Faculty of
INFORMATICS
and COMMUNICATION
Central Queensland University
MyInfoCom | Site Map | Search | Feedback

You are here: **InfoCom Homepage** -> **Courses** -> **spr2000** -> **95169** -> **Extra Examples** -> DFD Example

Quick Access

DFD Example

DFD

Purpose

Data Flow Diagrams (DFDs) are a graphical/representation of systems and systems components. They show the functional relationships of the values computed by a system, including input values, output values, and internal data stores. It's a graph showing the flow of data values from their sources in objects through processes/functions that transform them to their destinations in other objects. Some authors use a DFD to show control information, others might not. A DFD can be seen as a method of organizing data from its raw state.

Note: As starting students in Systems Analysis and Design I wouldn't expect you handle more than a context diagram (initial project scope) or a system diagram (in text) more commonly known as a first level DFD (Level 0 DFD)

Steps for Developing DFDs

1. Requirements determination
2. Divide activities
3. Model separate activities
4. Construct preliminary context diagram
5. Construct preliminary System Diagram/ Level 0 diagrams - (As far as I expect for starting students to get.)
6. Deepen into preliminary level n diagrams (primitive diagrams in text)
7. Combine and adjust separate level 0-n diagrams
8. Combine level 0 diagrams into definitive diagram
9. Complete diagrams

Step 1: Requirements determination

This is the result of the preceding phases. Through different techniques, the analyst has obtained all kinds of specifications in natural language. This phase never stops until the construction of the DFD is completed. This is also a recursive phase. At this moment, he should filter the information valuable for the construction of the data flow diagram. He should order the different data by:

Initial definition of :

1.
2. 1. entities

3.
4. 2. related activities
5.
6. 3. data flows
7.

Names

Constraints

Data stores

Incomplete ---> Get more Information

Step 2: Divide activities
Hereby, the analyst should separate the different activities, their entities and their required data. The completeness per activity can be achieved by asking the informant the textual specification with the lacking components in the activity.

Step 3: Model separate activities
The activities have to be combined with the necessary entities and data stores into a model where input and output of an activity, as well the sequence of data flows can be distinguished. This phase should give a preliminary view of what data is wanted from and given to whom.

Step 4: Construct preliminary context diagram
The organization-level context diagram is very useful to identify the different entities. It gives a steady basis in entity distinction and name giving for the rest of the construction. From here on, the analyst can apply his top-down approach and start a structured decomposition. This is the process of organizing the diagrams into a hierarchy of increasingly detailed views of processes.

Step 5: Construct preliminary level 0 diagrams
The overview, or parent, data flow diagram shows only the main processes. It is the level 0 diagram. This diagram should give a 'readable' overview of the essential entities, activities and data flows. An over-detailed level 0 diagram should generalize appropriate processes into a single process.

Step 6: Deepen into preliminary level n diagrams
This step decomposes the level 0 diagrams. Each parent process is composed of more detailed processes, called child processes. The most detailed processes, which can not be subdivided any further, are known as functional primitives. Process specifications are written for each of the functional primitives in a process.

Step 7: Combine and adjust level 0-n diagrams
During the structured decomposition, the creation of the different processes and data flows most often generate an overlap in names, data stores and others. Within this phase, the analyst should attune the separate parent and child diagrams to each other into a standardized decomposition. The external sources and destinations for a parent should also be included for the child processes.

Step 8: Combine level 0 diagrams into a definitive diagram
The decomposition and adjustment of the levelled diagrams will most often affect the quantity and name giving of the entities.

Example: Often the need for an entity 'Time' will only appear during the formation of levelled diagrams and needs therefore to be added in the definitive diagram.

Step 9: Completion
The final stage consists of forming a structured decomposition as a whole. The input and output shown should be consistent from one level to the next.
The result of these steps, the global model, should therefore obey all the decomposition rules.

Example: It should be possible to distinguish the separate data stores for use in Entity-Relationship Modelling

Some DFD rules

Overall:	1. Know the purpose of the DFD. It determines the level of detail to be included in the diagram.
	2. Organize the DFD so that the main sequence of actions reads left to right and top to bottom.
	3. Very complex or detailed DFD's should be levelled.
Processes:	4. Identify all manual and computer processes (internal to the system) with rounded rectangles or circles.
	5. Label each process symbol with an active verb and the data involved.
	6. A process is required for all data transformations and transfers. Therefore, never connect a data store to a data source or destination or another data store with just a data flow arrow.
	7. Do not indicate hardware or whether a process is manual or computerized.
	8. Ignore control information (if's, and's, or's).
Data flows:	9. Identify all data flows for each process step, except simple record retrievals.
	10. Label data flows on each arrow.
	11. Use data flow arrows to indicate data movement, not non-data physical transfers.
Data stores:	12. Dot not indicate file types for data stores.
	13. Draw data flows into data stores only if the data store will be changed.
External entities:	14. Indicate external sources and destination of data, when known, with squares.
	15. Number each occurrence of repeated external entities.
	16. Do not indicate persons or places as entity squares when the process is internal to the system.

An Example
Draw the DFD for a distance education university. The enrolment process works as follows:
Students send in an application form containing their personal details, and their desired course
The university checks that the course is available and that the student has necessary academic qualifications.
If the course is available the student is enrolled in the course, and the university confirms the enrolment by sending a confirmation letter to the student.
If the course is unavailable the student is sent a rejection letter.

1. Read the problem description carefully looking for:

    - people/organisations/things that supply information to or use information from the system => external entities (EE)

- actions/doing words/verbs => Processes (P)
- movement/exchange of information/data between external entities to processes, and processes to processes => data flows (DF)
- store/record information/data => data stores(DS)

2. It often helps to walk through the system in its logical sequence; eg starting with an external entity (source), add data flows, processes and data stores as the data provided by the entity is manipulated by the system.

- A student (EE) sends in an application form (DF) containing their personal details, and their desired course
- The university checks (P) that the course is available.
- If the course is available the student is enrolled (P) in the course, and the university confirms (P) the enrolment by sending a confirmation letter (DF) that they are registered for the course to the student.
- Or if the course is unavailable the student is sent a rejection letter (DF).

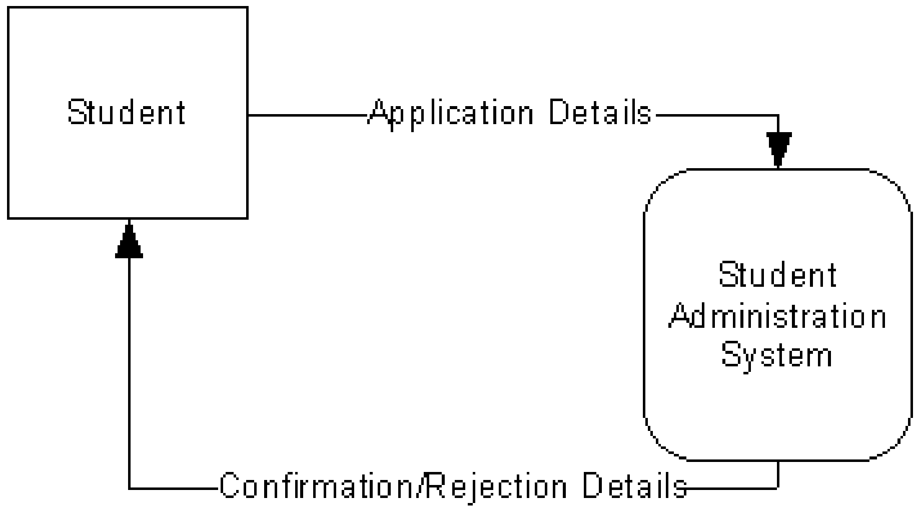
Note: The university, or more specifically the administration section, is the system being modelled, it is not an EE.

Context diagram

- Highest level DFD.
- Has data flows, external entities, one process (system in focus) and no data stores.
- Shows the system boundary and interactions with external entities.

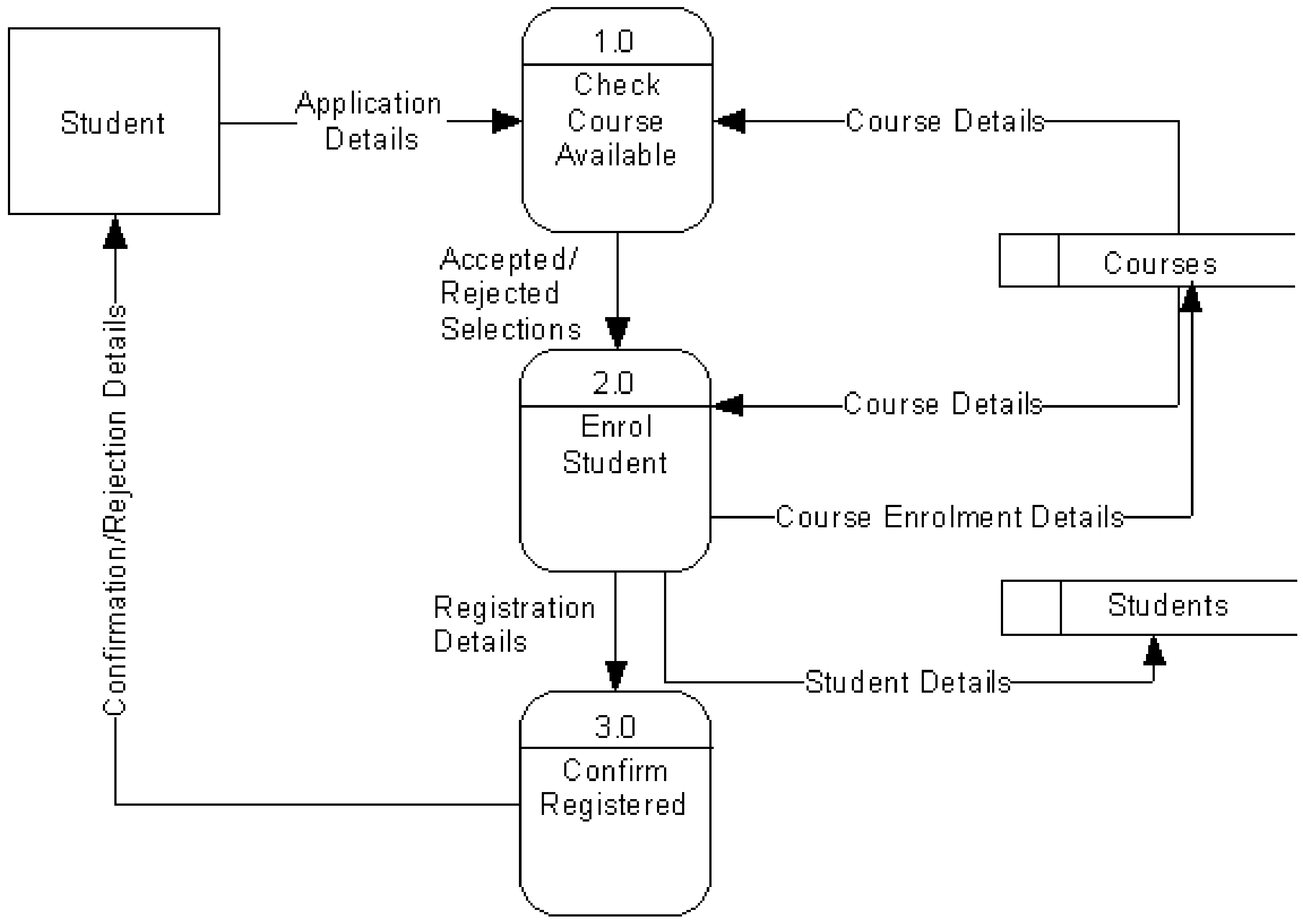
In this case:

External entity	- Student
Process	- Student Administration process application
Data Flows	- Application Form, Confirmation/Rejection Letter



System/Level 0 DFD

External entity - Student
Processes - Check available, Enrol student, Confirm Registration
Data Flows - Application Form, Course Details, Course Enrolment Details, Student Details, Confirmation/Rejection Letter
Data Stores - Courses, Students.



This System/Level 0 DFD raises some questions:

Q. The Data Store *Courses* has only data flows entering it, how does the data get stored in the first place?
A. This DFD is part of a larger, higher level DFD that models more than just the enrolment process. There must be another DFD that stores course details, eg the university's course development process.

Q. The process *Enrol Student* has many different data flows entering into and leaving it, how can we model this process in more detail?
A. Develop a more detail, lower level DFD for this process, that shows the processes that make up this process

You are here: **InfoCom Homepage** -> **Courses** -> **spr2000** -> **95169** -> **Extra Examples** -> DFD Example

Edit **Last Modified:** Tue Nov 21 10:30:12 2000 **by** webmaster.
Contact Details