not say that the new programmers should not be given chance on a project, rather we argue that the developer should test the team before starting the project or should hire the experienced programmers so that the project can be developed on time and with every test to be tested.

D. Money

Development of a project is done by team but to get a team to work on a project the developer needs the money. The money the developer demands from the client have to choose correctly so that in future the developer doesn't have to face the problem for getting broke.

The developer should use a good team, rather than thinking of making more money just by getting new developers and giving them low salary and making money. Such type of decision could cause problems for the development of project. So, the developer should ask for the money after thinking of the places where the money should be spent and how much.

II. Models

At technical level, software engineering begins with a series of modeling that lead to a complete specification of requirements for the software to be built. Here, structured analysis, we have discussed is a classical modeling method i.e., Structured analysis is a model building activity. While developing the software the developer should remember the main three principal objective of the analysis model:

1) To describe what the customer requires
2) To establish a basis for the creation of a software design
3) To define a set of requirements that can be validated once the software is built.

As to accomplish these objectives, the develop should remember the structured analysis. The structured analysis model is shown in fig. 2.1. At the core of the model is the data dictionary. The data dictionary is a location where the description of all data objects consumed or produced by the software is stored. There are three other diagrams that surround the core.

The entity relation diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties. The ERD is the notation that is used to conduct the data modeling activity. The attributes of each data object noted can be described using a data object description.

The data flow diagram (DFD) is a way of representing a flow of data through a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. A DFD has no control flow i.e., there are no loops and no decision rules. A description of each function presented in the DFD is contained in process specification (PSPEC).

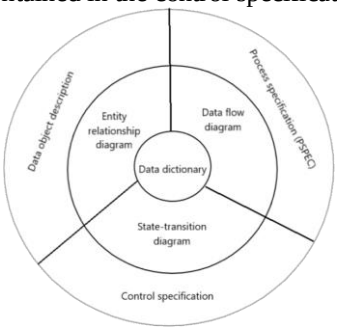
The data flow diagram serves two purposes:

(1) To provide an indication of how data are transformed as they move through the system.

(2) To depict the functions that transforms the data flow.

The state transition diagram (STD) indicates how the system behaves as a consequence of external events. To accomplish this, the STD represents the various modes of behavior (called states) of the system. It describes all of the states that an object can have, the events under which an object change state, the condition that must be fulfilled before the transition will occur, and the activities undertaken during the life of an object.

So, the STD serves as the basis for behavioral modeling. Additional information about the control aspects of the software is contained in the control specification (CSPEC).



A. Entity relation Diagram

The object/relationship pair can be represented graphically using the entity/relationship diagram. The primary purpose of the ERD is to represent data objects and their relationships with one another. A component of data, an object in this context is called as an entity. An entity set is a collection of similar entities. These entities can have attributes that define its properties.

1. Common Entity Relationship Diagram Symbols: An ERD is a means of envisioning how the information a system produces is related. There are five main components of an ERD:

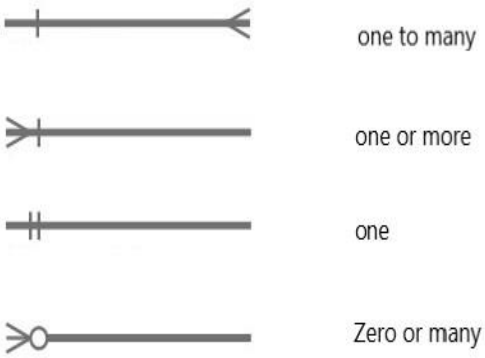
a) Entities: These are represented by rectangles. An entity is an object about which you want to store the information.

b) *Actions:* These are represented by diamond shapes. They show how two entities share information in the database. They are also named as relationship between Entity A and Entity B.

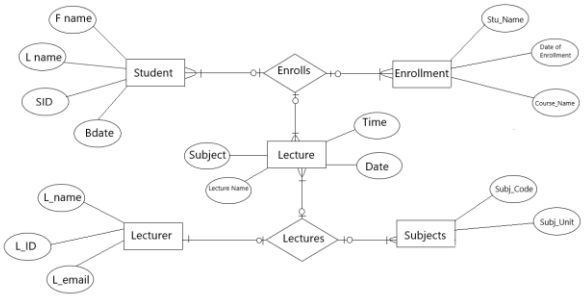
c) *Attributes:* These are represented by ovals. An attribute distinguishes the characteristics of the entity.

d) Connecting lines: These are the solid lines that connect attributes to show the relationships of entities in the diagram.

e) Cardinality: Number of instances of entity B that can be associated with each instance of entity A.



2. *Example for ERD:* Above we have discussed what the components of the entity relation diagram are. Here we have shown an example for the college enrolment system i.e., when student get enrolls and find their lectures to study.



B. Data Flow Diagram

A data flow diagram is a graphical representation that depicts information flow and the transforms that are applied as data move from input to output. The DFD also provides information about the outputs and inputs of each entity and the process itself. The DFD provides a mechanism for functional modeling as well as information flow modeling.

1) Notation: A data-flow is a path for move from one part of the information system to another. A data-flow may represent a single data element or it can represent set of data element.

- Straight lines with incoming rows are input data flow.
- Straight lines with outgoing arrows are output data flow

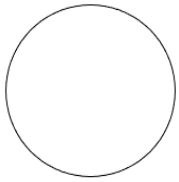
2) *Rules of Data Flow:* For developing DFD we have to remember some rules so that we don't face problems. These rules are discussed below:

- All flow must begin and end with and end at a processing step.
- All names should be unique.
- An entity cannot provide data to another entity without some processing occurred.
- Data cannot move directly from an entity to a data story without being processed.
- Data can't move directly from a data store without being processed.
- No cross Line in DFD

3) Symbols: The symbols use to show the data flow diagram:

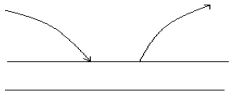
a) *Data flow:* The data flow is show through the arrow. The direction of the arrow shows where the data is flowing.

b) *Process:* The process in data flow diagram is shown by the circle. This is in between the input and output.



c) *Source or Sink:* Source is used for all the input that is coming in the data flow diagram and sink is used for all the output that are result of the data flow diagram.

d) *Data store (A repository of data):* The symbol used for this is:



4. *Leveling:* Leveling in data flow diagram is very important. This means that we define different levels in data flow diagram to show the deeper information i.e., more the level increase we show the deeper information that is taking place in the source.

Level-0 DFD: This leveling is the fundamental system model or the context diagram.

C. State- Transition Diagram

State-transition diagrams describe all of the states that an object can have, the events under which an object changes state, the conditions that must be fulfilled before the transition will occur and the activities undertaken during the life of object. State-Transition diagrams are very useful for describing the behaviour of individual objects over the full set of use cases that affect those objects.

The control specification contains a state transition diagram that is a sequential specification of behaviour. The control specification represents the behaviour of the system in two different ways. It can also contain a program activation table- a combinational specification of behaviour.

1) *Notation:* These are some of the notations that we use for the state-transition diagram.

a) State**:** A condition during the life of an object in which it satisfies some condition, performs some action, or waits for some event.

b) Event: An occurrence that may trigger a state transition. Event types include an explicit signal from outside the system, an invocation from inside the system, the passage of a designated period of time, or a designated condition becoming true.

c) Guard*:* A Boolean expression which, if true, enables an event to cause a transition.

d) Transition*:* The change of state within an object.

e) Action*:* One or more actions taken by an object in response to a state change

CONCLUSION

The Software Requirement analysis helps the developer to take correct decision for developing the software. The models help the developer to show the client how the software is being designed. This kind analysis helps the developer to make correct decisions which result in the success of the project. The different models discussed here are suitable for showing the information that has to be displayed on the software.

REFERENCES

- Software Engineering - A Practitioner's Approach - Pressman (5Th Ed, 2001) by Roger S. Pressman, Ph.D.
- ERD from the site giving the information [Online] https://www.smartdraw.com/entity-relationship-diagram/
- DFD from the site giving the information [Online] https://www.visual-paradigm.com/guide/data-flow-diagram/what-is-data-flow-diagram/
- STD from the site giving the information [Online] https://www.stickyminds.com/article/state-transition diagrams#:~:text=State%2Dtransition%20diagrams%20describe%20all,of%20an%20object%20(actions)

