Topic 13 Systematic Approach to Problem Solving

Stages of Problem Solving

The process of solving computer based problems is made of several different stages and these are used by developers to create and maintain systems. A system is more than just the computer program itself and often includes hardware, file structures and the people using the system.

The stages flow from one to another and depending on the problem at hand.

Problem Analysis

The first stage of problem solving is to identify and fully analyse the problem.

Defining the Problem

It is important to have a clear understanding of what problem a new system must solve. This could be a new area which needs to be developed, or there could be an existing system which needs to be improved upon.

It is important to keep an open mind at this stage and focus on understanding the problem rather than thinking about potential solutions. It is also important to clearly define the scope of the problem and to be realistic about how much of the problem can be solved by the new system.

It is important to clearly define the requirements for the new system along with any constraining factors such as time or cost which need to be considered. It is important that everyone involved in the project understands and agrees on these details.

Ways to Gather Information

It is likely that additional information may be needed to help understand the problem and determine the requirements for the new system. This is especially true if there is an existing system or process in place. In this case, it is worthwhile to understand what works well about the existing system as well as what needs to be improved.

User interviews involve talking with those who use and work with any existing system or will be using the new system. Different groups will have different needs and experiences with the system, and so it is important to interview different groups. For example, the customers of an online shopping website will have a different view to the shopkeeper. This is a time consuming process, but provides valuable information from people with first hand experience. People may give a very personal view of the system and so personal preferences such as menu colour should be filtered out from actual system requirements.

Questionnaires or surveys can be used to gather information more quickly than user interviews and make it easier to analyse the responses. They are however restricted to more closed questions.

Observing people using the existing system gives a good idea of how the system is used, and which parts of it are most frequently accessed. It also has the advantage of providing a second pair of eyes to spot any mistakes or things which others may have missed.

Design

Before the new system can be implemented, it needs to be designed. The design is based on the information, requirements and limitations gathered as part of the analysis stage.

Large projects are usually too big to be designed as a single problem and so are broken down into smaller modules. Each module is a self contained unit built and tested by a single programmer or small team. This allows several people to work on the solution at the same time. Programmers must make sure to consider how the modules will link together and make sure that modules have a consistent design and approach.

Data Structures

The design stage also looks at how the system will handle data, what data will be input and output, how the data will be processed and what data structures will be used. Careful planning at this stage will prevent problems occurring in the future. Programmers will need to consider what variables will be used, whether they need to be declared, and other characteristics such as the type and length of the variable.

Data Models

Data Dictionaries show what data will be used and how it is stored, allowing all those working on the project to clearly understand how data is handled. They can be seen as a database about data, they hold information about the data but not the data itself. A Data Dictionary might store the data type, length, title and any validation checks.

Data Flow Diagrams

A data flow diagram is a picture which shows how data moves through the system, making it easy to understand how data flows. It does not store information about the type of data.

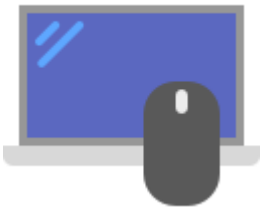
Designing Algorithms

The design stage also produces descriptions of any algorithms which are needed. The algorithms are not fully written at this stage, but basic pseudo code may be produced to give a better idea of the algorithms' function and purpose.

The Human User Interface

This term means any communication between the system and the people using it. This could be a graphical user interface on a computer or mobile device but could also be the layout of buttons on an alarm clock or the controls of a car or other vehicle. When designing the interface several factors need to be considered:

- Ease of use, meaning how intuitive the interface is to use and how easily common tasks can be carried out.
- Target audience. Different groups of people have different experiences and skills, and this should be considered when designing the interface. An interface for a system used by young children would look very different to one used by adults.
- Technology. The technology and hardware available to those using the system has a big impact on the design of the interface. An interface designed for a desktop computer with a large screen, keyboard and mouse will look very different to one designed for a mobile phone with a small touchscreen. Similarly, if users will be accessing the system via The Internet and have a slow connection, the interface will need to be kept simple and should not include large pictures or videos which would make it slow to use.



- Ergonomics. The interface needs to be comfortable to use, especially for systems which are used for extended periods of time. Programmers must think about how easy colours and fonts are to look at and read, and in some cases the physical placement of buttons and other controls.

Implementation

This stage is where the system is built and created using the design created in the previous stage. The people building the system will need to be aware of the requirements and considerations from the analysis stage as well as the design from the design stage.

Depending on the design and size of the system, several modules might be built at the same time. Sometimes, a module can't be built until another module has been created. This is known as one module being dependent on another. This also means that changes in one module may affect the design or requirements of other modules.

Testing

When building a new system, especially if it is large and complex, there is a good chance that it will not work correctly the first time. Sometimes these problems are obvious and easy to spot, but in other cases they may be more subtle or only occur under specific circumstances.

The testing stage aims to find and fix errors within the system, and ensure it performs as it should. There are several different types of testing which can be carried out and different data types used to help with this.

Testing is usually carried out as the system is being implemented as well as at the end. As individual modules are completed, they should be tested to incur the function correctly, and as the project progresses, the modules can be joined together and tested once more to ensure they function correctly together.

Test Data

Test data has a known result that the system should output and can be used to check that the system is processing data correctly. Test data will need to be generated to cover different ways in which the system might be used and to cover different possible scenarios. There are three types of test data:

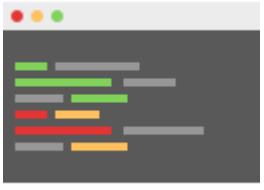
- Normal data is that which the system is expected to be able to handle and generate a known result. For example, normal data for a date of birth field might be 01/12/1982.
- Boundary data is at the extreme ends of what is acceptable but should still be processed by the system. For example, boundary data for an age field could be 1, or 110.
- Erroneous data is incorrect, and the system is not expected to process. The system should not process the data and generate an error without crashing. For an age, boundary data might be 999, -1 or even green.

Development Testing

Development testing takes place during the development of the system, with individual lines of code or modules being tested on their own or in small groups. Testing at this stage makes it easier to locate and correct errors without having to work through large numbers of modules:

- Black box testing involves entering test data and checking the output against the expected output. This tests that the code is functioning correctly and producing the correct output but does not look at how the code itself works.
- White box testing involves testing every aspect of the code, checking every possible pathway through the code, and adding extra commands if needed to fully understand what is happening.
- Unit testing makes sure that each individual unit or module carries out its own function correctly using both black and white box testing.
- Integration testing makes sure that the different modules tested during unit testing continue to function correctly when they have been joined together.

System Testing



System testing looks at the system as a whole to make sure it does not contain any errors and that it provides a solution to the initial problem.

- Alpha testing uses test data created in house to test the system under normal conditions looking at all the different ways the system might be used. Testing the system in house before it is used by live users makes it easier to stop, start and change the system if needed to fix errors. It also means users are less likely to be impacted by system issues.

- Beta testing uses an early version of the system known as a beta version. This is released to potential users who are encouraged to use and test the system and to report any bugs they find to the programmers. This allows the system to be tested with live data and real users, but without impacting a large number of users. Since the end users were not involved in building the system, they are likely to use it in different ways making it more likely for them to find errors missed by the original team.

Acceptance Testing

Acceptance testing is carried out by the person who asked for the system initially and those who will be using it. They enter their own live data and ensure that the system solves their problem as identified in the analysis stage. This allows them to ensure the system meets their needs and is fit for purpose.

Evaluation

At the evaluation stage, the problem has been analysed and a solution has been designed, implemented and tested. The evaluation looks at how the system compares with the initially agreed specification and whether any improvements or refinements may be needed in the future. These details often feed back into the analysis stage to form a cycle.

The evaluation process can take place over a long period of time, and complex systems are often under constant evaluation. The things considered during evaluation include:

- Functionality. Does the system function as intended?
- Ease of use. Are the intended users able to use the system effectively and efficiently?
- Ease of implementation. How easy was the system to bring online, and to transfer to from any old systems?
- Reliability. Does the system work without errors or failures?
- Performance. Does the system meet the performance requirements identified in the analysis and design stages?
- Cost effectiveness. How much does it cost not only to implement the system but to run and maintain it?
- Ease of maintenance and adaptability. How easy is the system to fix when it breaks and to change when needed?
- Longevity. How well is the system lasting, is it coping with any changes and future proof.

The answers to these questions often change as time goes by, the system ages and more data is added. Changes in what users need from the system or in the skills of those using and maintaining the system may also feed in here.

Prototypes

Creating a full system can be expensive and time consuming, especially for large systems. The people who have asked for or will be using the system might also ask for changes once they see the finished system.

Prototypes are key parts of the system which are built in advance with limited functions to give an idea of how the system will look and work. Feedback on the prototype can be used to change and update the design and any requirements.

A prototype usually only has one or two key areas included. For example, it may include a full user interface, but with no actions attached to any of the button.