Macro Architecture Design

What is the importance of software architecture?

- To reduce risk, and time-to-market
- To increase predictability, reliability and quality
- To provide early identification of potentially very large reuse opportunities
- To leverage experience by using the architecture to document design knowledge and to train

What is the importance of software architecture evaluation?

- To make sure one is using "good" ones
- To ensure good fit to requirements
- To ensure implementability of system

Architectural Views

- **Common and useful software structures:**
  - modular structure
  - conceptual or logical structure
  - process structure or coordinating structure
  - physical structure
  - Uses structure
  - call structure
  - data flow
  - control flow

System's architecture forces

- What forces shape the archicture?
  - Sytem´s **built-time** properties,
  - **run-time** properties
  - business **qualities**

System built-time requirements

- describe the properties and "behavior" the system should exhibit while is not running.
- These properties are not visible while the system is running; only when system in down and the hood is up.
  - maintainability,
  - flexibility,
  - extensibility,
  - configurability,
  - reusability
  - modifiability
  - portability
  - integrability
  - testability

System´s run-time requirements

- safety,
- reliability,
- performance,
- throughput,
- effectiveness,
- availability,
- fuctionality
- security
- usability (learnability, efficiency,memorability,error avoindance, error handling, satisfaction)

System´s Business requirements

- Time to market
- Cost
- Projected life-time of the system

- Sytem´s built-time properties, run-time properties and business qualities DO shape the systems architecture

Generic change cases in a system

- Port to a different hardware platform and OS
- Change presentation package without changing look and feel
- Change look and feel of user presentations
- Add a new actor who has a different pattern of using the system
- Add new services to the system
- A new kind of system state element is added to the domain that the system must handle
- Change the DB
- Configure multiple systems with different set of functionalities
- Change from a single user implementation to physically distributed, multiuser implementation.

Sources for changes to mine

- Changes anticipated for this system or experienced by similar systems
- Changes anticipated by mining requirements specifications for areas of uncertainty or ambiguity
- Decreases in functionality brought on because of a management decision to field a subset of the system

Sources for changes to mine

Keep and use a:

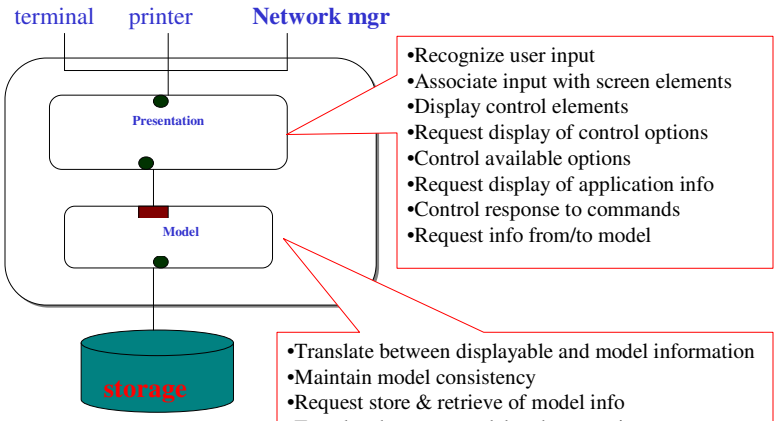
NNB!!

To measure goodness of an architectural design

Generic responsibilities for user driven systems

- Recognize user input
- Associate input with screen elements
- Display control elements
- Display application elements
- Request display of control options
- Control available options
- Request display of application information
- Control response to commands
- Request information from, provide information to model
- Translate between displayable and model information
- Maintain model consistency
- Request store and retrieve of model information
- Translate between model and storage medium

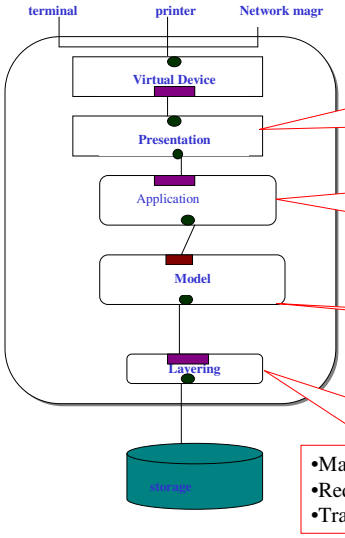
A Basic Template Architecture



Evaluation on the face of Change

- Port to a different hardware platform and OS
- Change presentation package without changing look and feel
- Change look and feel of user presentations
- Add new actor who has a different pattern of using the system
- Add new services to the system
- A new kind of system state element is added to the domain that the system must handle
- Change the DB
- Configure multiple systems with different set of functionalities
- Change from a single user implementation to physically distributed, multiuser implementation

Refinement 1 to Proposed Architecture



•Recognize user input
•Associate input with screen elements
•Display control elements
•Display application elements

•Request display of control options
•Control available options
•Request display of application info
•Control response to commands
•Request info from/to model

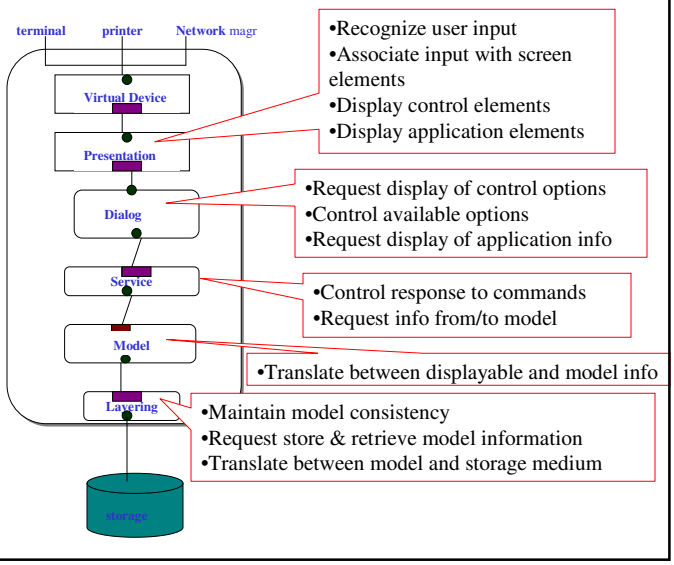
•Translate between displayable and model info

•Maintain model consistency
•Request store & retrieve model information
•Translate between model and storage medium

Evaluation on the face of Change

- Port to a different hardware platform and OS
- Change presentation package without changing look and feel
- Change look and feel of user presentations
- Add a new actor who has a different pattern of using the system
- Add new services to the system
- A new kind of system state element is added to the domain that the system must handle
- Change the DB
- Configure multiple systems with different set of functionalities
- Change from a single user implementation to physically distributed, multiuser implementation

Refinement II of proposed architecture



Evaluation on the face of Change

- Port to a different hardware platform and OS
- Change presentation package without changing look and feel
- Change look and feel of user presentations
- Add a new actor who has a different pattern of using the system
- Add new services to the system
- A new kind of system state element is added to the domain that the system must handle
- Change the DB
- Configure multiple systems with different set of functionalities
- Change from a single user implementation to physically distributed, multiuser implementation

Generic architecture template

- **Key aspect of any successful system architecture**
  - it has enough kinds of subsystems to gracefully support all of the anticipated changes.
  - Behaviors are allocated appropriately to the subsystems.
    - Cohesion
    - encapsulation
  - Design with quite small components

Useful set of subsystems

- **Device drivers and virtual devices**
  - Provided for each port that crosses the system
  - Provide portability and isolation from peripherals and platform.

Useful set of subsystems

- **Presentation (View) subsystems**
  - Provided for each kind of display required by actors
  - translated between internal data representation and external form.
  - Multiple presentation may be needed, even for single port.
  - The application side protocol for the presentation must hide specifics of it from other application subsystem.
  - UI presentations are well known
  - Presentations for hard-copy terminals, dbs, network channels are just as important.

Useful set of subsystems

- **Dialogue controller subsystems.**
  - Provided for each kind of user or interaction with system.
  - Responsible for managing content and sequence of dialogue between system and actor.
  - Invokes system services to carry out commands.

Useful set of subsystems

- **Service subsystems**
  - ❑ Provided for each unit of growth and change as indicated in the change cases.
  - ❑ Units of functional growth, change and configuration
  - ❑ Encapsulate functionality that may be added or removed from system.

Useful set of subsystems

- **Model subsytem**
  - – Subclasses which model the problem
  - – independence of presentation
  - – independence of database management
- **Database Layering subsystem**
  - – translate between model and storage system

How to evaluate alternative architectures

- Measured on ability to satisfy external requirements that emerge from system as a whole
- evaluate built-time requirements
- use change cases