

Lecture No.8

State Transition Diagrams

State transition diagrams (STDs) are another technique to document domain knowledge. In many cases, information flows from one place to the other and at each place certain action is taken on that piece of information before it moves to the next place. A file in an office is a typical office is example of such system. In this case, different people make comments and add information to that file and it moves from one table to the other this movement is controlled by a pre-defined set of rules which define under what condition the file moves from place A to place B and so on. We can easily document these set of rules with the help of state transition diagrams.

Following is an example of a use of STD to document the life cycle of a trouble ticket (this example has been taken from ITU-X.790 document).

A Trouble report and its life cycle – and introduction

From time to time all systems, including communications networks, develop problems or malfunctions referred to in this Recommendation as “troubles”. A “trouble” in a communications network is a problem that has an adverse effect on the quality of service perceived by network users. When a trouble is detected, possibly as a result of an alarm report, a trouble report may be entered by a user or the system may raise a report automatically. Management of that trouble report is necessary to ensure that it receives attention and that the trouble is cleared to restore the service to its previous level of capability.

At the time of a trouble, a network may have been inter-working with another network to provide a service, and the problem or malfunction may be due to the other network. Therefore it may be necessary to exchange trouble management information between management systems across interfaces which may be client to service provider or service provider to service provider interfaces and may represent inter-jurisdictional as well as intra-jurisdictional boundaries. In addition to exchanging information on trouble that has already been detected, advance information on service inaccessibility may also need to be exchanged. Thus, a service provider may need to inform a customer of future service inaccessibility (because of planned maintenance, for example).

Trouble report states and status

Referring to the State transition diagram in Figure 2, a trouble report may go through any of six states during its life cycle. In addition, a Trouble Status attribute is defined which qualifies the state (finer granularity) e.g. cleared awaiting customer verification. The time at which the status attribute change is also captured in the trouble report

.

Following is a description of states of a trouble report.

Dr. Syed Rafaqat Hussain Kazmi

Queued

A trouble report is in a queued state when it has been instantiated but the trouble resolution process has not yet been initiated. A trouble report which is in the queued state may be cancelled by the manager. The agent on receiving such a request will attempt to close the trouble report.

Open/active

The trouble report becomes “open/active” when appropriate actions to resolve the trouble are initiated.

An “open/active” trouble report may be “referred” to another Hand-off Person, or “transferred” to another Responsible Person for further processing. The state however remains unchanged as “open/active”. A trouble report in the open/active state may be cancelled by the manager. The agent on receiving such a request will attempt to close the trouble report.

Deferred

This state indicates that corrective action to resolve the trouble has been postponed. This can occur when the faulty resource is inaccessible for a period and repair activity cannot proceed. A deferred Telecommunications Trouble Report may become “open/active” again, or move directly to the “closed” state if it is cancelled for some reason. A trouble report in the deferred state may be cancelled by the manager. The agent on receiving such a request will attempt to close the trouble report.

Cleared

A trouble report is moved by the agent to the “cleared” state when it determines that the trouble has been resolved. If the manager needs to verify that the trouble has been resolved, verification may optionally be awaited by the agent prior to closure of the trouble report.

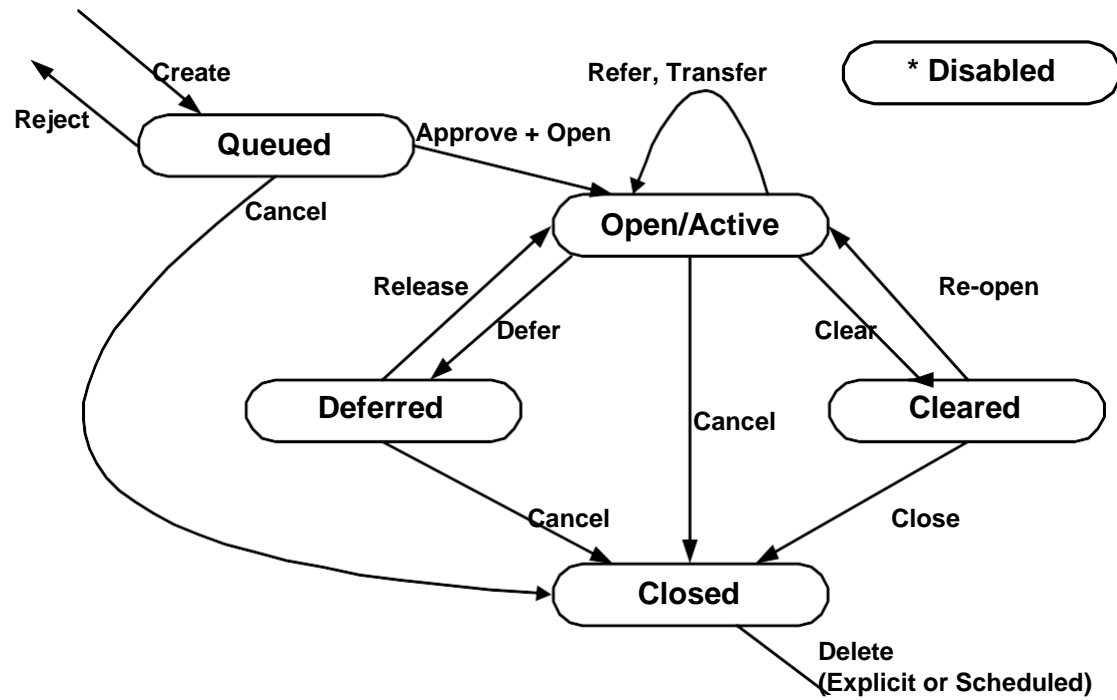
Closed

This state indicates that the trouble resolution process is complete. Upon closure, the trouble report attributes are captured in a historical event generated at trouble report closure which may then be stored in a log of trouble history records, for future reference. The trouble report may then be eliminated at the agent’s convenience. However, the agent may be required to maintain such records for a period of time as per business agreements.

Disabled

A “disabled” value is exhibited when a trouble report’s information cannot be updated due to local conditions. In the “disabled” condition only read operations can be performed.

The following figure shows the STD for a trouble ticket. This diagram depicts the movement of a trouble ticket from one state to the other, thus making it easy to understand.



Arranging information in tabular form

Sometimes it is better and more convenient to arrange information in a tabular form. This makes it easier for the reader to understand and comprehend the information and hence designing, coding, and testing become less challenging. As an example, let us look at the following definitions used for identifying different data functions in the function point analysis taken from International Function Point User's Group (IFPUG) Counting Practices Manual (CPM 4.1).

External Inputs

An external input (EI) is an elementary process that processes data or control information that comes from outside the application boundary. The primary intent of an EI is to maintain one or more ILFs and/or to alter the behavior of the system.

External Outputs

An external output (EO) is an elementary process that sends data or control information outside the application boundary. The primary intent of an external output is to present information to a user through processing logic other than, or in addition to, the retrieval of data or control information. The processing logic must contain at least one mathematical formula or calculation, or create derived data. An external output may also maintain one or more ILFs and/or alter the behavior of the system.

External Inquiry

An external inquiry (EQ) is an elementary process that sends data or control information outside the application boundary. The primary intent of an external inquiry is to present information to a user through the retrieval of data or control information from an ILF or EIF. The processing logic contains no mathematical formulas or calculations, and creates no derived data. No ILF is maintained during the processing, nor is the behavior of the system altered.

It is difficult to understand these definitions and one has to read them a number of times to understand what is the difference between EI, EO, and EQ and in which case a function would be classified as EI, EO, or EQ.

Now the same information is presented in the tabular form as follows:

Function	Transactional Function Types		
	EI	EO	EQ
Alter the behaviour of the system	PI	M	N/A
Maintain one or more ILFs	PI	M	N/A
Present information to the user	M	PI	PI

PI – Primary intent; M – may be; N/A – not allowed.

This table simply says that a function can alter the behaviour of the system, it can maintain one or more ILFs, and/or it can present information to the user. The next step is to determine

whether it is EI, EO, or EQ. For that we have to determine what is the primary intent (PI) of the function and in addition to this primary intent, what else does it

do. Identification of EQ is simple - in this case the only thing a function does is present information to the user, which is also its primary intent. If it alters the behaviour of the system or maintains and ILF then it can either be an EI or and EO but not an EQ. On the other hand if the primary intent of the function is to present information to the user but at the same time it also performs any of the first two operations, it is an EO. Finally, if the primary intent of the function is either to alter the behaviour of the system or maintain one or more ILFs, then it is an EI.

Hence by putting and organizing the information in the form of a table, we have not only made it simple to understand the definition but also given an holistic picture which was not easily visible otherwise.

Let us look at another example. This time the information is taken from the Income Tax Ordinance of Pakistan 2001. Consider the following statement that describes the income tax rates applicable to people with different brackets:

If the taxable income is less than Rs. 60,000, there will be no income tax. If the income exceeds Rs. 60,000 but is less than Rs. 150,000 then income tax will be charged at the rate of 7.5% for income exceeding Rs. 60,000. If the income exceeds Rs. 150,000 but does not exceed Rs. 300,000 then the income tax will be computed at 12.5% of the amount exceeding Rs. 150,000 plus Rs. 6,750. If the income exceeds Rs. 300,000 but does not exceed Rs. 400,000 then the income tax will be computed at 20% of the amount exceeding Rs. 300,000 plus Rs. 25,500. If the income exceeds Rs. 400,000 but does not exceed Rs. 700,000 then the income tax will be computed at 25% of the amount exceeding Rs. 400,000 plus Rs. 45,500. If the income exceeds Rs. 700,000 then the income tax will be computed at 35% of the amount exceeding Rs. 700,000 plus Rs. 120,500.

The same information can be organized in the form of a table, making it more readable and easier to use.

Income	Tax
Less than Rs. 60,000	0%
Between Rs. 60,000 and Rs. 150,000	7.5% of (Income - 60,000)
Between Rs. 150,000 and Rs. 300,000	12.5% of (Income - 150,000) + 6,750
Between Rs. 300,000 and Rs. 400,000	20% of (Income - 300,000) + 25,500
Between Rs. 400,000 and Rs. 700,000	25% of (Income - 400,000) + 45,500
Greater than Rs. 700,000	35% of (Income - 700,000) + 120,500

Once the information has been organized in the tabular form, in many cases it can be simply stored and mapped onto an array or a database table and the programming of this

kind of a rule is simply reduced to a table or dictionary lookup. This reduces the complexity of the domain and hence reduces the over all effort for designing, coding, testing, and maintaining the system.

Data Flow Model

- Captures the flow of data in a system.
- It helps in developing an understanding of system's functionality.
- What are the different sources of data, what different transformations take place on data and what are final outputs generated by these transformations.
- It describes data origination, transformations and consumption in a system.
- Information is organized and disseminated at different levels of abstraction. Thus this technique becomes a conduit for top down system analysis and requirements modeling.

The Notation

There are several notations of the data flow diagrams. In the following, four different shapes are explained.

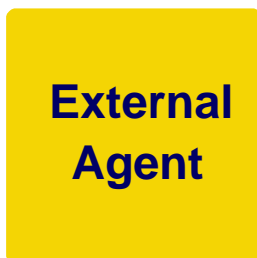
Process

- What are different processes or work to be done in the system.
- Transforms of data.



External Agent

External systems which are outside the boundary of this system. These are represented using the squares



Data Store

- Where data is being stored for later retrieval.
- Provides input to the process
- Outputs of the processes may be going into these data stores.

Dr. Syed Rafaqat Hussain Kazmi



Data Flow

- Where the data is flowing.
- Represents the movement of the data in a data flow diagram.



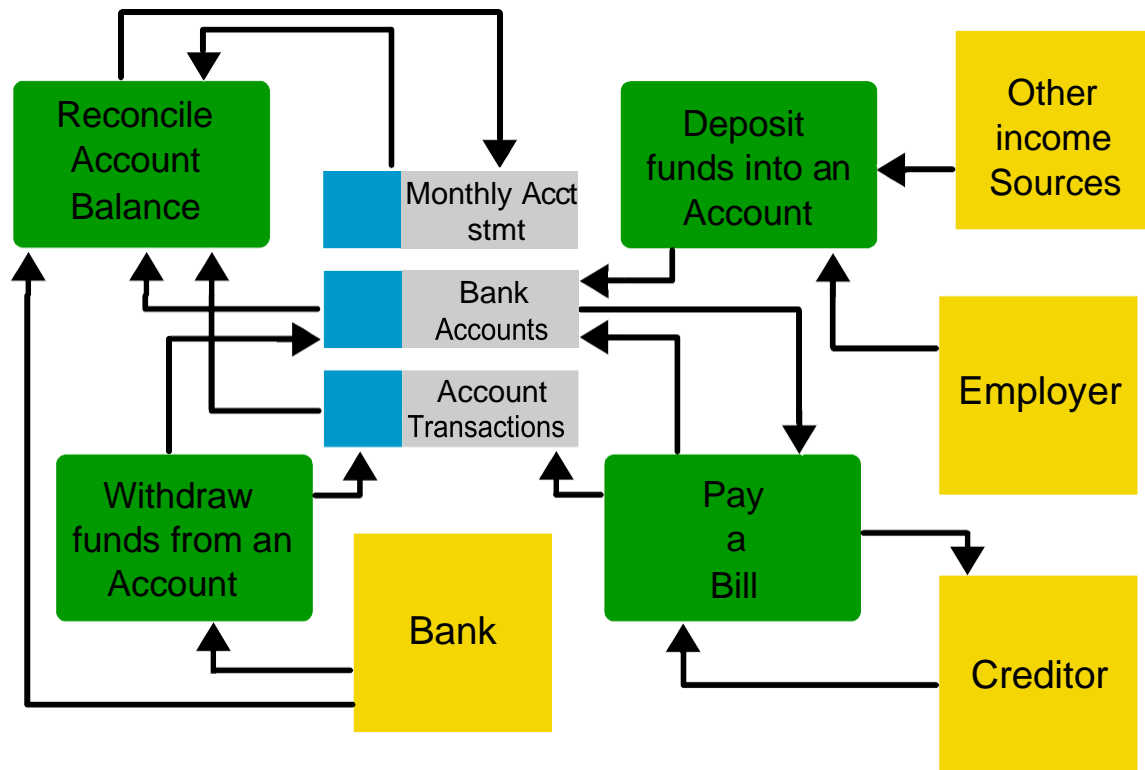
DFD versus Flow Charts

Flow charts are usually used to describe flow of control in a system. It describes control flow in an algorithm. Flow charts are quite detailed. Whereas DFD does not captures control flow information, it just shows the flow of the data in a system. Flow charts show the sequential activities of an algorithm. So, decisions are made, loops or iterations are described. On the other hand, DFD does not show the sequential activities. It just displays the business flow (without sequence among activities). As if you visit an organization, business activities are being performed in parallel. Therefore, DFD does not contain control or sequential activities just data transition is captured.

DFD	Flow Chart
<ul style="list-style-type: none"> • Processes on a data flow can operate in parallel. • Looping and branching are typically not shown. • Each process path may have a very different timing. 	<ul style="list-style-type: none"> • Processes on flowcharts are sequential. • Show the sequence of steps as an algorithm and hence looping and branching are part of flowcharts.

Data Flow Model – Bank Account Management System

In the following, we are presenting a data flow model that describes an accounts management system for a bank. This data flow diagram consists of the following entities



Processes

1. Reconcile account balance
2. Deposit funds into an account
3. Pay a bill
4. Withdraw funds from an account

External agents

1. Bank
2. Creditor
3. Employer
4. Other income sources

Data stores

1. Monthly Account statement
2. Bank accounts
3. Account transactions

Description:

First we shall discuss 'withdraw funds from an account' process. In this process, information about the accounts and account transactions is retrieved (from the data stores) and bank releases the funds. After this, it sends this information to 'reconcile account balance' process which prepares a monthly account statement. In this statement, information regarding bank accounts and account transactions are described. Next is the 'pay a bill' process through which a creditor pays his dues and the corresponding accounts are updated against the cash transaction. A receipt is issued back to the creditor. The fourth process is 'deposits funds in an account' in which an employer deposits salaries of his

employees and the salary information is deposited in the corresponding

bank accounts of the employees. Similarly, income received through other income sources is also received and deposited in the corresponding bank accounts.

Data Flow Modeling

When data flow modeling is used to model a system's functionality, following points need to be remembered

- Data flow model captures the transformation of data between processes/functions of a system. It does not represent the control flow information that is occurring in a system to invoke certain functionality.
- A number of parallel activities are shown in this diagram where no specific sequence among these activities is depicted
- All the previous models that we studied like business process models, state transition diagrams, are used to capture business domain irrespective of their automation.
- However, in data flow models, we represent only those processes which we need to automate as they involve certain computation, processing or transformation of data that can be best implemented using an automated system.
- For example, we may consider a mail desk in an office that receives mail and just forwards it to their respective addressees. In this example, as the mail desk does not process the mail, just forwards it, therefore it does not include any process that need to be automated. Hence, we shall not use data flow diagrams to model this process.
- In nutshell, processes that just move or transfer data (do not perform any processing on that data), should not be described using data flow models.
- Taking the same example, if we modify the scenario such that a mail desk clerk receives the mail, notes it down into a register and then delivers it to their respective addressees then a processing has got involved in this scenario. At least one process is there that can be automated. That is, the recording of mail information into the register. Now we can use a data flow model in which we shall use a data transformation that captures the detail of recording mail information into a register (or a data store). Thus with this addition, it makes sense to use data flow model to capture the details of this process.