# SOFTWARE DESIGN & ANALYSIS (Week-2)


USAMA MUSHARAF

MS-CS (Software Engineering)

*LECTURER (Department of Computer Science)*

*FAST-NUCES PESHAWAR*

# CONTENTS OF WEEK # 2

- Software Design & Architecture
- Software Development Process Models

# WHAT IS DESIGN?

- A staircase that leads right into  a wall!

- A door that would drop you 10 feet down!

# WHAT IS DESIGN?

- Design is the first step in the development phase for any engineered product or system.
- Design is about HOW the system will perform its functions.

# SOFTWARE DESIGN

- A software design is a meaningful engineering representation of some software product that is to be built.

- “The process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization”.

# SOFTWARE DESIGN - SIMPLIFIED

Requirements specification was about the WHAT the system will do

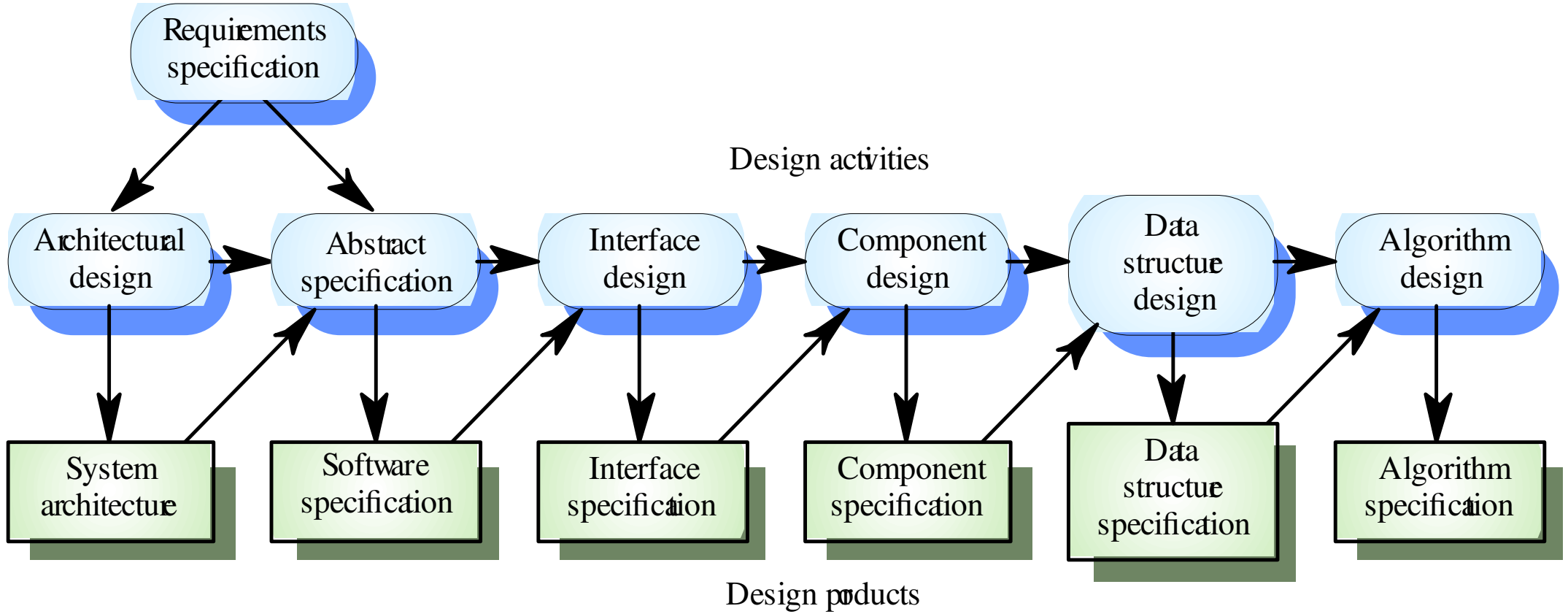
Design is about the HOW the system will perform its functions

- provides the overall decomposition of the system.
- allows to split the work among a team of developers.
- also lays down the groundwork for achieving non-functional requirements (performance, maintainability, reusability, etc.)

# DESIGN PROCESS ACTIVITIES

- Architectural design
  - Modules, inter-relationships etc
- Abstract specification
  - Services of each sub-system, constraints etc
- Interface design
  - Interface to other sub-system or outside environment
- Component design
  - Services allocated to components and their interfaces designed
- Data structure design
- Algorithm design

# THE SOFTWARE DESIGN PROCESS



# LEVELS OF SOFTWARE DESIGN

## Architectural design (high-level design)

- architecture - the overall structure, main modules and their connections
- addresses the main non-functional requirements (e.g., reliability, performance)
- hard to change

## Detailed design (low-level design)

- the inner structure of the main modules
- detailed enough to be implemented in the programming language

# DESIGN VS. ARCHITECTURE

- Architecture is concerned with the selection of architectural elements, their interaction, and the constraints on those elements and their interactions

- Design is concerned with the modularization and detailed interfaces of the design elements, their algorithms and procedures, and the data types needed to support the architecture and to satisfy the requirements.

- Architecture…is specifically not about…details of implementations (e.g., algorithms and data structures.)

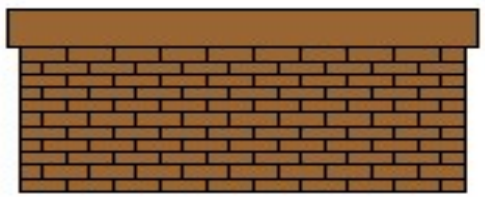
# Software Development

- Lists
- Arrays
- Class
- Object
- Procedures
- Functions
- Algorithms
- Etc.

```
//********************************************************************
import inventoryItem;
import java.util.StringTokenizer;
import java.io.*;

public class Inventory
{
  //-----------------------------------------------------------
  //  Reads data about a store inventory from an input file,
  //  creating an array of InventoryItem objects, then prints them.
  //-----------------------------------------------------------
  public static void main (String[] args)
  {
    final int MAX = 100;
    InventoryItem[] items = new InventoryItem[MAX];
    StringTokenizer tokenizer;
    String line, name, file="inventory.dat";
    int units, count = 0;
    float price;

    try
for (int scan = 0; scan < count; scan++)
        System.out.println (items[scan]);
    }
    catch (FileNotFoundException exception)
    {
      System.out.println ("The file " + file + " was not found.");
    }
    catch (IOException exception)
    {
      System.out.println (exception);
    }
  }
}
```

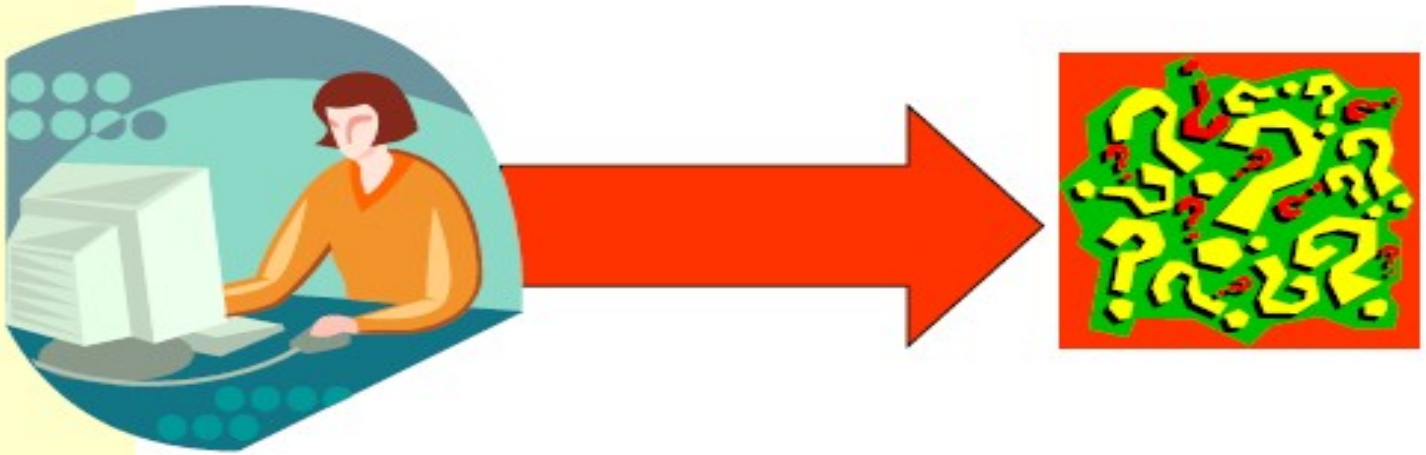


# Large-scale,complex software systems...

- Large (Distributed) System
- Many people working on the same problem
- Overly Complex
- Millions of Code...
- Should be deliverd on time and within budget!

# Coding only will not do…

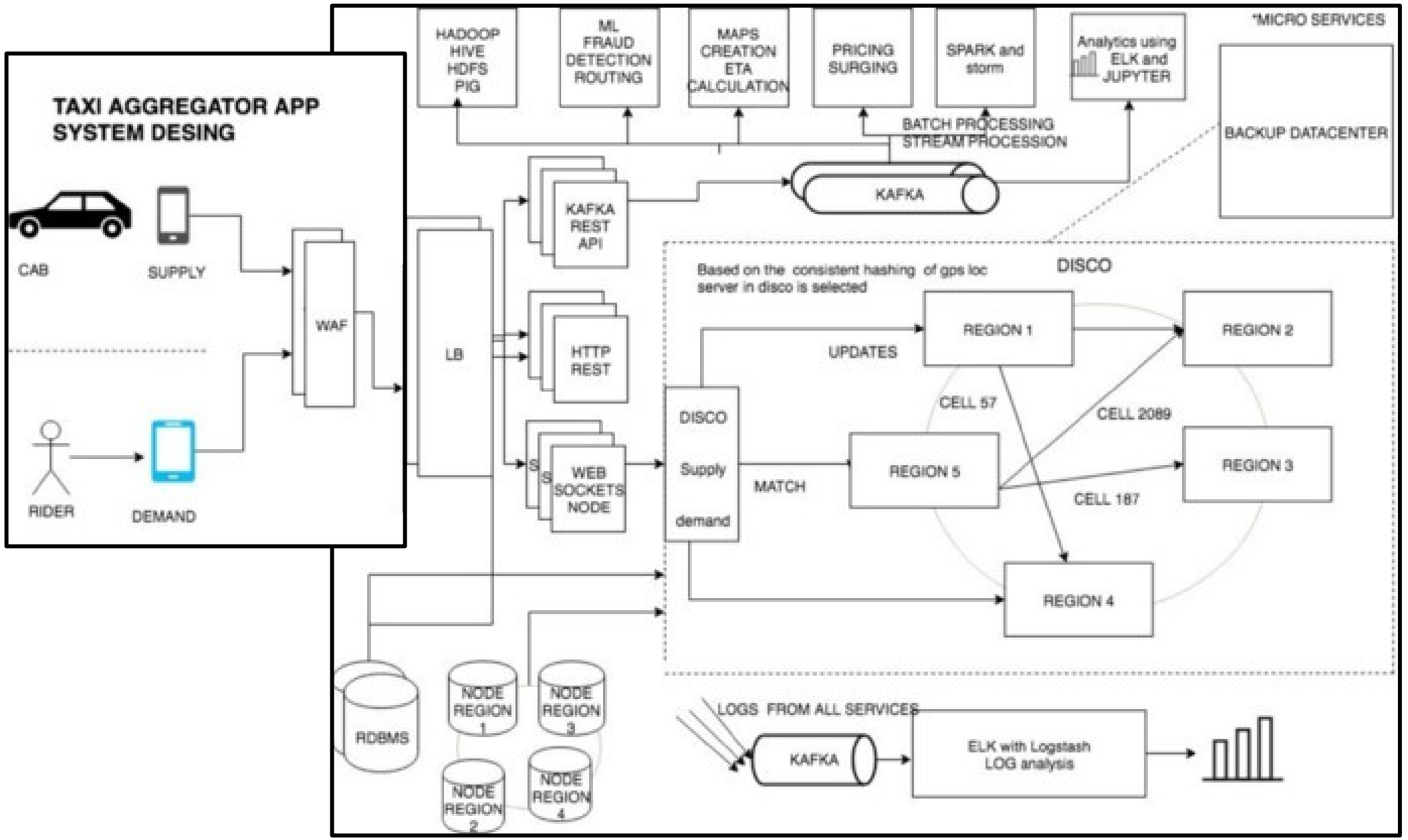
- Lists
- Arrays
- Class
- Object
- Procedures
- Functions
- Algorithms
- Etc.



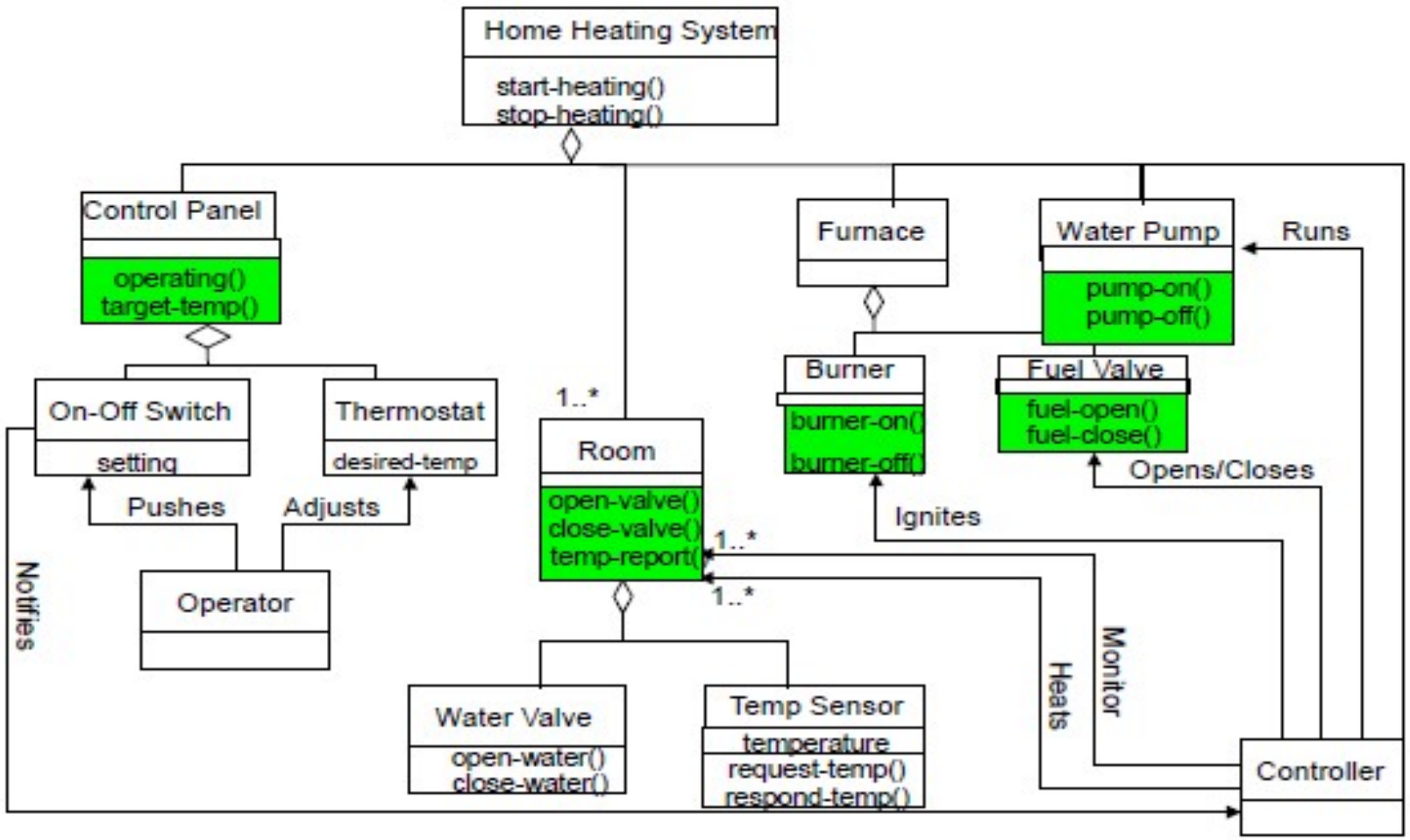
# More programmers...?

# Software Architecture

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 between them.


TAXI AGGREGATOR APP
SYSTEM DESING
CAB
SUPPLY
RIDER
DEMAND
WAF
LB
KAFKA REST API
HTTP REST
WEB SOCKETS NODE
HADOOP HIVE HDFS PIG
ML FRAUD DETECTION ROUTING
MAPS CREATION ETA CALCULATION
PRICING SURGING
SPARK and storm
Analytics using ELK and JUPYTER
*MICRO SERVICES
BACKUP DATACENTER
BATCH PROCESSING
STREAM PROCESSION
KAFKA
DISCO
Based on the consistent hashing of gps loc server in disco is selected
UPDATES
REGION 1
REGION 2
REGION 3
REGION 4
REGION 5
CELL 57
CELL 2089
CELL 187
DISCO
Supply
demand
MATCH
RDBMS
NODE REGION 1
NODE REGION 2
NODE REGION 3
NODE REGION 4
LOGS FROM ALL SERVICES
KAFKA
ELK with Logstash
LOG analysis

# DESIGN EXAMPLE



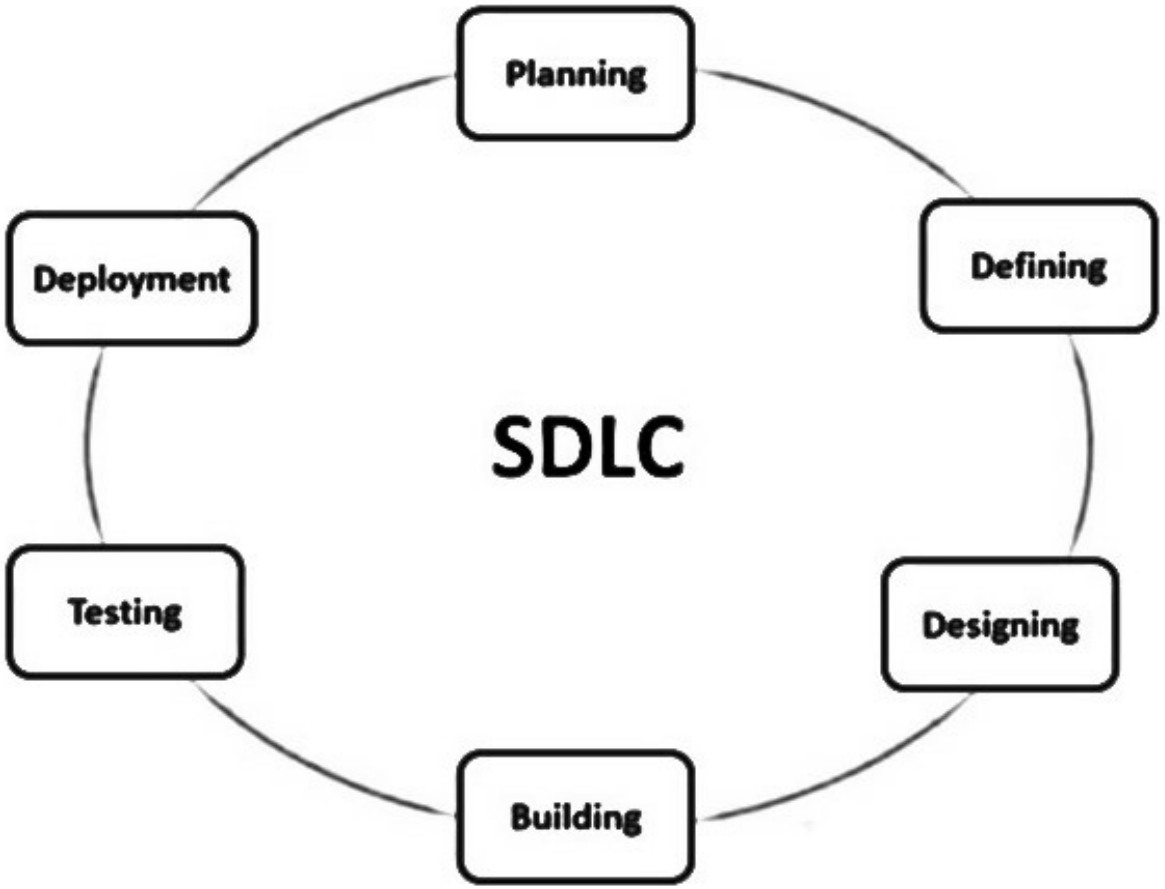
# SOFTWARE PROCESS MODELS

# WHAT IS A SOFTWARE PROCESS?

- SP is a **set of activities** whose goal is the development or evolution of software.
- Fundamental activities in all software processes are:

  - **Specification** - what the system should do and its development constraints.
  - **Development** - production of the software system (design and implementation)
  - **Validation** - checking that the software is what the customer wants
  - **Evolution** - changing the software in response to changing demands

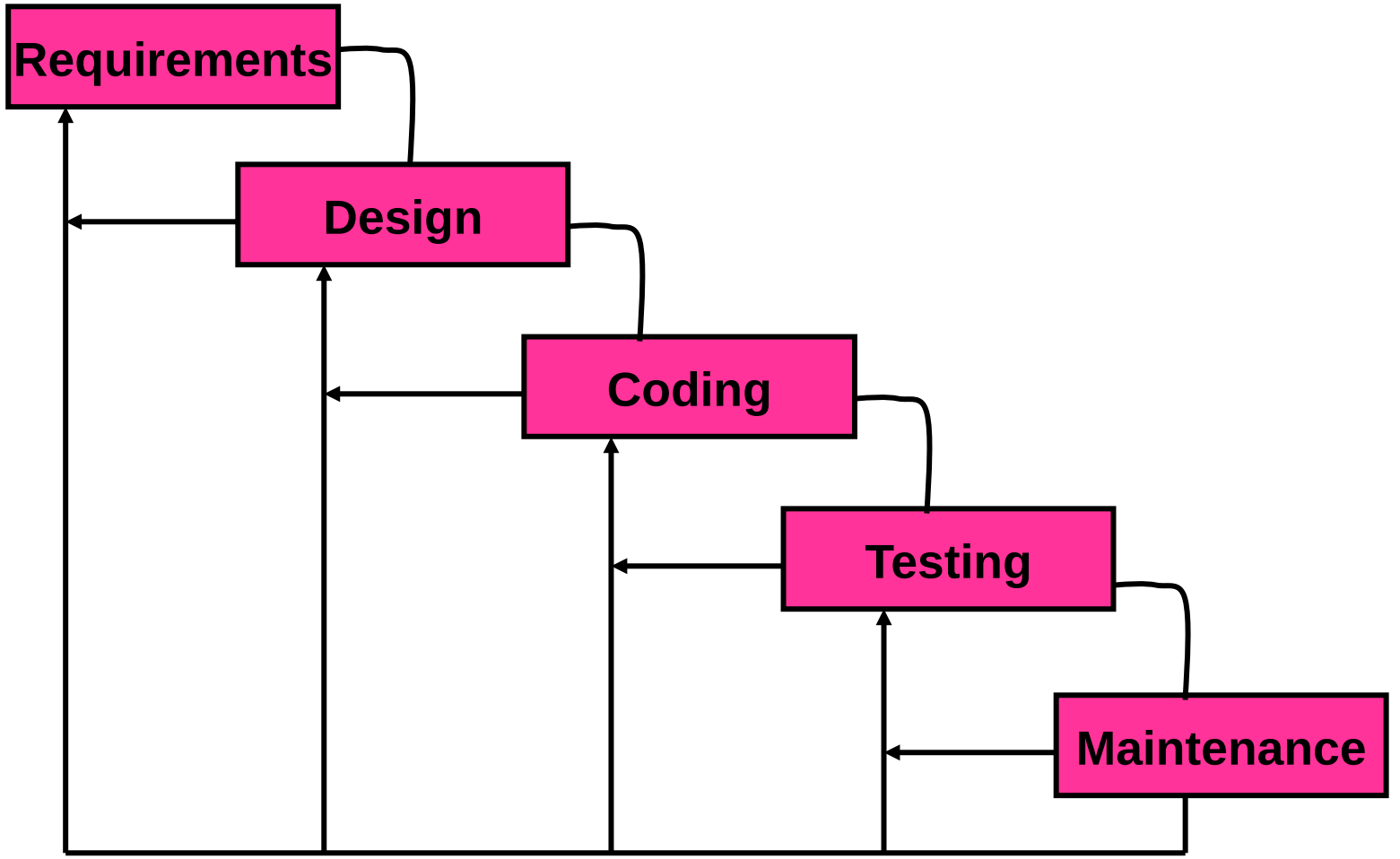
# SOFTWARE DEVELOPMENT LIFE CYCLE



# SOFTWARE PROCESS MODELS

1. **Waterfall model**
2. **V Model**
3. **Rapid prototyping model**
4. **Evolutionary s/w process Model**
   - **Incremental model**
   - **Spiral Model**

# WATERFALL PROCESS MODEL
# AKA
# LINEAR SEQUENTIAL MODEL



# WATERFALL MODEL ADVANTAGES

- Simple and easy to understand and use
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

# WATERFALL MODEL PROBLEMS

The drawback of the waterfall model is the difficulty of accommodating change after

the process is underway.

This makes it difficult to respond to changing customer requirements

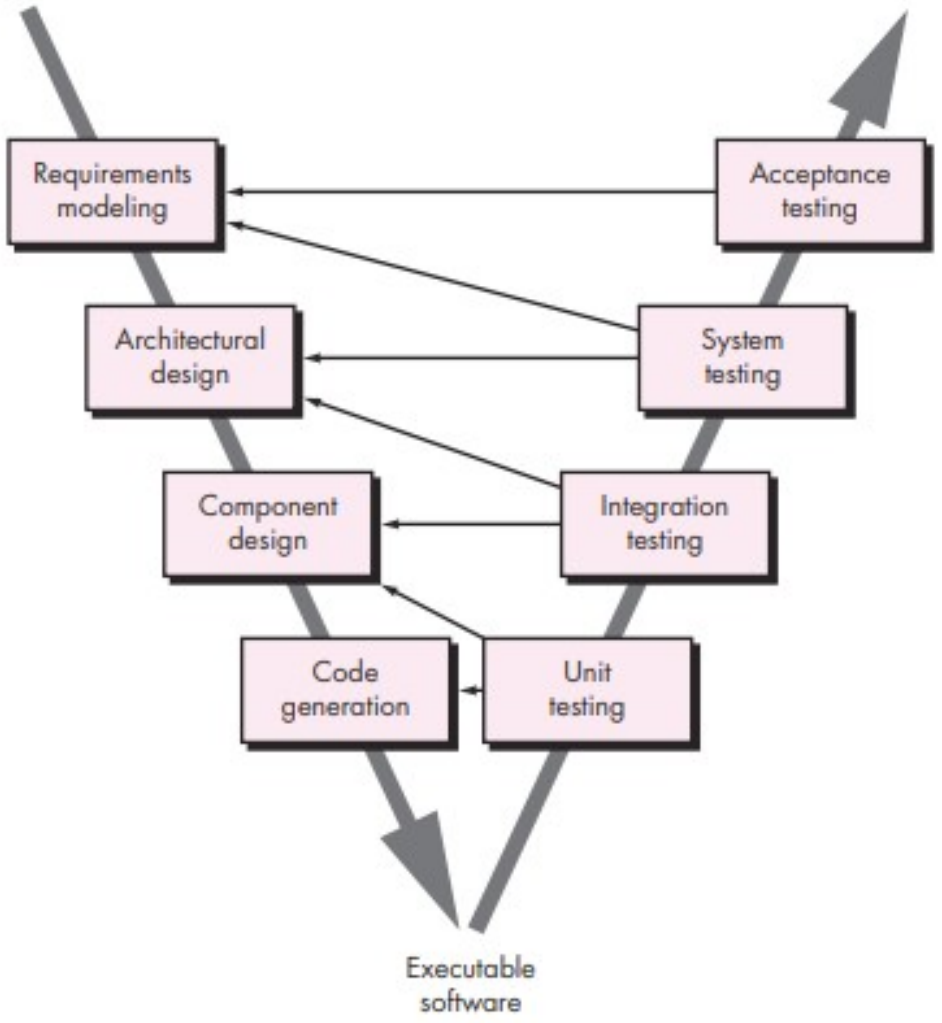
- Therefore, this model is only appropriate when the requirements are well-understood.

**Waterfall model describes a process of stepwise refinement**

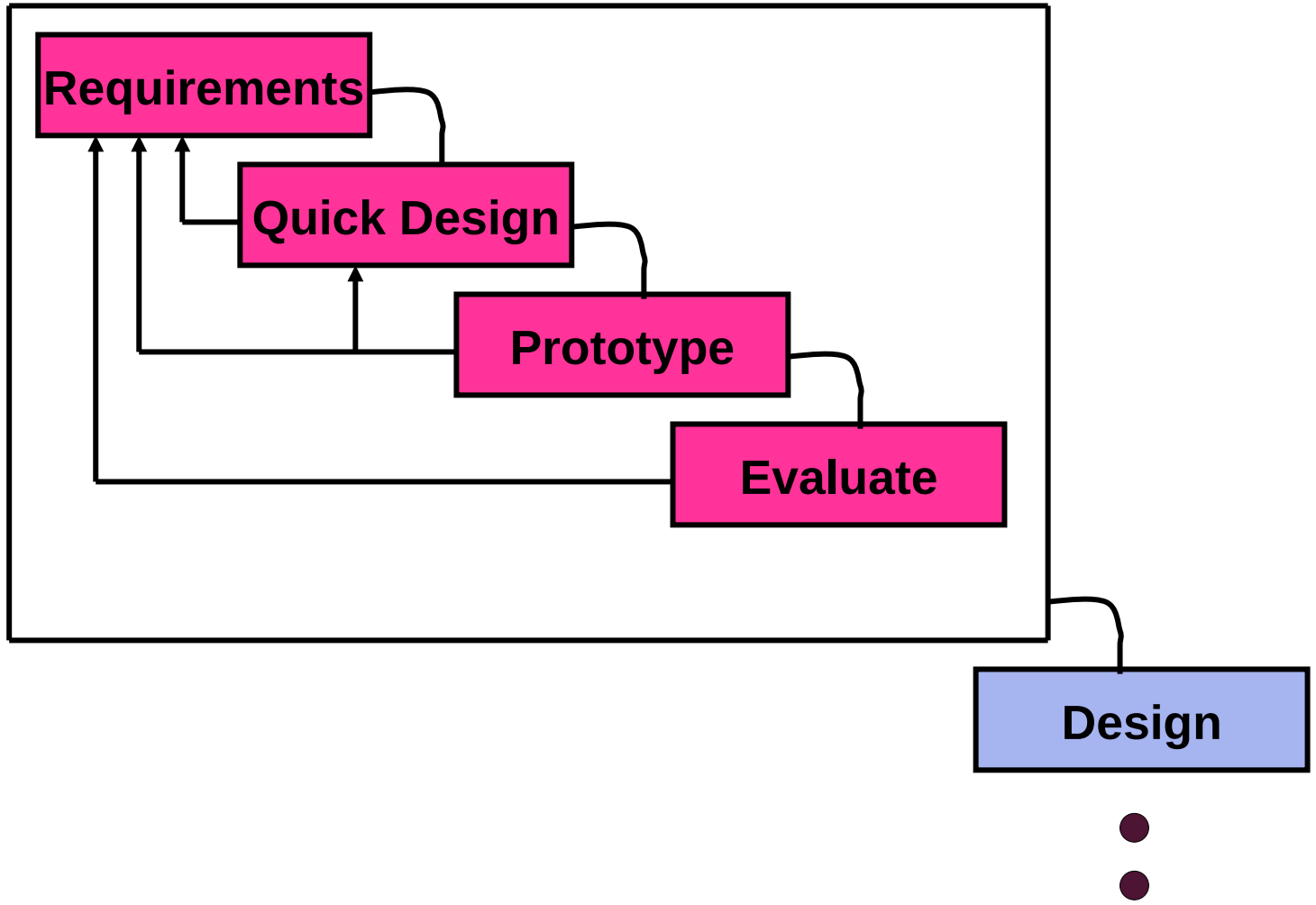
# WATERFALL MODEL (PROBLEMS)

- Doesn't support iteration, so changes can cause confusion
- Difficult for customers to state all requirements explicitly and up front
- Requires customer patience because a working version of the program doesn't occur until the final phase

# V MODEL


Requirements modeling
Acceptance testing
Architectural design
System testing
Component design
Integration testing
Code generation
Unit testing
Executable software

# RAPID PROTOTYPING PROCESS MODEL



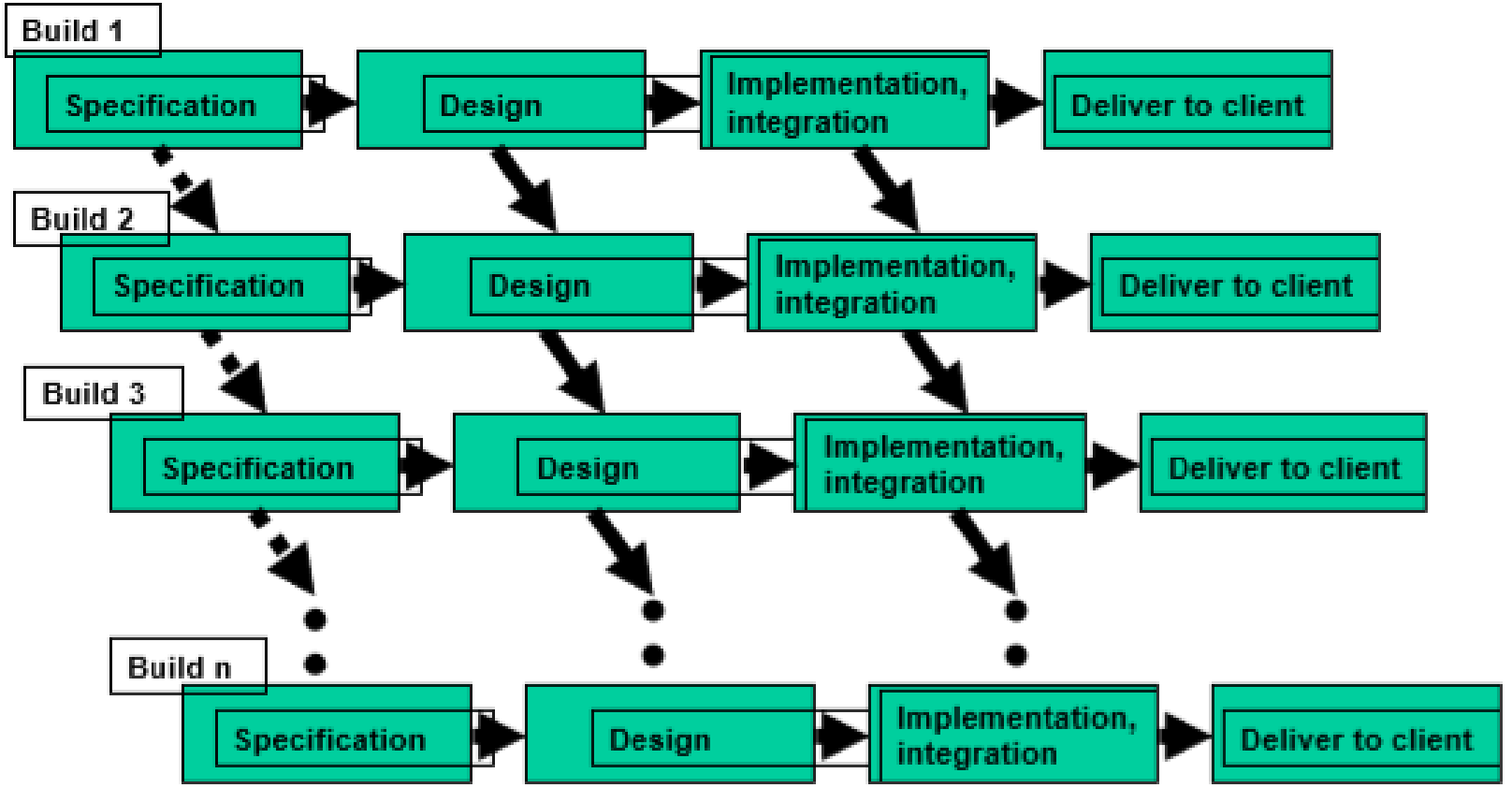
# RAPID PROTOTYPING MODEL

- The Rapid Prototyping Model is used to overcome issues related to understanding and capturing of user requirements.

# Evolutionary s/w process Model

1. Incremental model
2. Spiral Model

# INCREMENTAL OR ITERATIVE PROCESS MODEL



# INCREMENTAL OR ITERATIVE PROCESS MODEL

- Rather than deliver the system as a single delivery, **the development and delivery is broken down into increments** with each increment delivering part of the required functionality.

- **Once the development of an increment is started, the requirements are frozen** though requirements for later increments can continue to evolve.

# INCREMENTAL OR ITERATIVE DEVELOPMENT ADVANTAGES

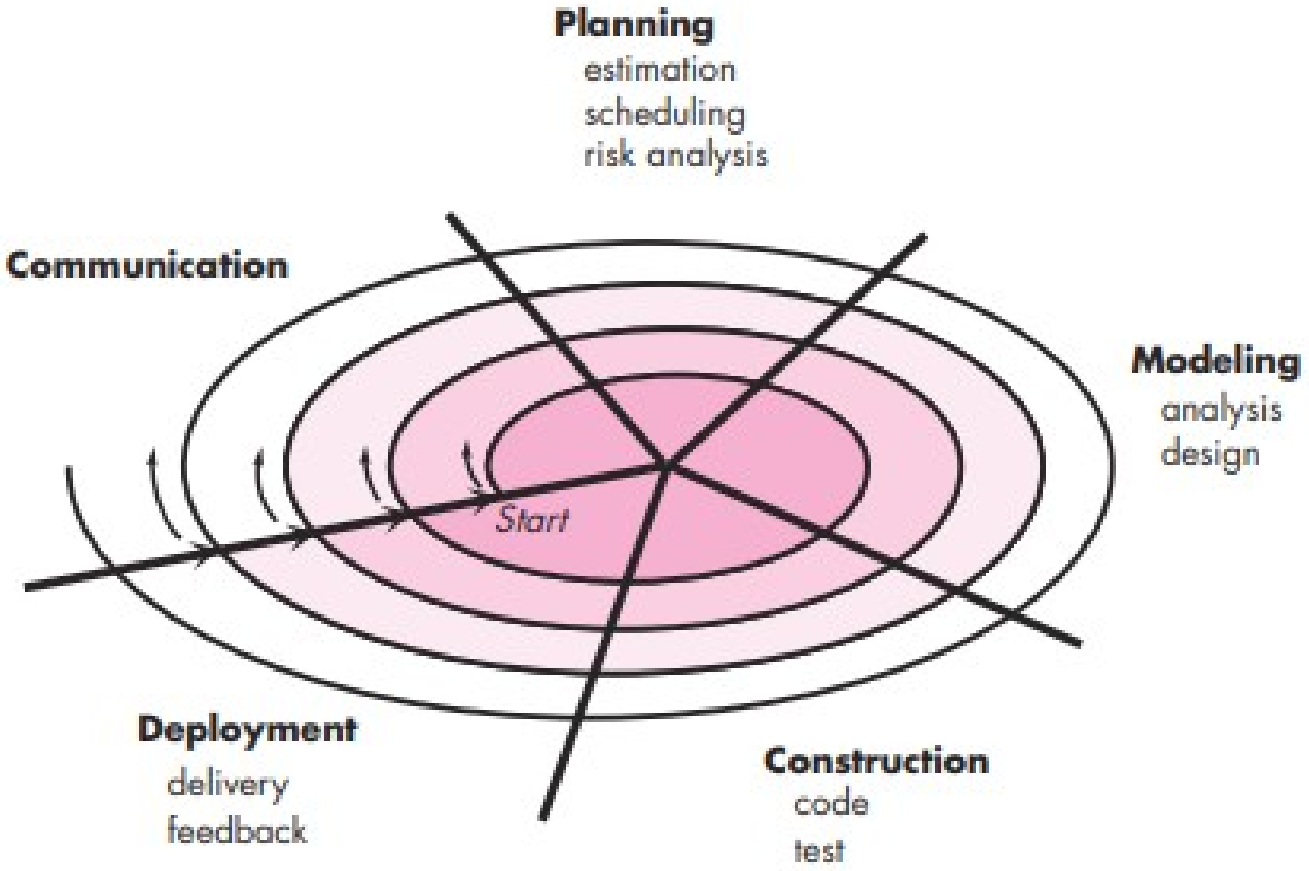
- **Customer value** can be delivered with each increment so system functionality is available earlier.
- **Early increments** act as a prototype to help elicit requirements for later increments.
- Lower risk of overall project failure.

# SPIRAL PROCESS MODEL

- The spiral model, originally proposed by Boehm is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the linear sequential model.

- Using the spiral model, software is developed in a series of incremental releases.

# SPIRAL PROCESS MODEL



# SPIRAL MODEL(DESCRIPTION)

- Follows an evolutionary approach
- Used when requirements are not well understood and risks are high
- Operates as a risk-driven model…a go/no-go decision occurs after each complete spiral in order to react to risk determinations
- Requires considerable expertise in risk assessment
- Serves as a realistic model for large-scale software development

# GENERAL WEAKNESSES OF EVOLUTIONARY PROCESS MODELS

1) Prototyping poses a problem to project planning because of the uncertain number of iterations required to construct the product

2) Evolutionary software processes do not establish the maximum speed of the evolution
   - If too fast, the process will fall into chaos
   - If too slow, productivity could be affected

# HAVE A GOOD DAY!