Large Scale Systems Design
G52LSS

Lecture 15 – Constructing DFDs

- Steps to Construct DFDs
- Examples of Creating DFDs

Learning outcomes: describe the steps to approach the construction of multi-level DFDs; understand the relationship between DFDs and use case analysis; develop context diagrams; develop Level 0 DFDs; decompose DFDs; engage in discussions when creating DFDs.

Steps to Construct DFDs

1. Build the context diagram

2. Create the level 0 diagram using (if necessary) the requirements definition, use case diagrams, use cases, user stories, summary of business activities, etc.

3. Decompose level 0 processes into level 1 DFDs

4. Decompose level 1 processes into level 2 DFDs and decompose further if needed

5. Balance and validate DFDs to ensure completeness and correctness

Important Considerations When Constructing DFDs

- Keep focus on modelling flow and storage of data
- Identify processes that transform data
- Maintain the balance between diagrams
- Assess the need for decomposition
- Follow conventions to name DFD elements
- Think of data flows as 'data in motion'
- Think of data stores are as 'data at rest'
- Think of processes as actions performed on data
- Do not worry about how data is produced or used by the external entities

Creating the context diagram

- Identify 'actors' or 'user roles' (external entities) from any documentation available
- Identify the information that each external entity sends to the system and receives from the system

Creating the level 0 DFD

- Identify the major functions of the system (processes) from any documentation available
- Identify which processes produce the data sent to the external entities and which processes require the data produced by the external entities
- Identify data flows and data stores and their interaction
- Data flow to data store means 'update', data flow from data store means 'retrieve'

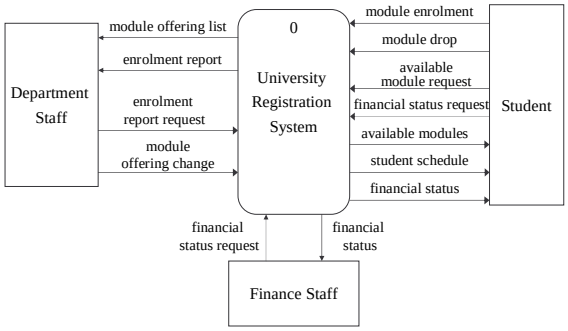
Creating level 1, level 2, etc. DFDs

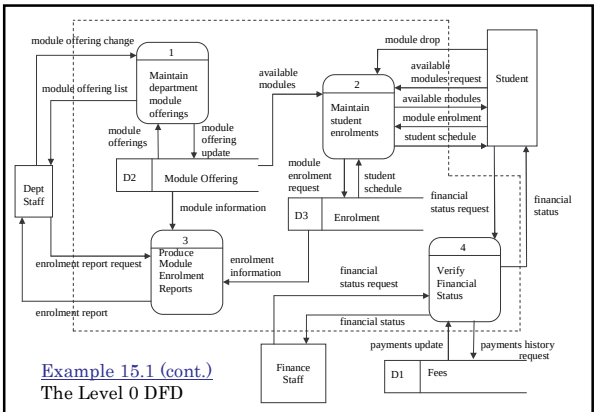
- Functional decomposition in an iterative process which results in DFDs that give more details about a process
- The lowest level DFDs are called primitive DFDs
- Describing a process can help to decide whether it is necessary to decompose the corresponding DFD
- If a process requires to carry out several tasks (logical functions), it is a good candidate for decomposition
- Ensure the conservation of inputs and outputs (data flows) when decomposing processes (balancing)
- Data splits and data joins can be used in the decomposition process to provide more detailed view of how data is transmitted
- Ensure completeness, consistency, no timing dependence, iterative development, creation of primitive DFDs

Example 15.1 Creating a multi-level DFD for this scenario.

University Registration System. The system should enable staff of each academic department to examine the modules offered by their department, add and remove modules, and change the information about them (e.g. the maximum number of students permitted). It should permit students to examine currently available modules, add and drop modules to and from their schedules, and examine the modules for which they are enrolled. Department staff should be able to print a variety of reports about the modules and the students enrolled in them. The system should ensure that no student takes too many modules and that students who have any unpaid fees are not permitted to register (students can verify their fee paying status). Note: assume that a fees data store is maintained by the university's financial office and this data store is accessed by the registration system but the fees data store is not modified by the registration system.

Example 15.1 (cont.) The Context Diagram





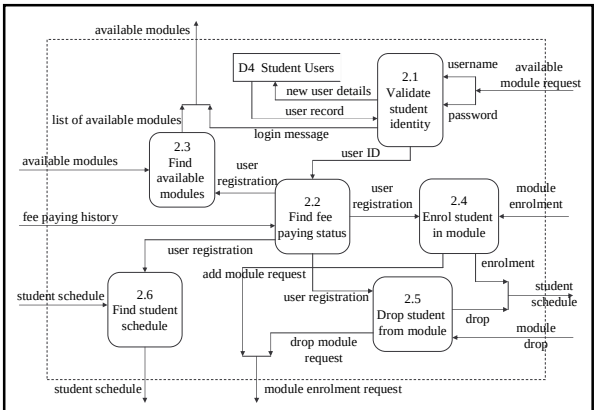
Example 15.1 (cont.)
The Level 0 DFD

Example 15.1 (cont.) Level 1 DFD

Process 2 (maintain student enrolments) of the Level 0 DFD can be decomposed to a number of child processes. The corresponding use case and user story can be used to guide the decomposition.

Examples of possible child processes are:

2.1 Validate student identity
2.2 Find fee paying status
2.3 Find available modules
2.4 Enrol student in module
2.5 Drop student from module
2.6. Find student schedule



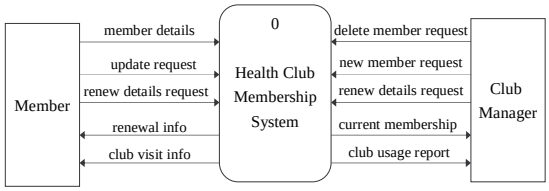
Example 15.2 Creating a multi-level DFD for this scenario.

Health Club Membership System. When members join the health club, they pay a fee for a certain length of time. Most memberships are for one year, but memberships as short as 2 months are available. Throughout the year, the health club offers a variety of discounts on their regular membership prices (e.g. two memberships for the price of one for Valentine's Day). It is common for members to pay different amounts for the same length of membership. The club wants to mail out reminder letters to members asking them to renew their memberships 1 month before their memberships expire. Some members have become angry when asked to renew at a much higher rate than their original membership contract, so the club wants to track the price paid so that the manager can override the regular prices with special prices when members are asked to renew.

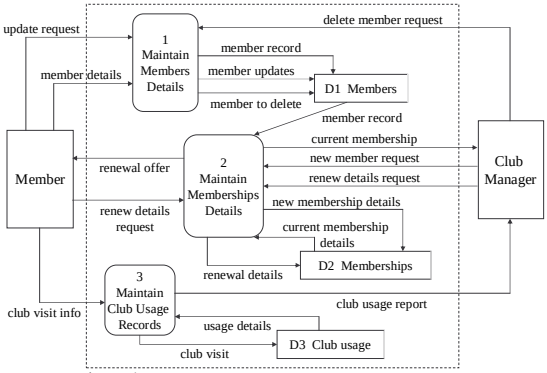
The system must track these new prices so that renewals can be processed accurately. One of the problems in the health club industry is the high turnover rate of members. While some members remain active for many years, about half of the members do not renew their memberships. This is a major problem, because the health club spends a lot in advertising to attract each new member.

The manager wants the system to track each time a member comes into the club. The system will then identify the heavy users and generate a report so that the manager can ask them to renew their memberships early, perhaps offering them a reduce rate for early renewal. Likewise, the system should identify members who have not visited the club in more than a month, so the manager can call them and attempt to re-interest them in the club.

Example 15.2 (cont.) The Context Diagram



Example 15.2 (cont.) The Level 0 DFD



Example 15.2 (cont.) Level 1DFD

From the above Level 0 DFD, a good candidate for further decomposition is process 2: Maintain memberships details. This is because of the various types of memberships, discounts and offers mentioned in the scenario narrative.

Examples of possible child processes are:

2.1 Validate renewal request

2.2 Find membership status

2.3 Find available membership types

2.4 Validate renewal payment

Additional Reading

Chapter 6 of (Dennis et al., 2006)

Chapter 7 of (Kendall and Kendall, 2005)