Creating An Architectural Design

What is software architecture?

Software architecture models the structure of the system, and how the data and components collaborate with one another. The structure refers to software components and their relationships. The externally visible properties of components is specified. The work product of the architecture design is the architectural model and includes data architecture and program structure.

Architecture design consists of → Data Design, Architectural Design

Why is architecture important?

- Representation of software architecture acts a base or reference for communication among stakeholders
- It encodes early design decisions that are easy to evaluate and correct than later which is during implementation
- It is a high-level and thus understandable representation of the entire system (parts and relationships). The parts are called components.

What is the purpose of data design?

Its purpose is to translate data objects from analysis model into data structures at component level (design). Data design may also result in database design at the application level – architectural-level. It may also lead to data warehouse design.

what's the purpose of architectural design?

Its purpose is to identify/design the architectural style and architectural patterns for a system

Architectural style determines/applies the structure of the entire system like the components, connectors, constraints on how to integrate components and semantic models

Architectural patterns apply to solving specific (structural/behavioral) issues that occur commonly in a given architectural style.

what are the various types (taxonomy) of architectural styles?

- Data-Centered
- Data-Flow eg: pipes & filters
- call and return architecture
- Layered architecture
- Object-Oriented architecture

what are some types of architectural patterns?

- Concurrency: A pattern that provides solutions to simulate parallelism in an system eg: task scheduler pattern, OS process mgmt pattern
- Persistence: A pattern that solves how to store/handle persistent data eg: DBMS pattern, application-level persistence pattern
- Distribution: This pattern provides solutions to the issue of communication of distributed processes within a system. eg: broker pattern.

How does one go about building/constructing/modeling an architectural design?

I. Model Architectural context.
II. Define the archetypes — abstract building blocks of architectural design
III. Refine architecture into components
IV. Describe instantiations of the system

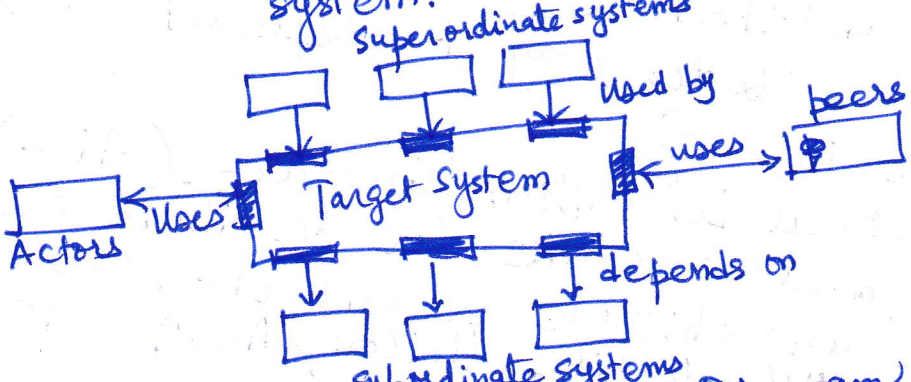
Archetype is a class or a pattern that represents core abstraction that is critical to design of an architecture of a target system.

What is architectural context?

Architechtural context represents how the software interacts with entities lying outside the system

The types of systems/entities that interact with target system are:

— Superordinate: target system is part of a (superordinate) bigger system
— Subordinate: target system uses a subordinate system (subsystem) to achieve a task
— Peer-level systems: these systems collaborate with target system as equals producing and consuming information
— Actors: these systems (people, devices) interact with target system.



Architectural Context Diagram

What are the sources of architecture components?

1. Application domain:
2. Interface domain: eg: GUI components etc.
3. Infrastructure domain: eg memory mgmt, communication, database, task mgmt. etc

Name some techniques to assess alternative architectural designs?

1. Architecture Trade-off Analysis Method (ATAM)
2. Software Architecture Analysis Method (SAAM)

We asses a number of architectural designs based on user requirements & future (2-3 yrs) changes and choose most appropriate architectural design.

How does one assess architectural complexity?

Assessing architectural complexity involves assessing
- Sharing dependencies
- Flow dependences &
- Constraint dependencies among components of an architecture

What is a better way to represent and describe an soft. architecture?

Architectural Description Language (ADL) provides the syntax and semantics for describing an architecture in an unambiguous & understandable fashion.
eg: UML, Rapide, Unicon, Aesop etc.

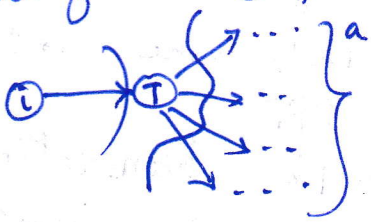
What are the various types of data flow assuming Structured Analysis and Design technique?

There are two types of data flows considered in SAD

1) Transform flow: which basically consists of incoming flow (input-i), transform center (processing-p) and outgoing flow (output-o).

2) Transaction flow: In this kind a single data item triggers other data flow along many paths

This has incoming flow (input-i), transaction centre (T) and outgoing (action paths) flows.



After performing flow analysis, one has to map the DFD (Data Flow Diagram) bubbles (transforms) into an architecture consisting of modules (rectangles) (Here we assume call and return architecture)

What are the mapping techniques one you have a

I. Transform mapping: this technique maps transforms into specific architectural style. Its steps:

a. Review fundamental system model
b. Review and refine data flow diagrams for the software
c. Determine whether DFD has transform or transaction flow characteristics
d. Isolate transform center by specifying incoming and outgoing paths (flows) boundaries
e. Perform first-level factoring
f. Perform second-level factoring
g. Refine the first-iteration architecture using design heuristics for improved software quality.

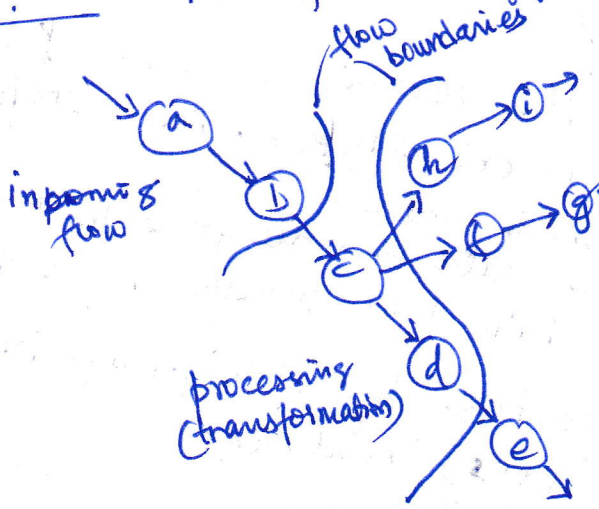
6 Of

II. Transaction Mapping: Once you identify transaction flow in data transformation, this technique maps it into a specific architectural style.

The steps are as follows:

a. Review the fundamental system model
b. Review and refine data flow diagrams for the software
c. Determine both transforms and transaction flows in DFD
d. For transaction flow identify transaction center and flow characteristics along action paths
e. Map the transforms into modules for transaction flows (reception path, transaction center (dispatcher) & action paths)
f. factor and refine transaction structure and structure of each action paths

eg: Transform mapping (after level 0 DFD)



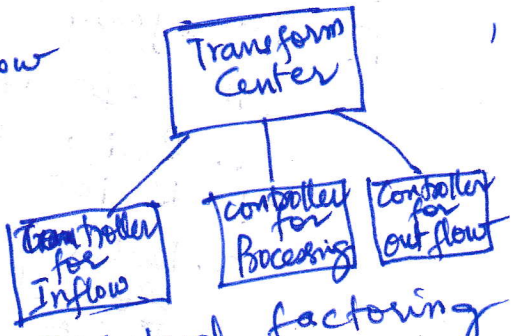


Fig: First-level factoring

○ : Data transform
□ : Component/Module

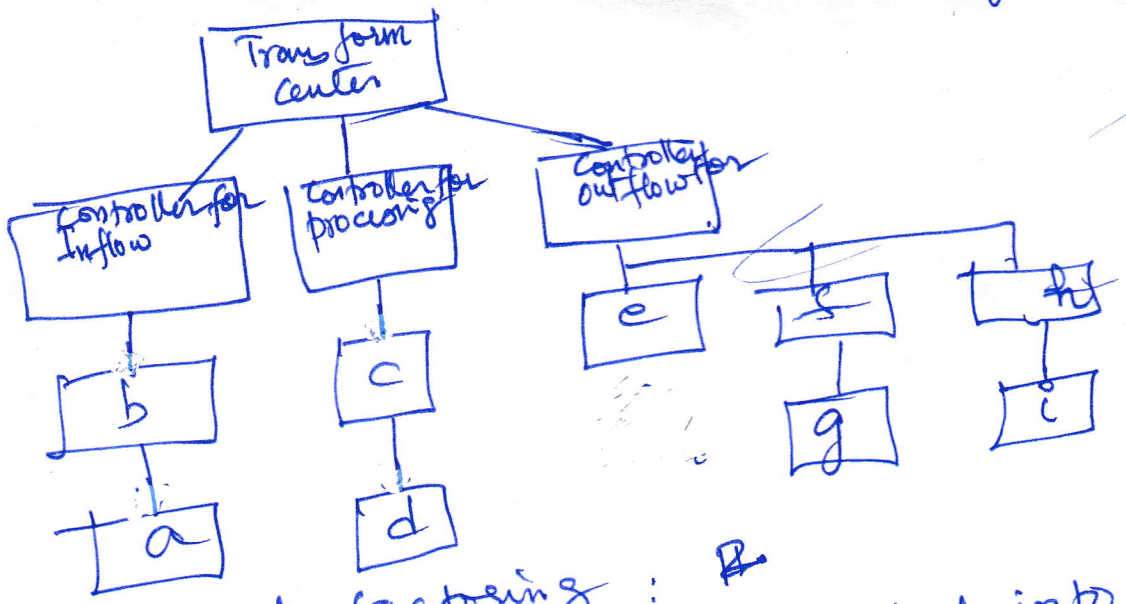
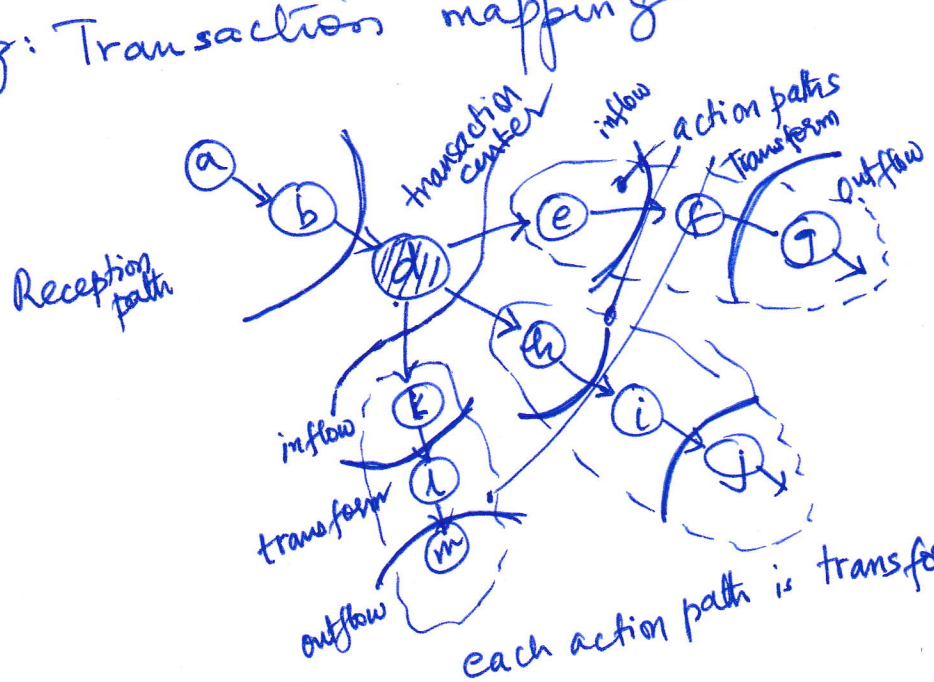
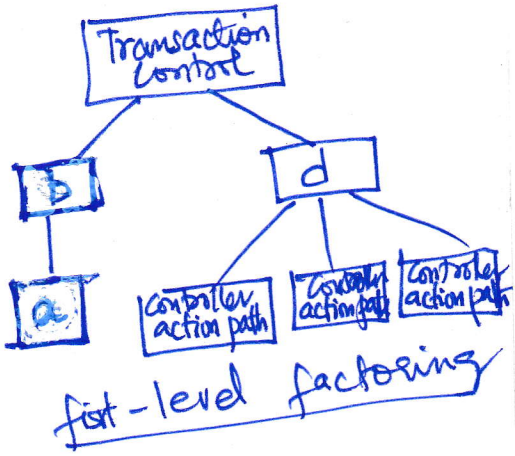


Fig: Second-level factoring :

Here each transform of DFD is mapped into a module. and the architecture (Call and Return)

eg: Transaction mapping



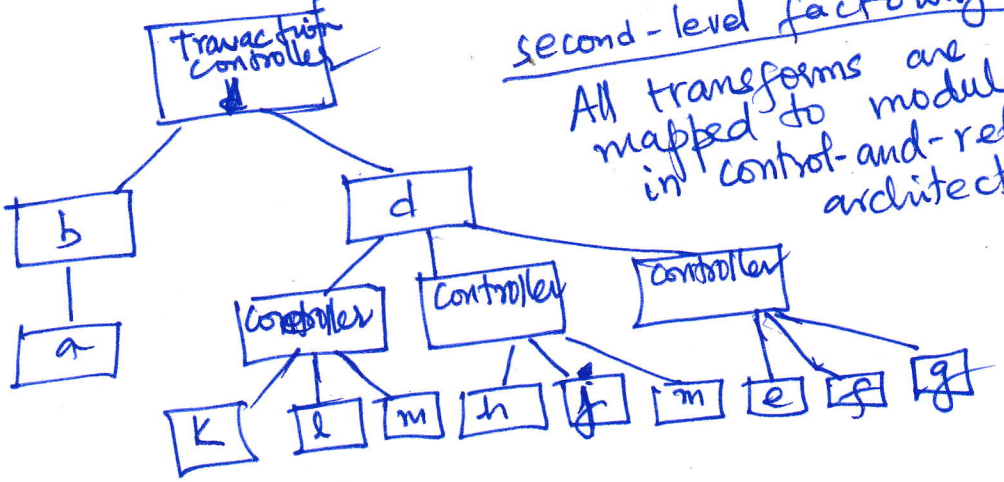


fist-level factoring

each action path is transform flow

○ : Data transforms

□ : component module



second-level factoring

All transforms are mapped to modules in control-and-return architecture