Architectural Design

Outline

- Definition of an Architecture.
- Architecture styles :
  - Data-centered architectures,
  - Data flow architecture
  - Call and return architectures
  - Object-oriented architectures
  - Layered architectures.
- Architecture Design :
  - Component
  - Properties
  - Interaction
  - Archetype
  - Architecture context diagram
  - Architecture diagram
- Architecture design using dataflow.

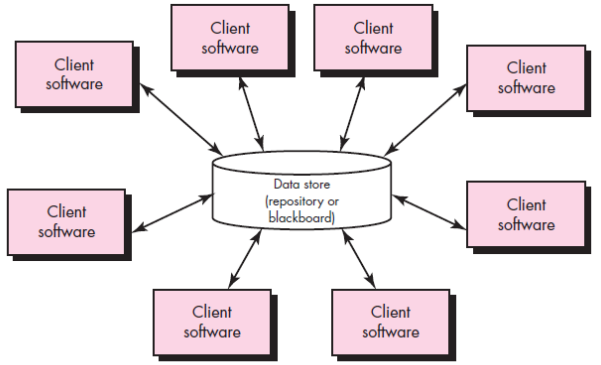
Architecture

- SA definition : The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them.
- The architecture is a representation that enables you to
  - (1) analyze the effectiveness of the design in meeting its stated requirements,
  - (2) consider architectural alternatives at a stage when making design changes is still relatively easy, and
  - (3) reduce the risks associated with the construction of the software.
- Architecture design focus on representation of the structure of software components, their properties, and interactions.

Architecture Styles

- An architectural style is to establish a structure for all components of the system.
- Each style describes
  - (1) a **set of components** (e.g., a database, computational modules) that perform a function required by a system;
  - (2) a **set of connectors** that enable "communication, coordination and cooperation" among components;
  - (3) **constraints** that define how components can be integrated to form the system; and
  - (4) the overall **properties of a system**

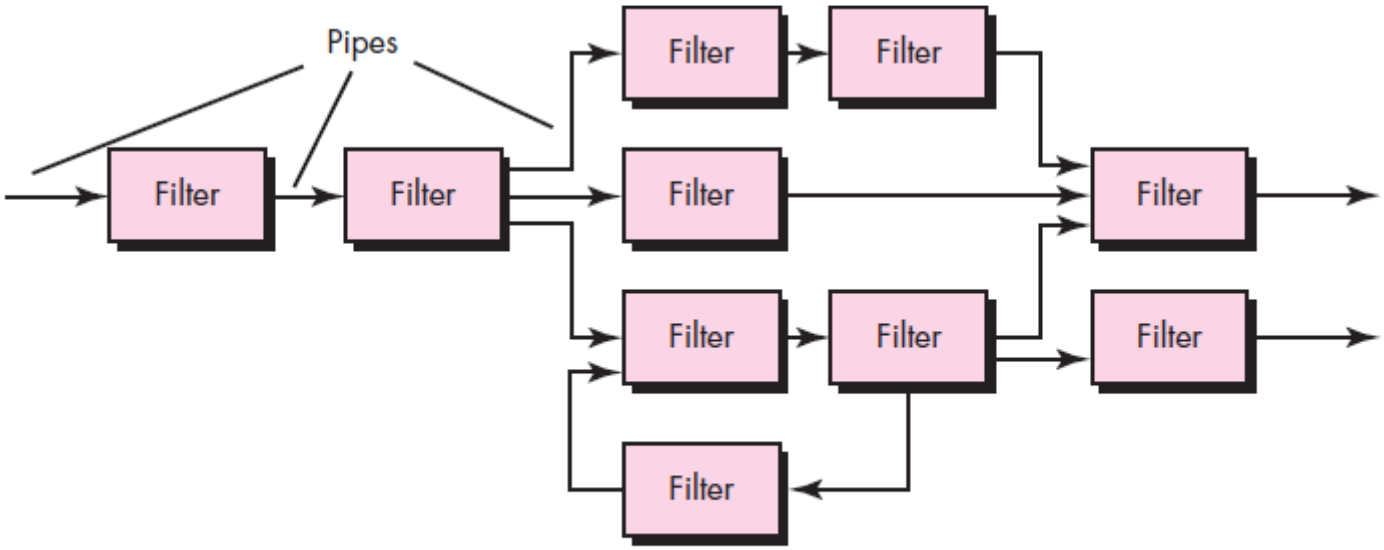
Architectural styles



- **Data-centered architectures.** A data store (e.g., a file or database) resides at the center of this architecture and is accessed frequently by other components that update, add, delete, or otherwise modify data within the store.
- Data-centered architectures promote *integrability. That is, existing* components can be changed and new client components added to the architecture without concern about other clients (because the client components operate independently).

Architectural styles

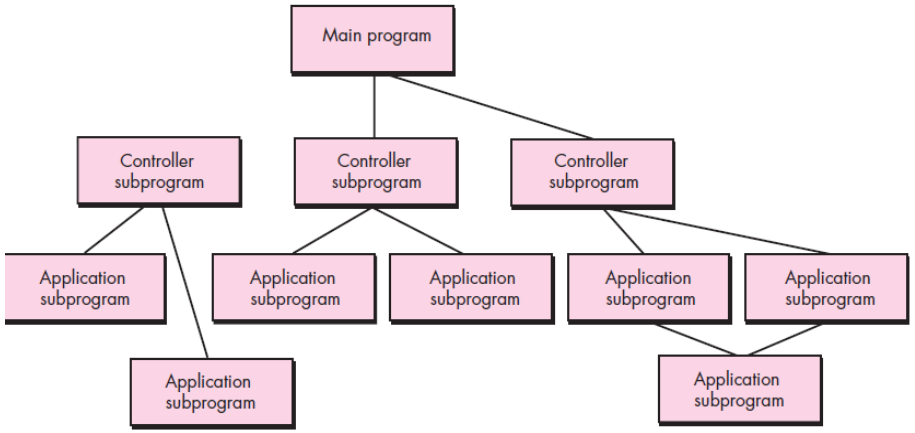
- **Data flow architecture** : A pipe-and-filter pattern has a set of components, called *filters, connected by pipes that transmit data from one component to the next.*
- *Each* filter works independently of those components upstream and downstream, is designed to expect data input of a certain form, and produces data output (to the next filter) of a specified form.
- However, the filter does not require knowledge of the workings of its neighboring flters.



Pipes and filters

Architectural Styles

- **Call and return architectures.** This architectural style enables you to achieve a program structure that is relatively easy to modify and scale. A number of substyles exist within this category:
- *Main program/subprogram architectures. This classic program structure* decomposes function into a control hierarchy where a “main” program invokes a number of program components that in turn may invoke still other components. Figure illustrates an architecture of this type.
- *Remote procedure call architectures. The components of a main* program/subprogram architecture are distributed across multiple computers on a network.

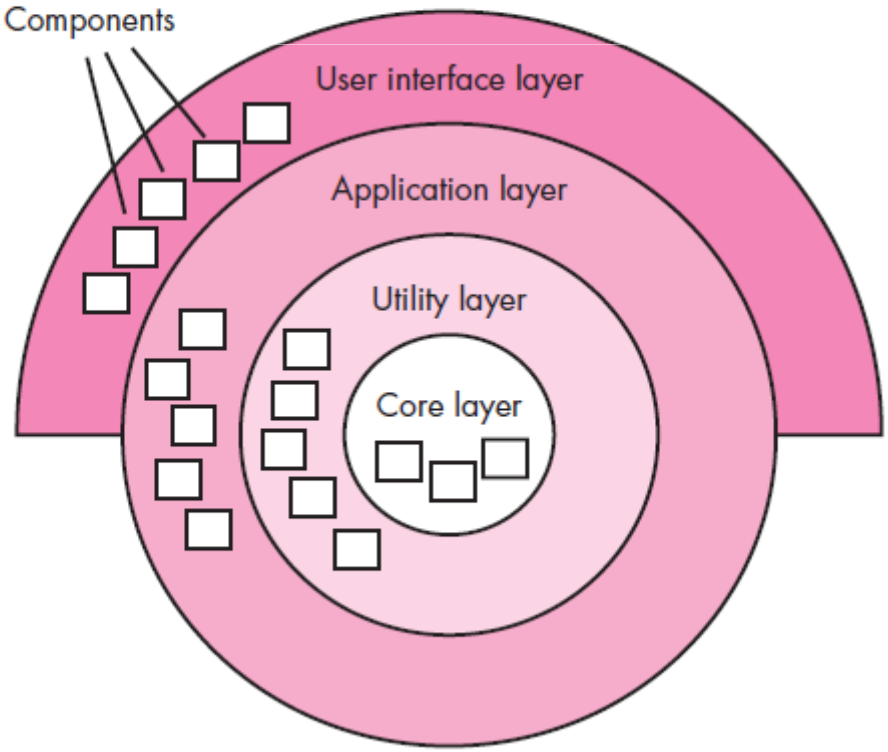


Architectural Styles

- **Object-oriented architectures. The components of a system encapsulate data** and the operations that must be applied to manipulate the data. Communication and coordination between components are accomplished via message passing.

Architectural Styles

- **Layered architectures.** A number of different layers are defined, each accomplishing operations that progressively become closer to the machine instruction set. At the outer layer, components service user interface operations. At the inner layer, components perform operating system interfacing. Intermediate layers provide utility services and application software functions.



Architectural style

- Organization and Refinement (detailed analysis of the architecture.)
  - to establish a set of design criteria that can be used to assess an architectural design that is derived.
- Some of criteria to access are as follows:
  - How do components transfer control within the system?
  - How is control shared among components?
  - What is the control topology (i.e., the geometric form that the control takes)?
  - Is control synchronized or do components operate asynchronously?
  - How are data communicated between components?
  - Do data components (e.g., a blackboard or repository) exist, and if so, what is their role?
  - How do data and control interact within the system?

Architecture design

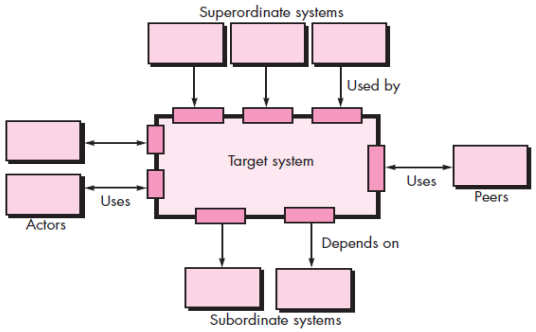
- Component – can be simple as a program module or a OO class. Also, include DB and middleware configuration of a network of clients and servers.
- Properties – characteristics necessary to understand the components interaction with other components.
- Architectural level interaction – relationship between component procedure call from one module to another or a complex as a DB access protocol.

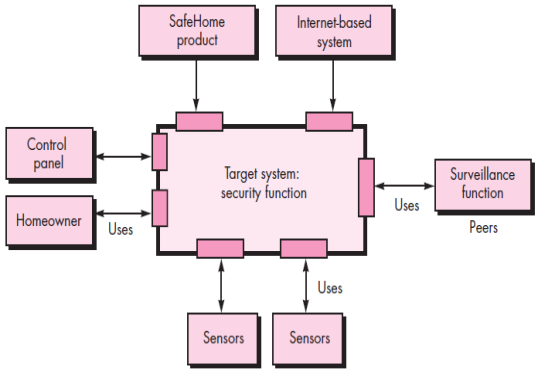
Architecture Design

- The designer specifies the structure of the system by defining and refining software components that implement each archetype.
- An *archetype is an abstraction (similar to a class) that represents* one element of system behavior. The set of archetypes provides a collection of abstractions that must be modeled architecturally if the system is to be constructed, but the archetypes themselves do not provide enough implementation detail.
- This process continues iteratively until a complete architectural structure has been derived.

Architecture Design

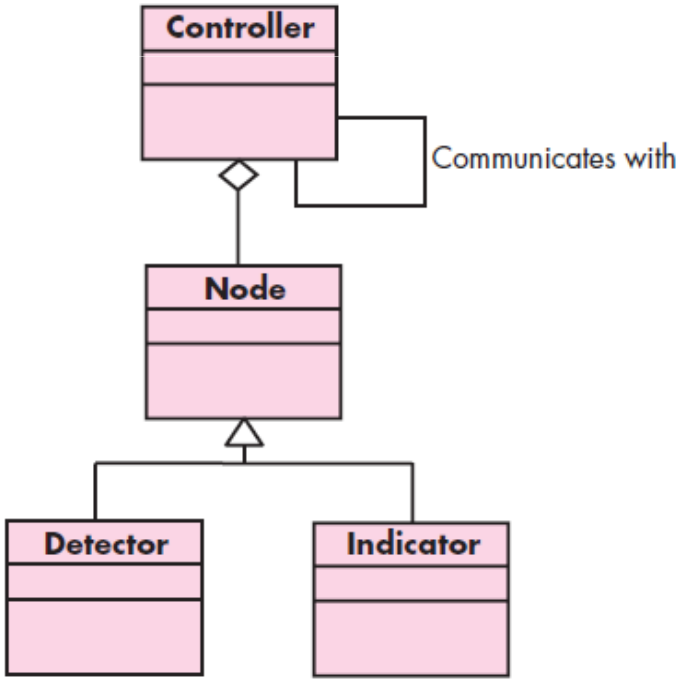
- **Representing the System in Context :** software architect uses an *architectural context diagram* (ACD) to model the manner in which software interacts with entities external to its boundaries.
- Each of these external entities communicates with the target system through an interface (the small shaded rectangles). External entities are represented as
  - *Superordinate systems—those systems that use the target system as part of* some higher-level processing scheme.
  - *Subordinate systems—those systems that are used by the target system and* provide data or processing that are necessary to complete target system functionality.
  - *Peer-level systems—those systems that interact on a peer-to-peer basis (i.e.,* information is either produced or consumed by the peers and the target system).
  - *Actors—entities (people, devices) that interact with the target system by* producing or consuming information that is necessary for requisite processing.





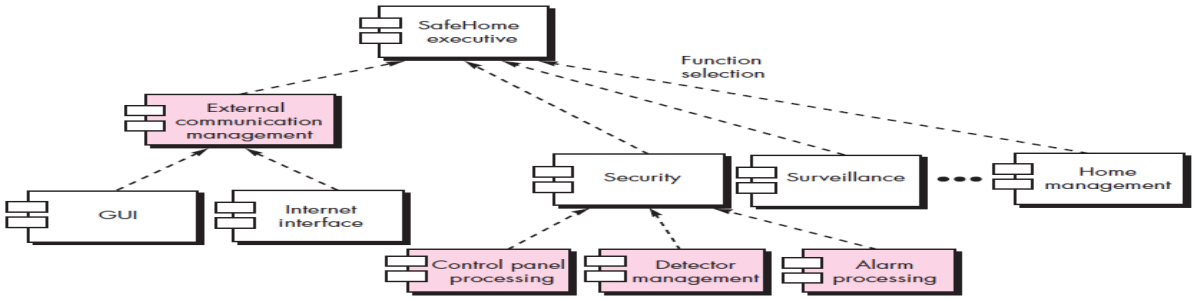
Architecture design

- Defining Archetypes : An *archetype is a class or pattern that represents a core abstraction that is critical to* the design of an architecture for the target system.
- Archetypes can be derived by examining the analysis classes defined as part of the requirements model.
- For example, might define the following archetypes : Node, Detector, Indicator, Controller



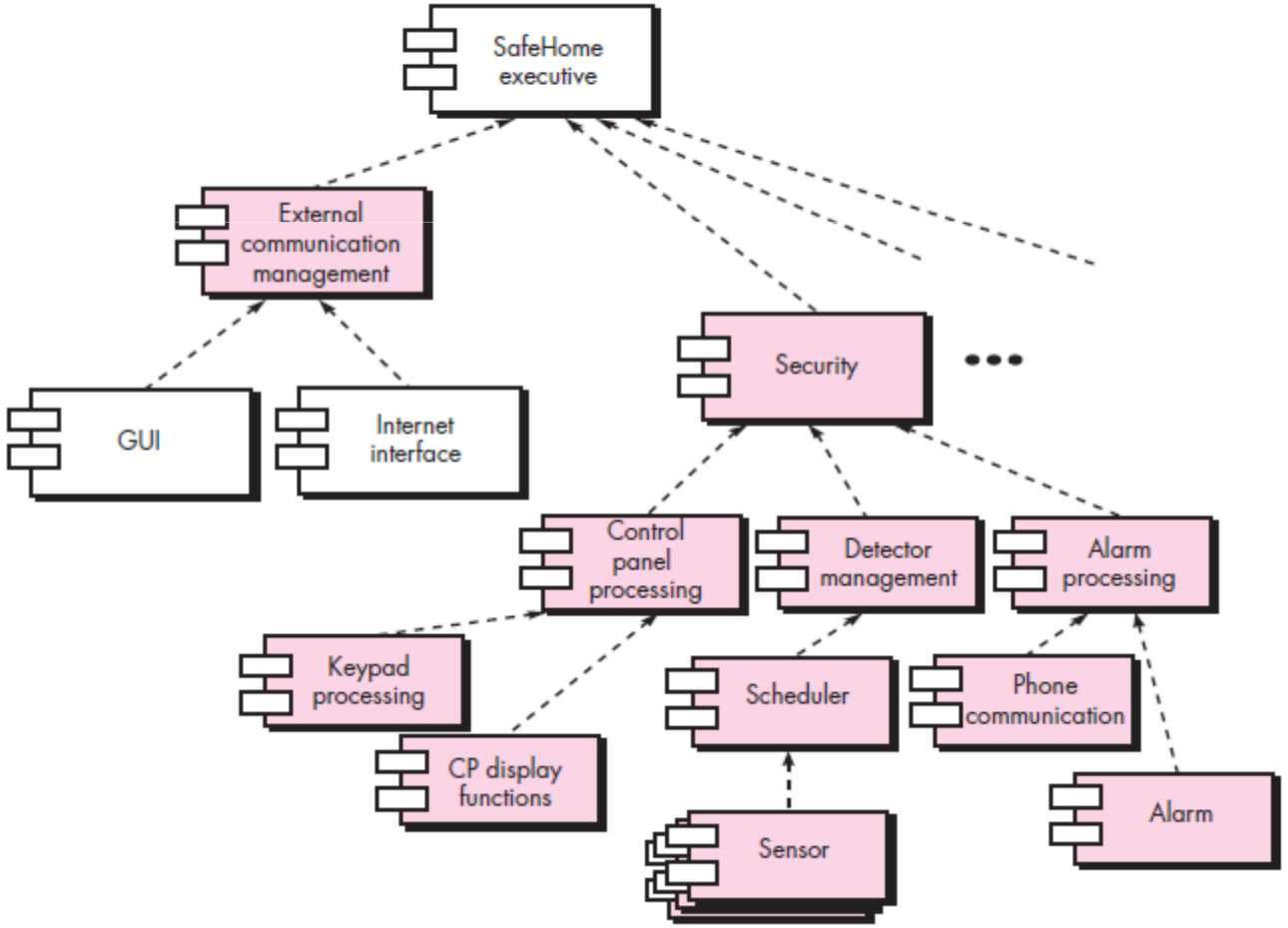
Architecture design

- Refining the Architecture into Components :
- Software architecture is refined into components. The components are derived from analysis class within application domain.
- Also, many infrastructure components are derived apart from application domain components. Eg. Memory management component.
- For eg, Based on the functionality, the following components are derived from SafeHome home security function: external communication management, control panel process, detector management, alarm processing.
- Design class would be defined after component level design.
- Fig overall architecture structure for safeHome home security function



Architecture design

- Describing Instantiations of the System : Architecture design at the high level contains the following details: archetype, overall structure of the system, major system components.
- To further refine the architecture, the components are elaborated to show additional details.



Architectural design using data flow

- Call and return architecture, eg. Client server architecture.
- A mapping technique, called *structured design,* provides a convenient transition from a data flow diagram to software architecture using the following 6 steps.
- (1) the type of information flow is established,
- (2) flow boundaries are indicated,
- (3) the DFD is mapped into the program structure,
- (4) control hierarchy is defined,
- (5) the resultant structure is refined using design measures and heuristics, and
- (6) the architectural description is refined and elaborated.