Chapter #09 (SE)

Q4 Long Architectural mapping using data flow

Mapping using Data flow Diagrams (DFDs)

A mapping technique, called Structured Design, is often characterized as a data flow design model, because it provides a convinent transition from a data flow diagram to software architecture

The transition of information and data flow to a program structure is accomplished by processing following six steps:

(i) The type of information flow is established.

(ii) Flow boundries are indicated

(iii) The DFD (data flow diagram) is mapped into program structure.

(iv) Control hierarchy is defined

(v) The resultant structure is refined using design measure

(vi) The architecutural description is refined & elaborated.

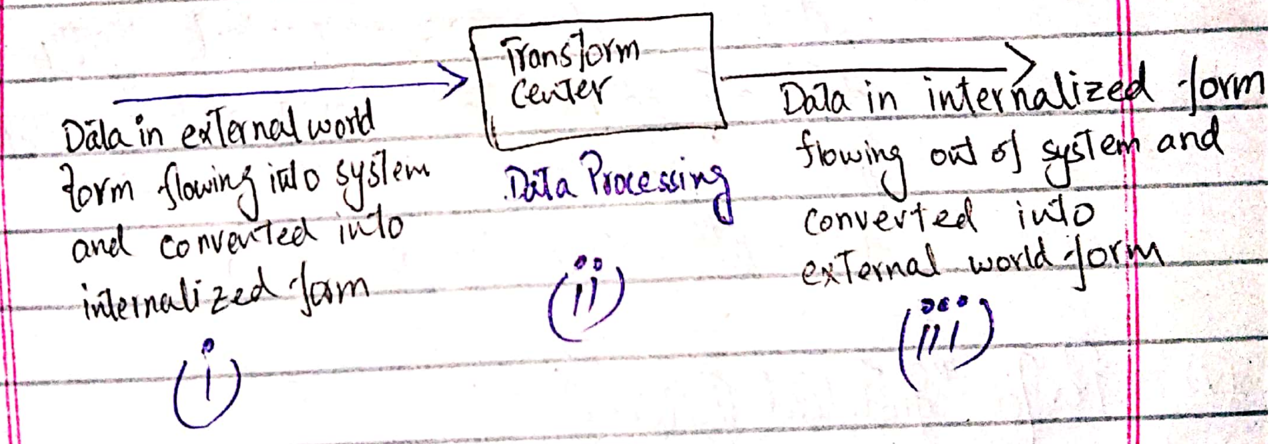
In order to perform mapping, following steps are ~~taken~~ performed.

(i) The type of info... must be determined

(ii) Data flows into the system along an incoming flow path, where it is transformed from its form into internalized form.

Once, it has been internalized, it is processed at Transform center

(iii) Finally, it flows out of the system along an outgoing flow path that transform data into external world form



Data in external world form flowing into system and converted into internalized form

Data in internalized form flowing out of system and converted into external world form

Transform Mapping:

it is a set of ~~predefined~~ designed steps that allows a DFD to be mapped into a specific architectural design.

Steps

(i) Review the fundamental system model

involves evaluation of system specification and software requirments

(ii) Review & Refine (DFD) for the software

info obtained from analysis model is further refined to produce details

(iii) Determine whether DFD has transform characteristics or transaction characteristics.

if DFD has obvious transaction characteristic then a different design mapping is recomended.

(iv) Isolate the transform center by specifying incoming and outgoing flow boundries

different designers may select slightly different points in the outgoing and incoming flow as boundries that can provide ~~differ~~ alternative design solution

(v) Perfor "first-level factoring"

Factoring results in

top lvl module: performs decision making
Middle L-vl " : " some control
low-lvl " : " input, output, computation

(vi) Perform "second-level factoring"

Second-lvl factoring is accomplished by mapping individual transforms of a DFD into appropriate modules within the architecture

(vii) Refine the first-iteration architecture using design heuristic for improved software quality.

First-iteration architectures can be refined by applying concepts of module independence.

Scanned with CamScanner