



# Inxhinieria Sistemeve Softuerike

## **Modelimi/Dizajni i Sistemit:**

---

**FAKULTETI: SHKENCA KOMPJUTERIKE DHE INXHINIERI**

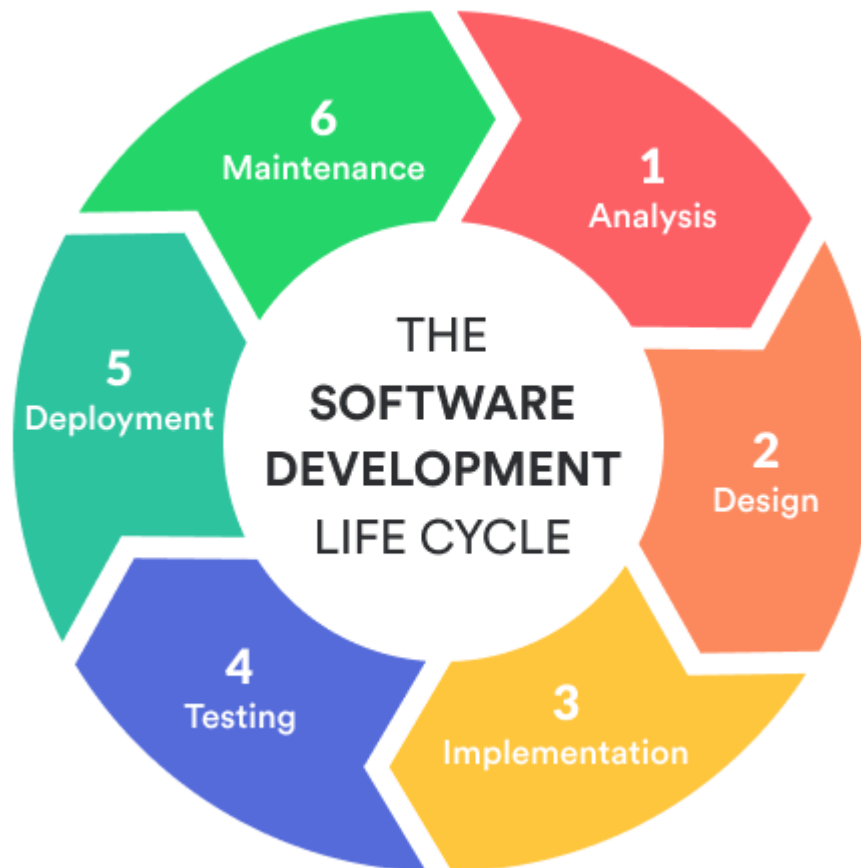
# Kalimi nga kërkesat në Dizajn apo Modelim

---





# *Ri-kujtim:* Cikli i Jetës së Zhvillimit të Softuerit



# Idetë kryesore

□ *Kërkesat* -> **Dizajni** -> *Implementimi*

□ Në Analizën e Sistemeve ne kuptojmë...

- Çfarë është nevojat e biznesit

□ Në Modelimin/Dizajnimi i Sistemit ne kuptojmë...

- “*Si*” sistemi do të *konfigurohet* dhe *ndërtohet* që të plotëson **ato nevoja**

□ E gjithë puna “**logjike**” nga *analiza e sistemit* është konvertuar në “**fizike**”

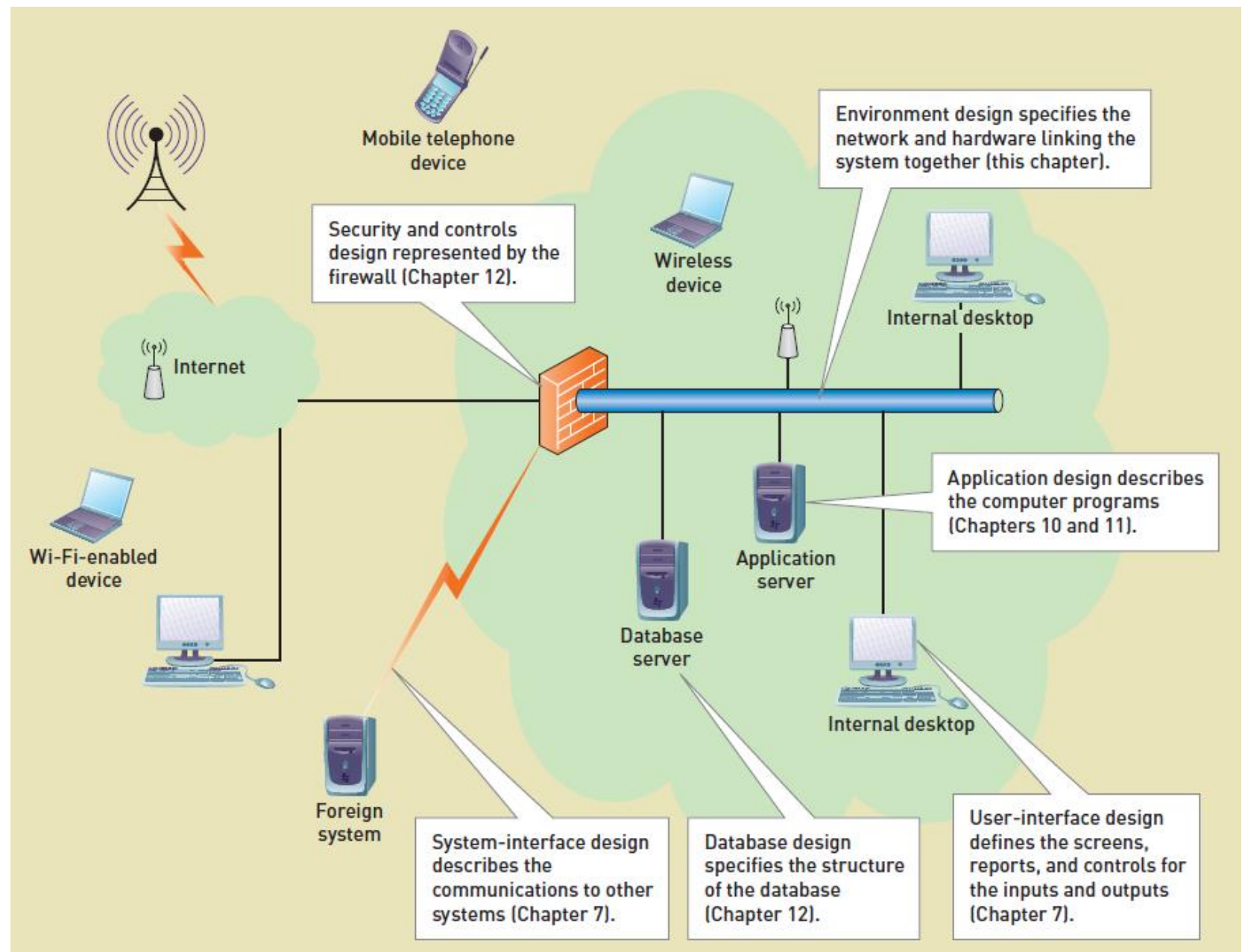


# Mënyrat për të krijuar një system të ri

---

- ❑ Zhvillimi i sitemit të *përsionalizuar* (customized) (*ndërtimi nga e para*) brenda shtëpisë
- ❑ *Blejë pakon* e softuerit (dhe mundësisht aranzhoje)
  - Instalo në rrethinen (kompjuterint) tuaj ose
  - Siguro qasje nga ofruesit e softuerve (host)
- ❑ Zhvillimi nga *burimet e jashtme (outsource)* nga palët e treta, të cilët mund
  - Ndërtojn sistemin nga e para për ne ose
  - Blejnë softuerin për ne, përsionalizo/aranzho dhe instalo atë

***Shembull:***  
Komponentet për Dizajn  
të një sistemi softuerik



# Karakteristika e Dizajnit të Softuerit

- Dizajni i softuerit është një proces përsëritës (interativ) përmes të cilit kërkesat përkthehen në një "*plan*" për ndërtimin e softuerit.
- **Tre karakteristika** që shërbejnë si një udhëzues për vlerësimin e një modeli të mirë:
  - 1) Dizajni *duhet* të zbatojë të gjitha *kërkesat e eksplicite* të përfshira në modelin e kërkesave dhe duhet të *përfshijë* të gjitha kërkesat *implicite* të dëshiruara nga *palët e interesit*.
  - 2) Dizajni duhet të jetë një *udhëzues* i *lexueshëm*, i *kuptueshëm* për ata që *gjenerojnë kodin* dhe për ata që *testojnë* dhe *mbështesin* më pas programin.
  - 3) Dizajni duhet të ofroj një *pamje të plotë të softuerit*, duke adresuar të dhënat, funksionale dhe *sjelljes* nga një perspektivë zbatimi

# Dy nivele të dizajnit

## □ Dizajni i Arkitekturës (niveli i lartë i dizajnit)

- Dizajni i nivelit të lartë – *strukturimi* i përgjithshme i sistemit
- Po ashtu quhet edhe dizajn i *përgjithshëm* dhe dizajn *konceptual*

## □ Dizajn i detajuar (niveli i ulët i dizajnit)

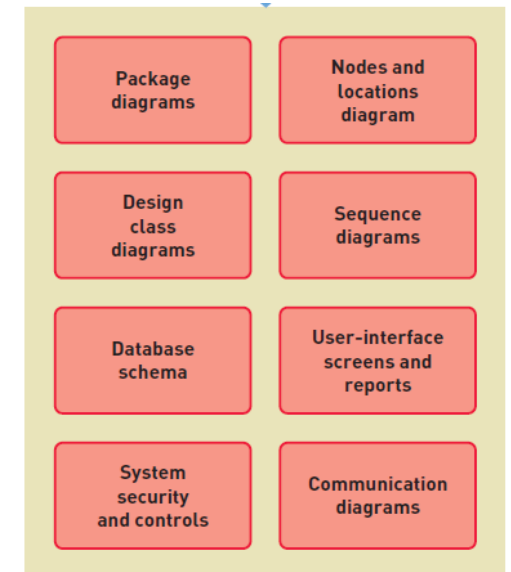
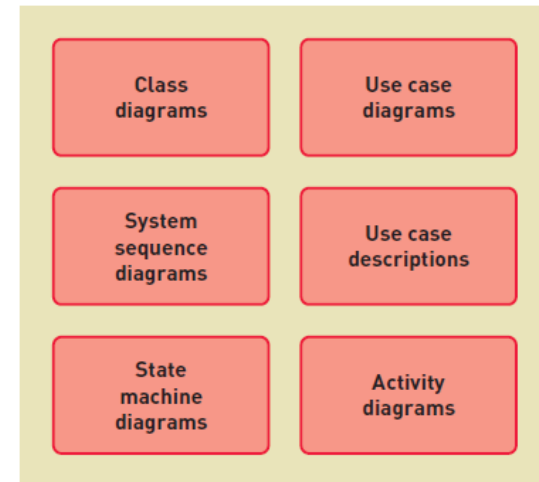
- Dizajn i nivelit të ulët që përfshin dizajni i detajeve specifike të programit
  - Dizajni i secilit rastë të përdorimit (use case).
  - Dizajni i bazës së të dhënës.
  - Dizajni i programit (class diagrams, moduleve, komponenteve,..., etj)
  - Dizajni interfaces së përdoruesit dhe sistemit
  - Dizajni i kontrolleve dhe siguris



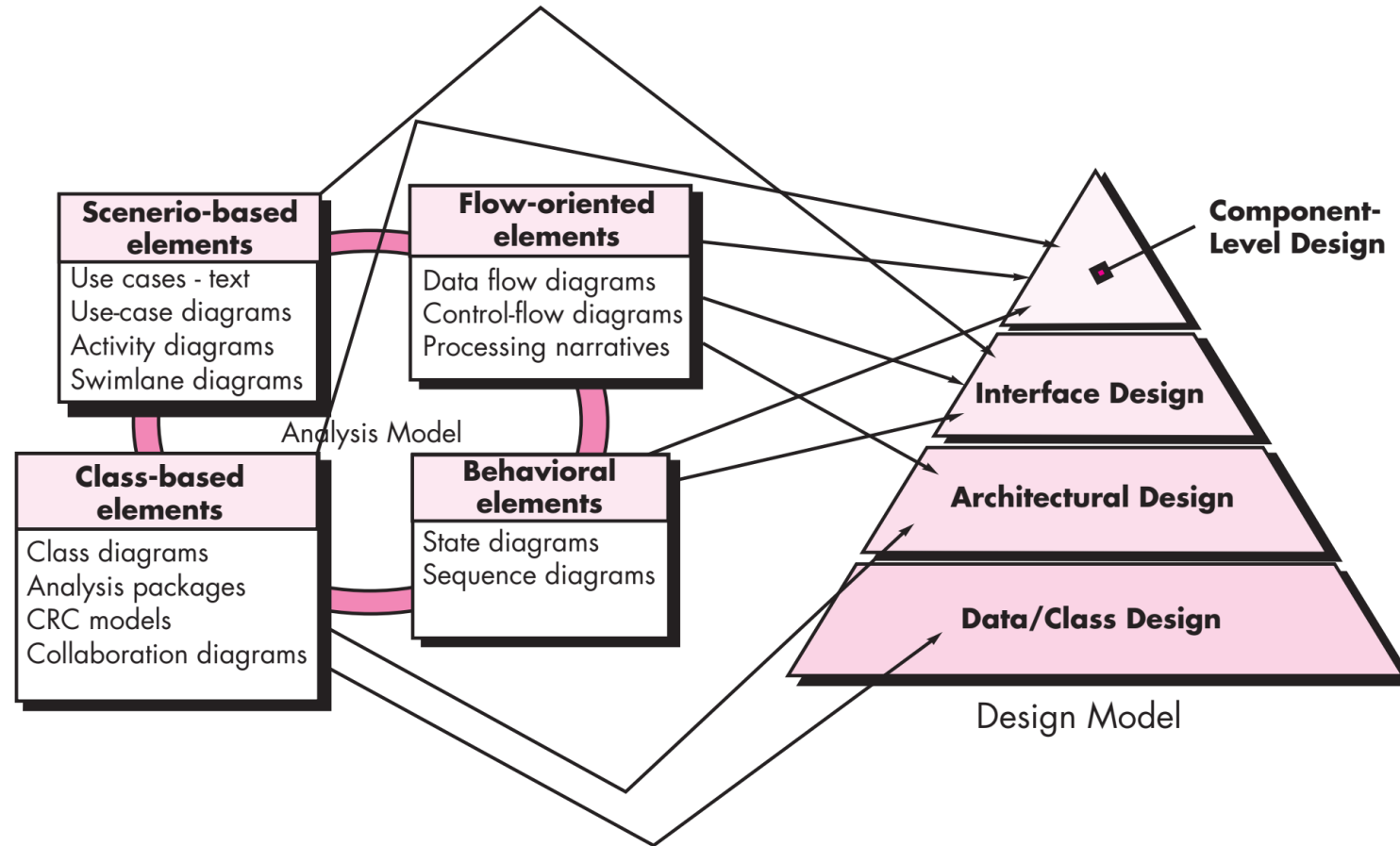
# Dizajnimi i Aplikacionit: Arkitektura dhe Programi (1)

- Definim të arkitektures softuerike
  - Tre shtresor, model-view-controller ose SOA/Mikro-shëbime
- Ndarja e sistemit në nën-sisteme
- Dizajni i detajuar për secilen rastë përdorimit
  - Dizajni i diagramit të klasës
  - Diagramet e Aktivitetit
  - Diagramet e sekuençes
  - Diagramet e gjendjës së makines
  - Diagramet e komponentëve
  - Diagramet e zbatimit (deployment)

Analiza vs.  
Modelet e dizajnit



# Dizajnimi i Aplikacionit: Arkitektura dhe Programi (2)





# Procesit të Dizajnit të Softuerit (1)

## 1. Dizajni duhet të ekspozojë:

- një *stil* ose *patern* arkitektonike të njohur,
- komponimi i **komponenteve** me karakteristika të mira në dizajn dhe
- implementimi në mënyre evolucionare, duke faselituar zbatimin dhe testim.

## 2. Dizajni duhet të jetë *modular*:

- softveri duhet të jetë i ndarë në mënyrë logjike në elementë ose nënsisteme.

## 3. Dizajn duhet të përmbaj në form të theksuar apo të veqant:

- të *dhënat*, *arkitekturën*, *interfaces*, dhe *komponentët*.

## 4. Dizajn duhet të ju drejtoj:

- në strukturim të të dhënave të përshtatshme për *klasat* që do të implementohe dhe që janë të *varura nga paternat* të njohura të të dhënave.
- te komponentët që shfaqin ose ekspozojnë karakteristika *funksionale të pavarura*.
- në *ndërfaqet (interfaces)* që *thjeshtzojnë kompleksitetin e lidhjeve* midis komponentëve me mjedisin e jashtëm.



# Procesit të Dizajnit të Softuerit (2)

---

## 5. Një dizajn duhet të rrjedhin:

- duke përdorur një *metodë përseritse* që është nxitur nga informacionet e marra gjatë analizës kërkesat softuerit.

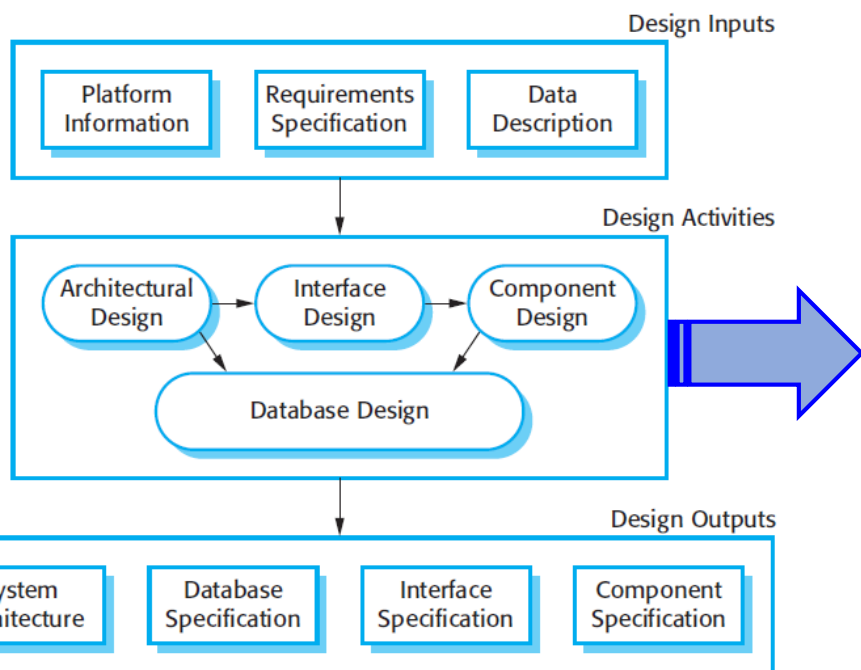
## □ Një dizajn duhet të reprezentohet duke përdorur:

- *simbolet* që në mënyrë efektive dhe të *qartë komunikojn* përmbajtjen e tyre.

# Hapat e fazës së Dizajnimit

- ❑ Përcaktoni *strategjinë* e implementimit të sistemit (të bëni, blini ose outsource)
  - ❑ Përcaktoni *arkitekturën* teknike për sistemin
  - ❑ Adresoni shqetësimet e *sigurisë* dhe çështjet e globalizmit
  - ❑ Bëni përzgjedhje *harduerit* dhe *softuerëve*
  - ❑ Përcaktoni mënyrën se *si përdoruesit do të bashkëveprojnë* me sistemin (GUI, inputet dhe rezultatet)
  - ❑ Dizajno *programet* për *komponentet* themelore të nevojshëm
  - ❑ Dizajno mënyrën e *ruajtjes* së të dhënave
- Krijoni dorëzimin përfundimtar - *specifikimi i sistemit*










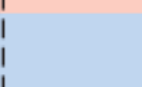
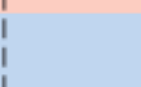
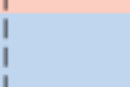
























# Aktivitetet e Dizajnit (1)



Aktiviteti i Dizajnit	Elementet ose Çështje
Dizajni i rrethinës ose mjedisit	Pajisjet, Rrjetet, Serverët
Dizajni i arkitektures së aplikacionit dhe softuerit	Softueri, Konfigurimet
Dizajni i interfasës së sistemit	Mjeti për komunikim me sisteme tjera
Dizajni i ndërfaqeve të përdoruesve	Dritarja e përdoruesit , Raportet
Dizajni i bazës së të dhënave	Magazinimi i informacioneve apo të dhënave
Dizajni i kontrollit dhe Sigurës së sistemit	Firewalls, Qasjet/Kontrolli qasjes

# Aktivitetet e Dizajnit (2)

Design activities
Design the environment.
Design application architecture and software.
Design user interfaces.
Design system interfaces.
Design the database.
Design system controls and security.

Core processes	Iterations					
	1	2	3	4	5	6
Identify problem and obtain approval.						
Plan and monitor the project.						
Discover and understand details.						
Design system components.						
Build, test, and integrate system components.						
Complete system tests and deploy solution.						

dizajni i ***komponenteve***të sistemit

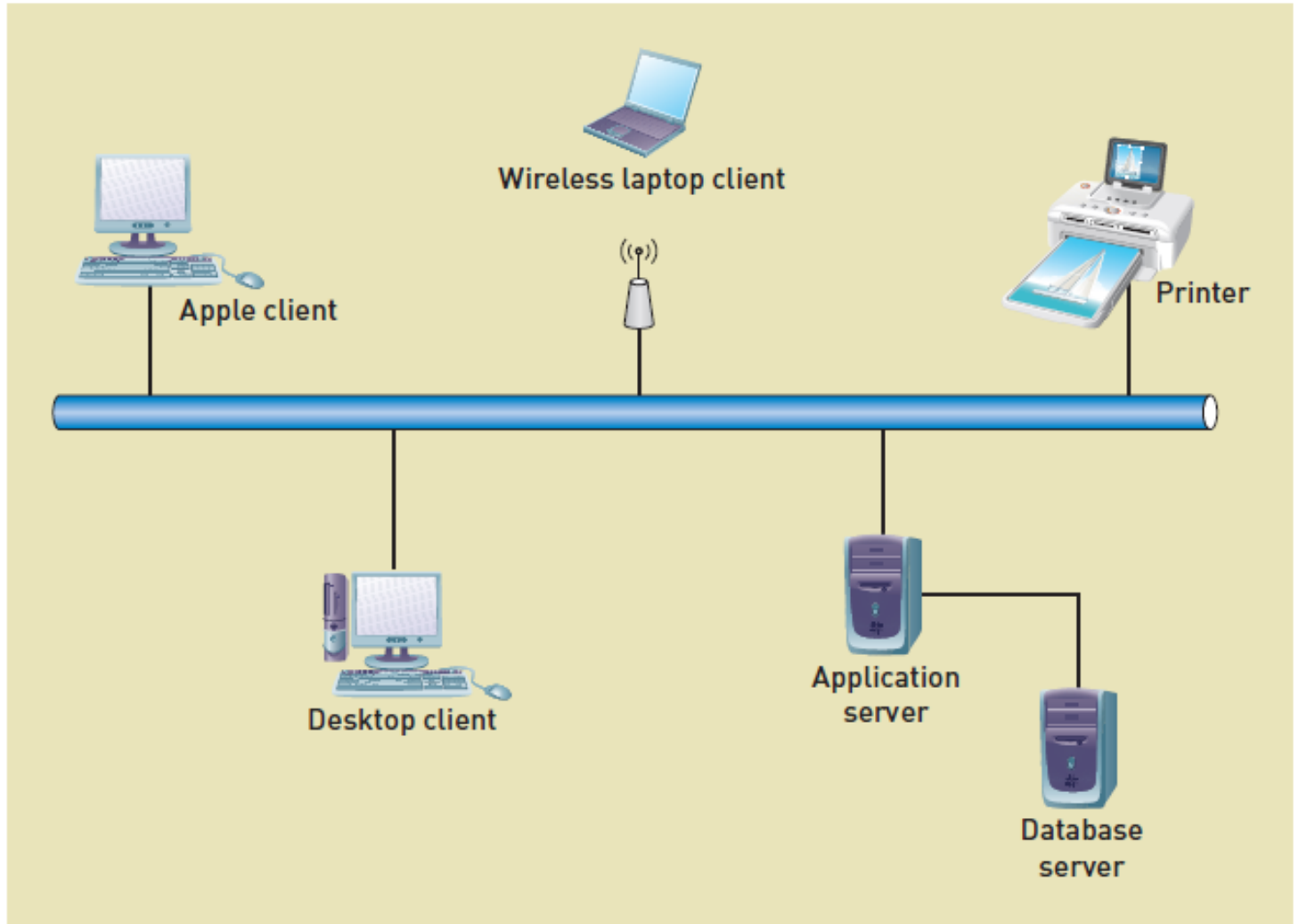
# Dizajnimi i mjedisit/rrethinës

- ❑ **Mjedisi** është e gjitha e *teknologjisë të nevojshme* për të mbështetur *zbatimin* e softuerit.
  - Serveret, Kompjuteret Personal, Pajisjet Celulare, Sistemet Operative
  - Aftësit Komunikimit, Inputet dhe Rezultatet
  - Kjo quhet edhe si arkitektura e teknologjisë
- ❑ Dizajni për **zbatim** apo **dorzim (deployment)** të mbrendshme
  - Stand alone system softuerike
    - Ekzekutim në një paisje pa përdorim të rrjetit.
  - Sistem i bazuar në rrjet të mbrendshme
    - Rrjeti në zonën lokale (LAN), Arkitektura klient-shërbyes (c/s)
    - Aplikacionet Desktop dhe aplikacionet e bazuar-shfletues të webit
- ❑ Arkitektura **klient/shërbyes** (c/s) *tre-shtresorë*
  - Shtresa e prezentimit, shtrasa e domainit and shtresa e të dhënave
  - Aplikacionet Desktop dhe aplikacionet e bazuar-shfletues të webit



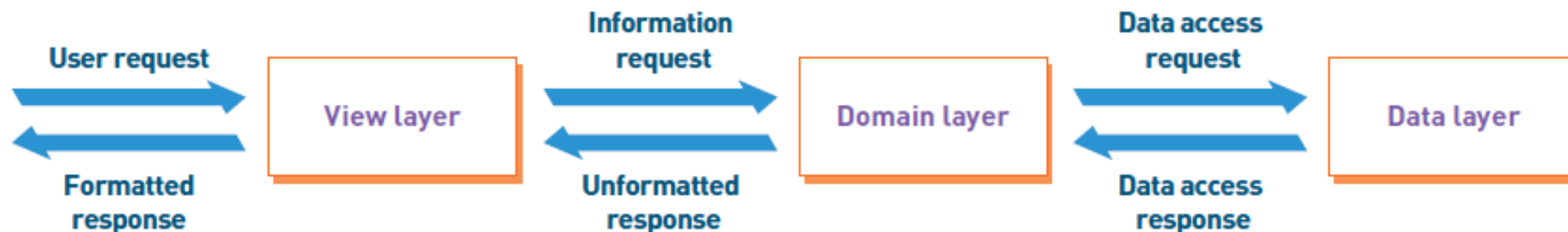
# Diagrami i rrjetit

Sistemi i mbrentshëm i rrjetit

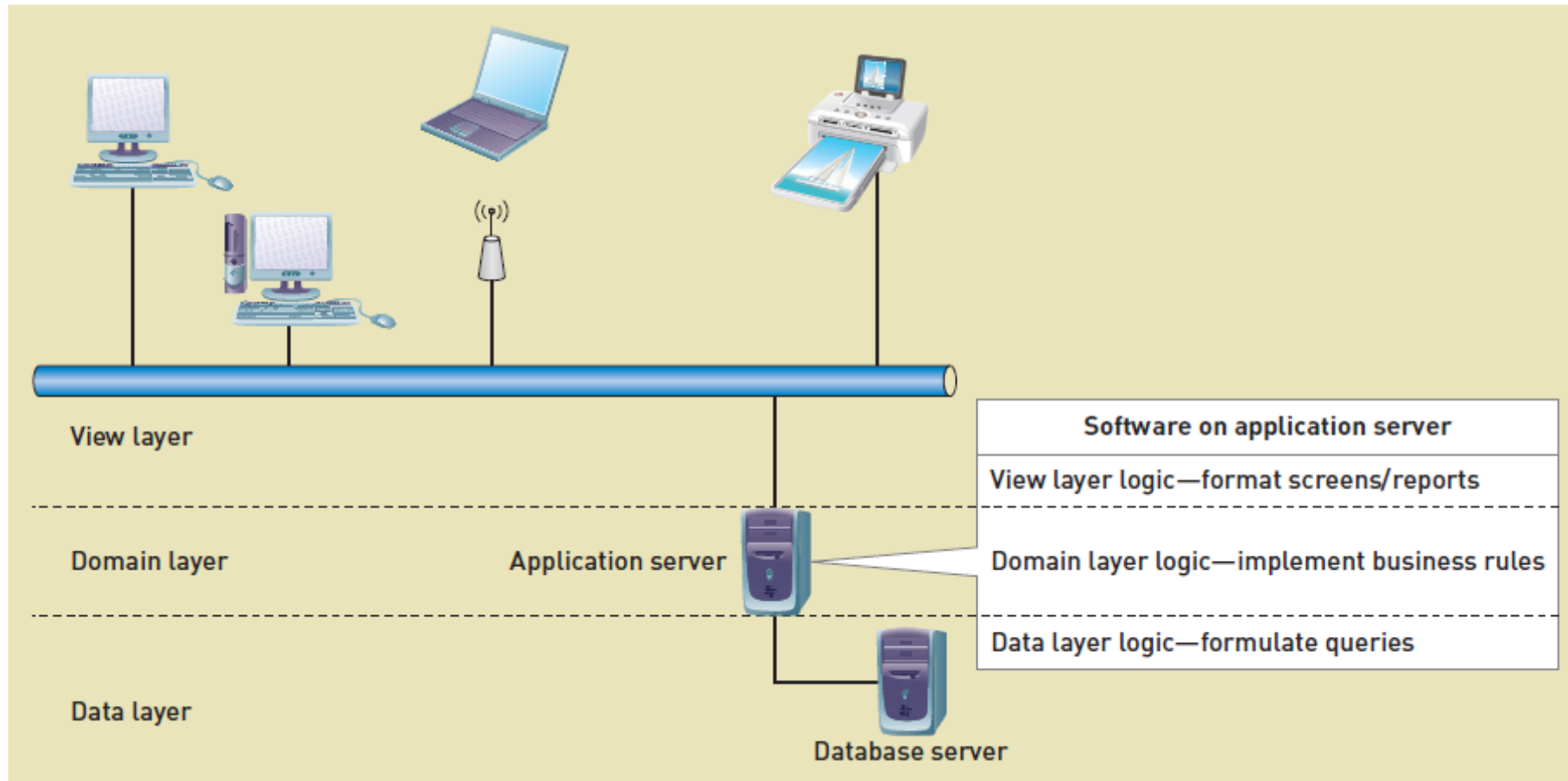


# Arkitekturë abstrakte me *tre Shtresa*

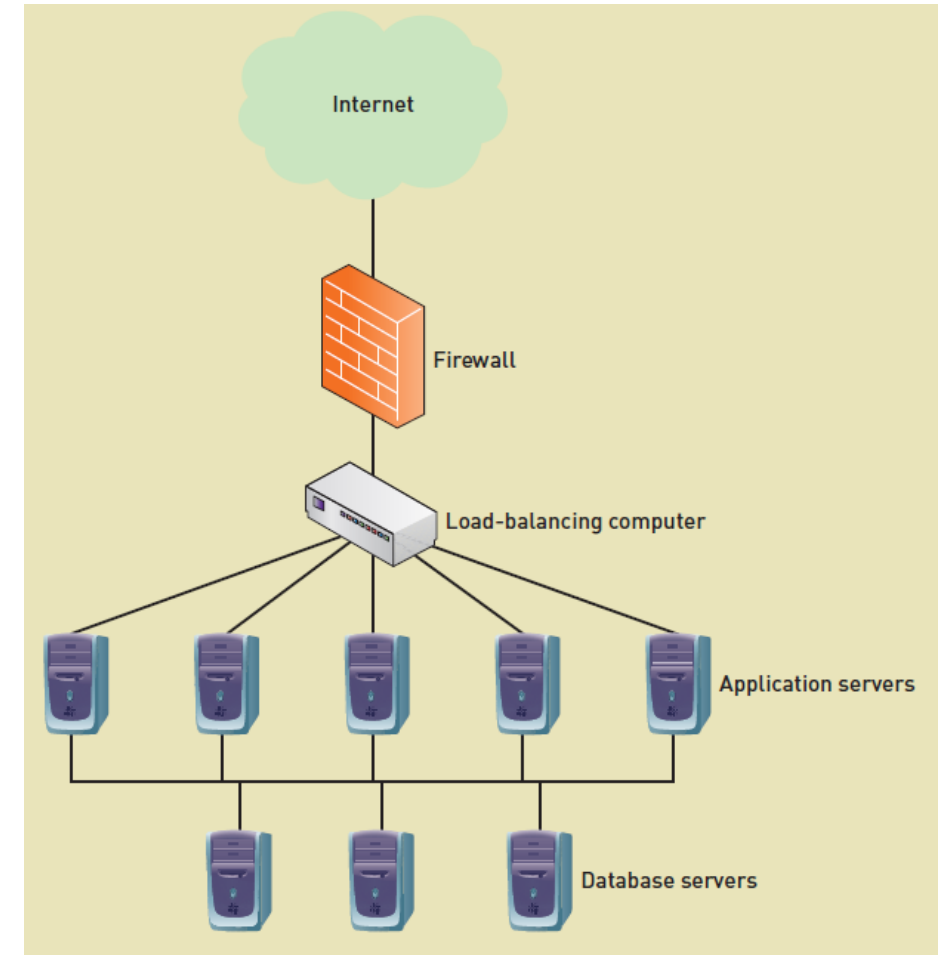
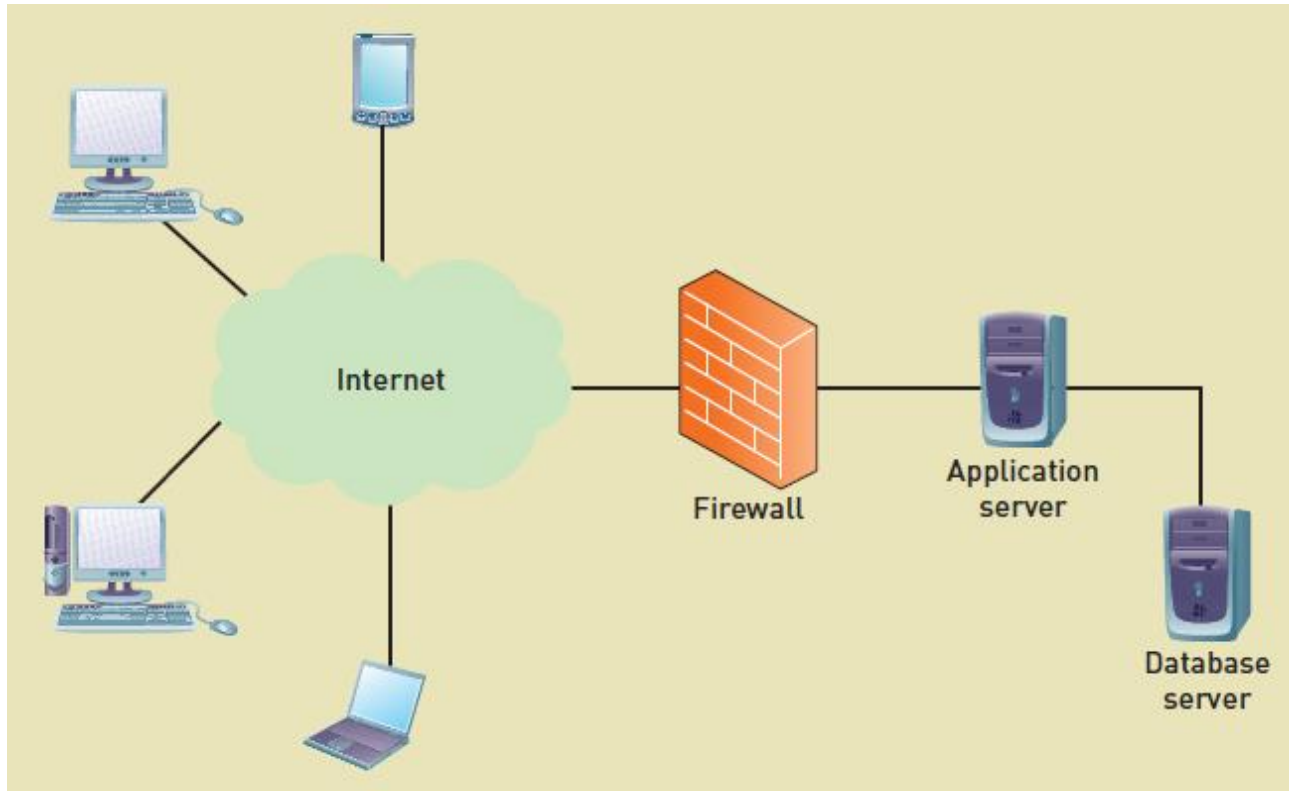
- Arkitektura **3-shtresore Klient-Shërbyes** (Client-Server)
  - një arkitekturë klient/server që ndan një aplikacion në shtresën e prezantimit, shtresën e logjikës së biznesit dhe shtresën e të dhënave
- Shtresa e **prezantimit** - pjesa e arkitekturës me tre-shtresa që përmban ndërfaqen (interface-in) e përdoruesit
- Shtresa e **logjikës së biznesit** - pjesë e një arkitekture *tre-shtresa* që përmban pjesë e programit që implementojnë operacionet ose proceset e biznesit
- Shtresa e të **dhënave** - pjesa e një arkitekture *tre-shtresore* që bashkëvepron me të dhënat



# Zbatimi/vendosja e brendshme me Arkitektura me tre shtresa

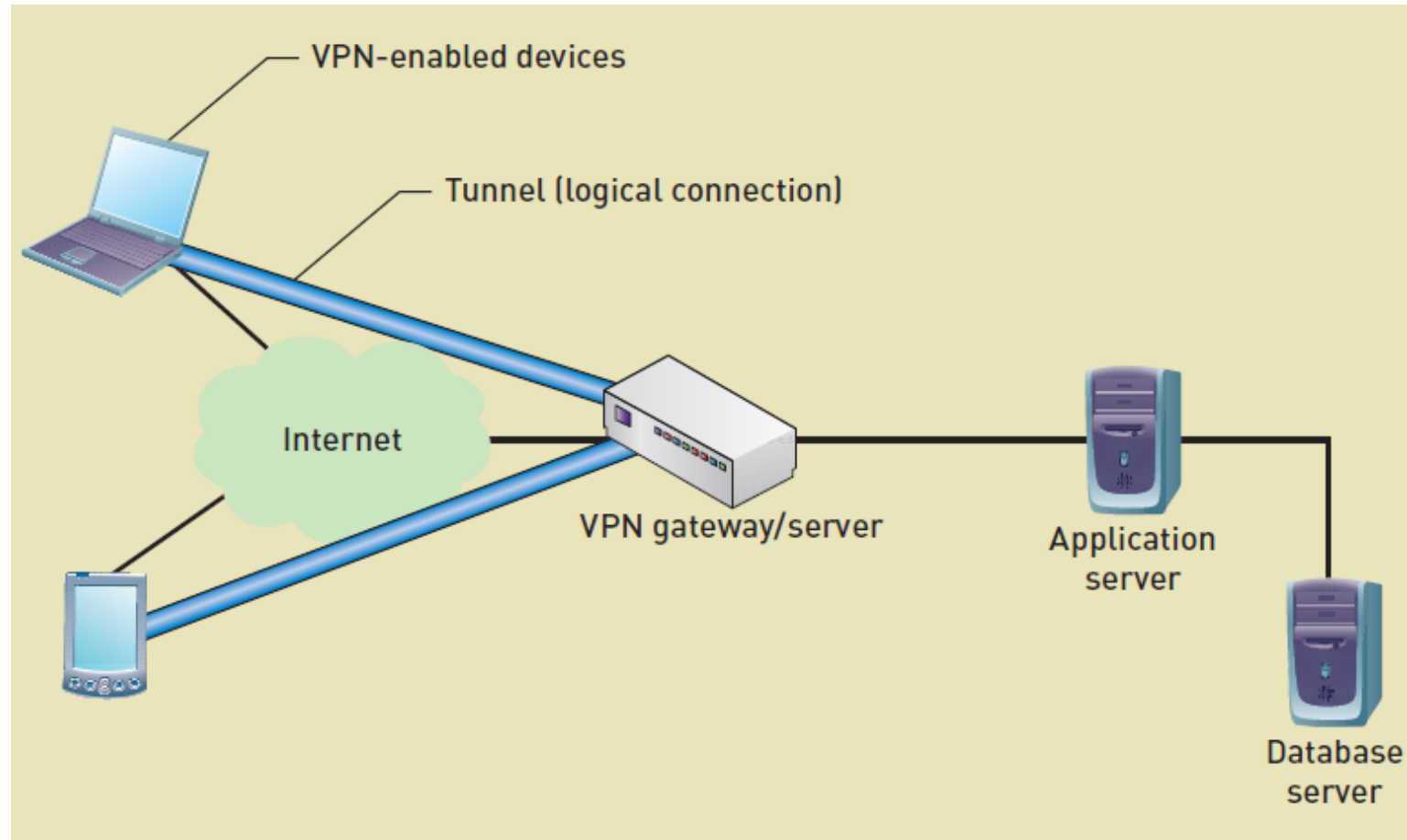


# Konfigurimi për zbatim në internet



Performanca është përmirësuar nga konfigurimet e shumëfishta të serverëve

# Rrjeti Virtual Privat (VPN)





# Alternativat e Hostimit për Zbatim në Internet (1)

---

## □ Hostimi (Hosting):

- Ekzekutimin dhe mirmbajtjen e sistemit ose aplikacionit në emer të klienti ku aplikacioni ose baza e të dhënave është. Një emertim që përdoret në sistemi cloud është 'Softuer si shërbim (software as service)' ose SaaS.
- Procesi i sigurimit të serverave fizikë në një vend të sigurt dhe shitjen e këtyre shërbimeve për bizneset e tjera që dëshirojnë të vendosin Uebfaqe

## □ Çështje kur konsideron alternativat **hostimit**

- Besueshmëria, siguria, objektet fizike, stafi, potencial për rritje



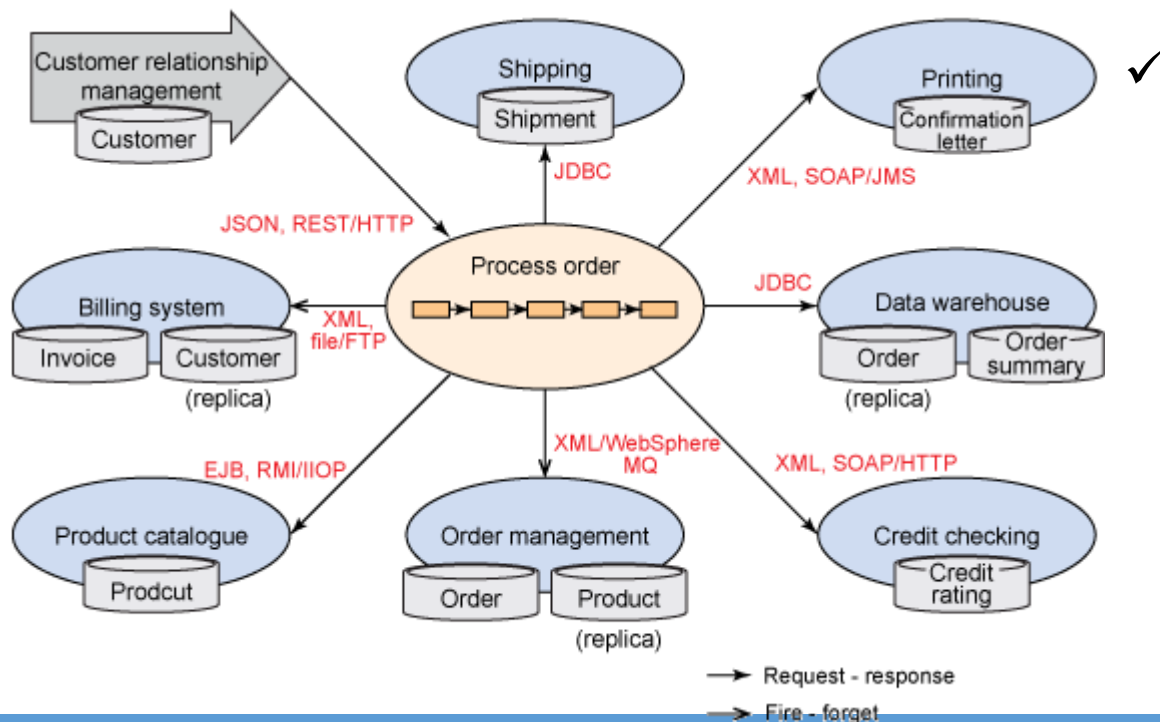
## Alternativat e Hostimit për Zbatim në Internet (2)

HOSTING OPTIONS				
Service options	Colocation	Managed services	Virtual servers	Cloud computing
Hosting service provides building and infrastructure	Yes	Yes	Yes	Yes
Client owns computer	Yes	Perhaps	No	No
Client manages computer configuration	Yes	No	Possible	No
Scalability	Client adds more computers	Client adds more computers	Client buys larger or more virtual servers	Client adds small increments of computing power
Maintenance	Client provides	Host provides	Host provides	Host provides
Backup and recovery	Client provides	Host provides	Available	Available

# Dizajni i interfaces së sistemit (1)

□ **Interfaca e sistemit** përpunojnë inputet, *bashkëveprojnë* me sisteme të tjera në kohë reale.

- Kështu që sistemet e tjera mund të flasin me njëri-tjetrin



✓ Ndërfaqet e sistemit lidhen me sisteme të tjera në mënyra të ndryshme

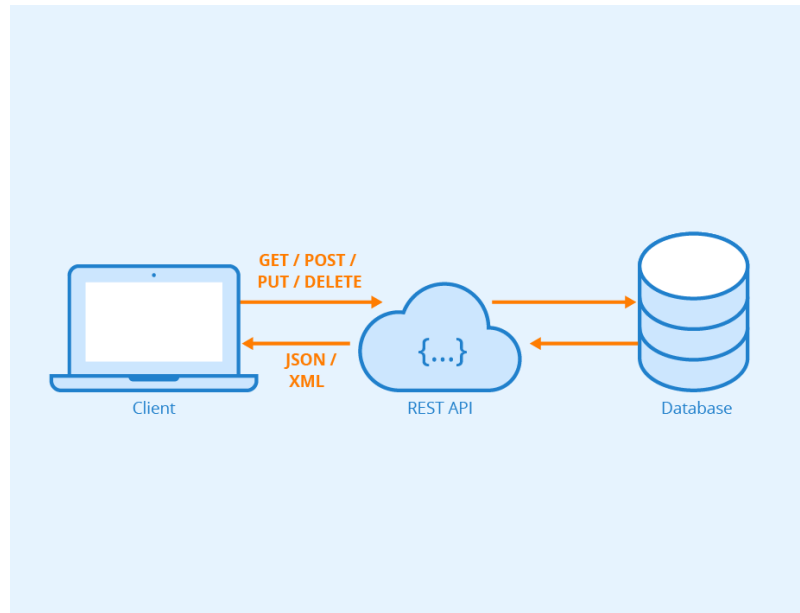
- ✓ Ruani të dhënat, sistemi tjetër e përdor.
- ✓ Lexoni të dhënat e një sistemi tjetër
- ✓ Bëne kërkesa në kohe reale për informacion.
- ✓ Shërbime të softuerike (SaaS)



# Dizajni i interfaces së sistemit (1)

## XML/JSON Interface e Sistemit

```
<inventoryRecord>
  <productItem>WS39448-7</productItem>
  <inventoryItem>48763920</inventoryItem>
  <itemCharacteristics>
    <size>large</size>
    <color>blue</color>
    <options>withzippers</options>
  </itemCharacteristics>
  <orderRules>
    <quantityOnHand>54</quantityOnHand>
    <averageCost>38.27</averageCost>
    <reorderQuantity>25</reorderQuantity>
  </orderRules>
  <dates>
    <dateLastOrder>06042012</dateLastOrder>
    <dateLastShipment>08072012</dateLastShipment>
  </dates>
</inventoryRecord>
```

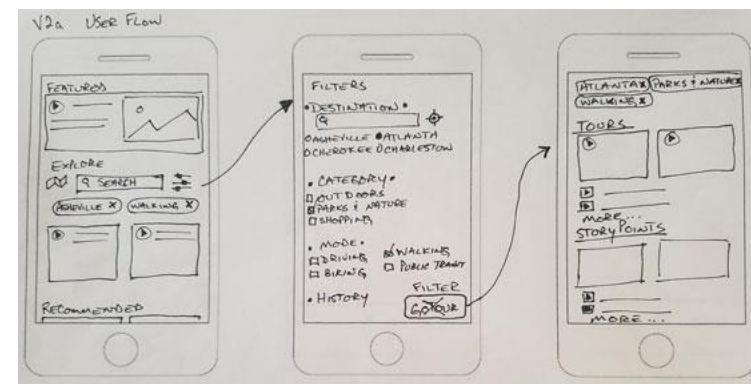
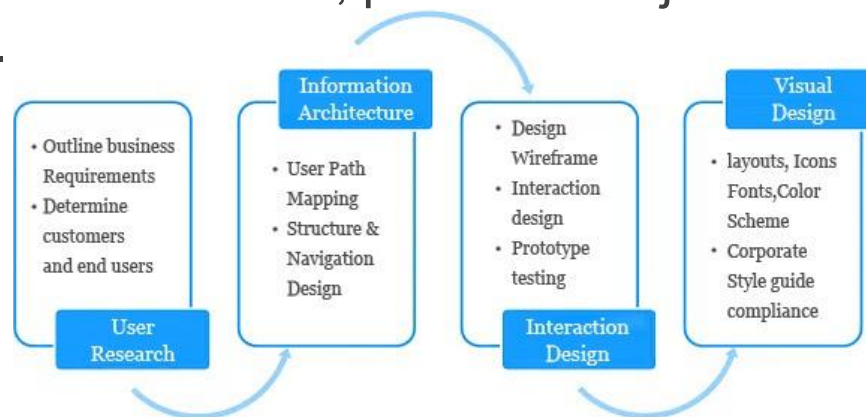


```
tutorial-azure.sql • {} employees.json x ≡ Untitled-1
1  [
2  {
3    "EmployeesId": "1",
4    "Name": "Jared",
5    "Location": "Australia"
6  },
7  {
8    "EmployeesId": "2",
9    "Name": "Nikita",
10   "Location": "India"
11 },
12 {
13   "EmployeesId": "3",
14   "Name": "Tom",
15   "Location": "Germany"
16 },
17 {
18   "EmployeesId": "4",
19   "Name": "Jake",
20   "Location": "United States"
21 }
22 ]
```

Application program interface (API) –  
The set of public methods that are available to the outside world

# Dizajnimi i interfaces së Përdoruesit

- ❑ **Dizajni i ndërfaqes së përdoruesit** ose dizajni **UI** në përgjithësi i referohet paraqitjes vizuale të elementeve me të cilët një përdorues mund të bashkëveprojë në një aplikacion, ose produkt teknologjik.
  - “për përdoruesin, intrerface është sistemi!!”
- ❑ **UI** mund të jetë **butonat**, lista **drop-down**, ose **paraqitja/shtrirja vizuale** e një faqe në Ueb.
- ❑ Modelet e **ndërfaqes së përdoruesit** jo vetëm që duhet të jenë **tërheqëse** për përdoruesit e mundshëm, por duhet të jenë **funksionale** dhe të krijuara me përdoruesit në mendje.



# Dizajnimi i Bazës së të Dhënës

□ Përdorimi i modelit të domainit të klases ose ERD

□ Arkitektura e *Bazës së të dhënave*

■ **BDH centralizuar**

- + Më e lehtë për të organizuar, modifikuar, kërkuar dhe backups
- - Mund të jetë më i ngadalshëm për shkak të shfrytëzimit/ngarkesë të lartë

■ **BDH shperndar ose distributuar**

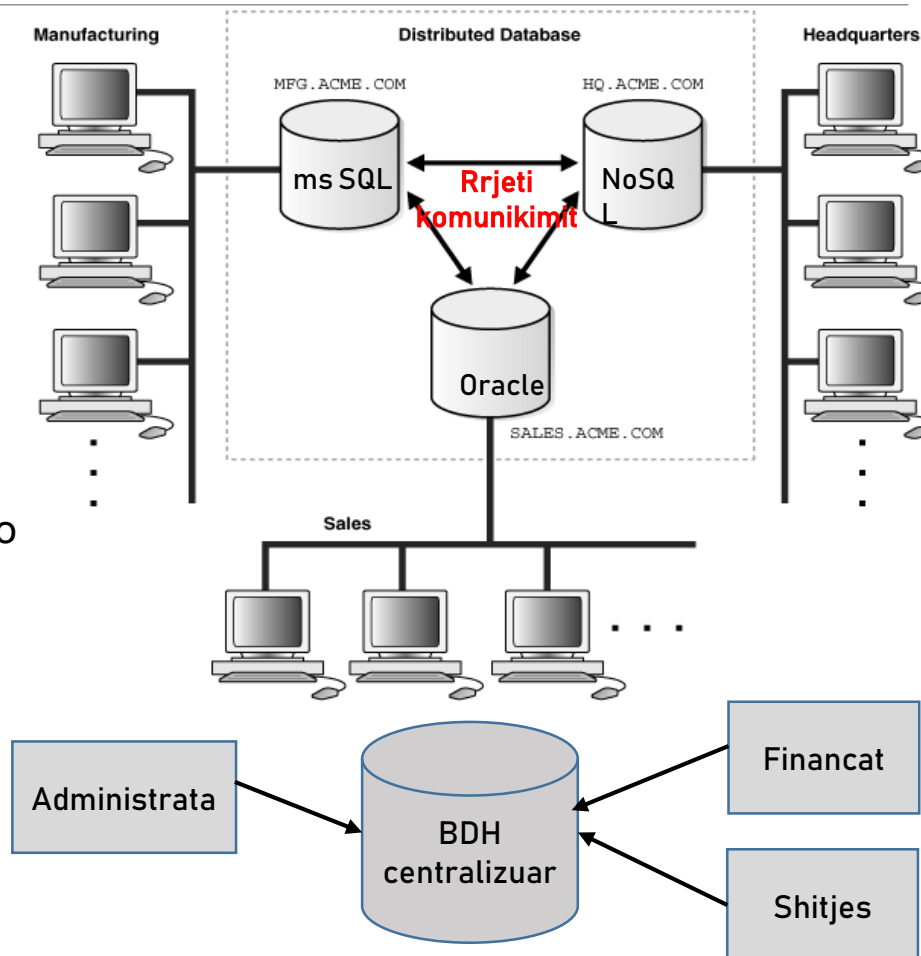
- + Qasja e të dhënave dhe kërkimi më shpejtë për system të madhë me ngarkesa të larta.
- - Mund të jetë më i ngadalshëm në përdorimin e të dhënave jo lokale
- - Duhet të siguroheni që të dhënat janë të qëndrueshme / të sinkronizuara

□ Skemat

- Tabelat dhe kolonat në relacion

□ Kufizime të integritetit

- Referencat për çelësat e huaj - për tabela lidhëse



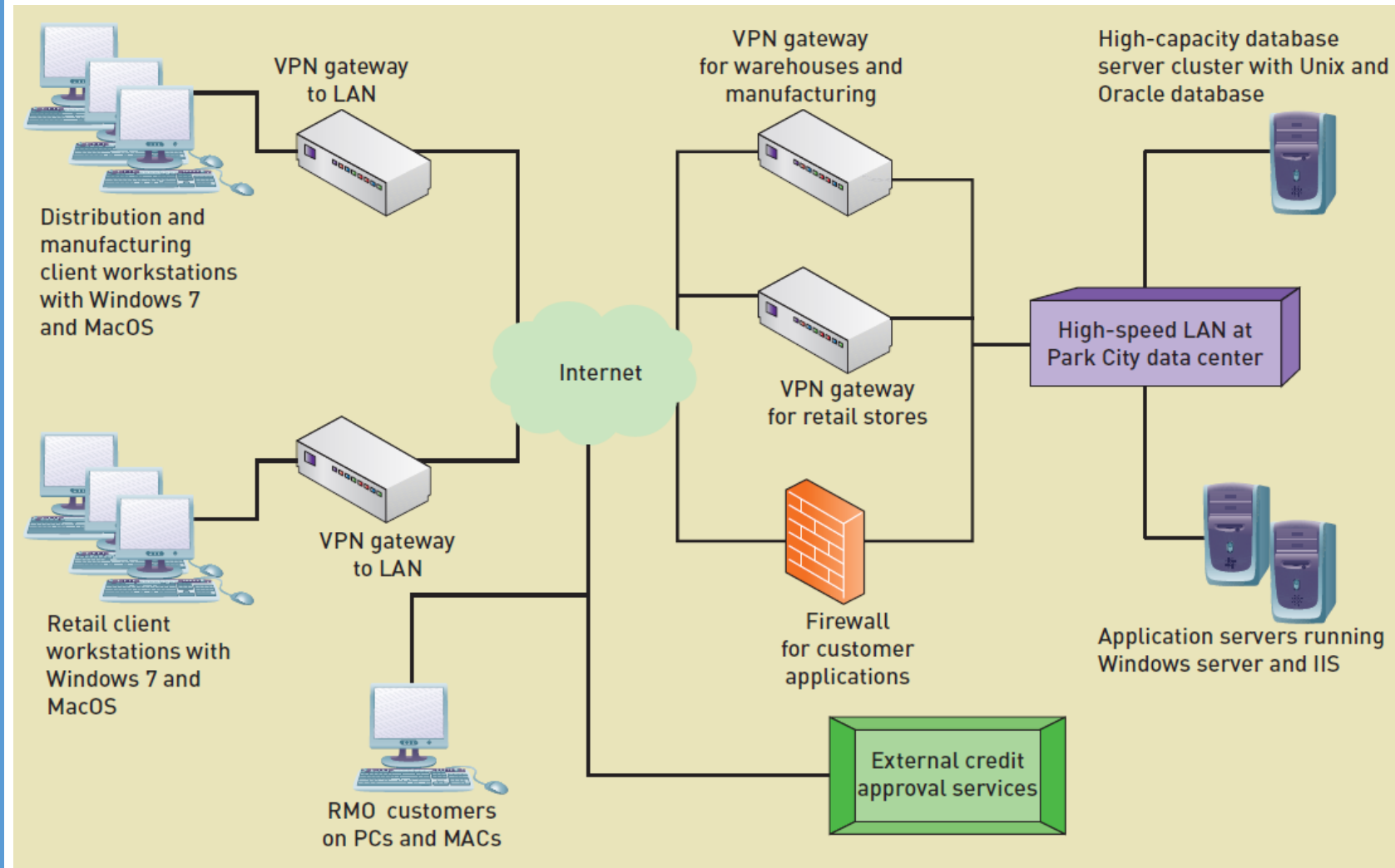


# Dizajnimi i siguris dhe kontrollit të sistemit

---

- ☐ Kontrolli i ndërfaqes së përdoruesit
  - Autorizimet e përdoruesit
- ☐ Kontrolli në nivel të aplikacionit
  - Transaksionet janë "atomike"
- ☐ Kontrollet në bazë të të dhënave
  - Nuk ka anomali të bazës së të dhënave
- ☐ Kontrollet në Rjet
  - Firewalls, qasjet

# Arkitektura aktuale e teknologjisë



# Dizajni në Agile

---

PJESA DYTË



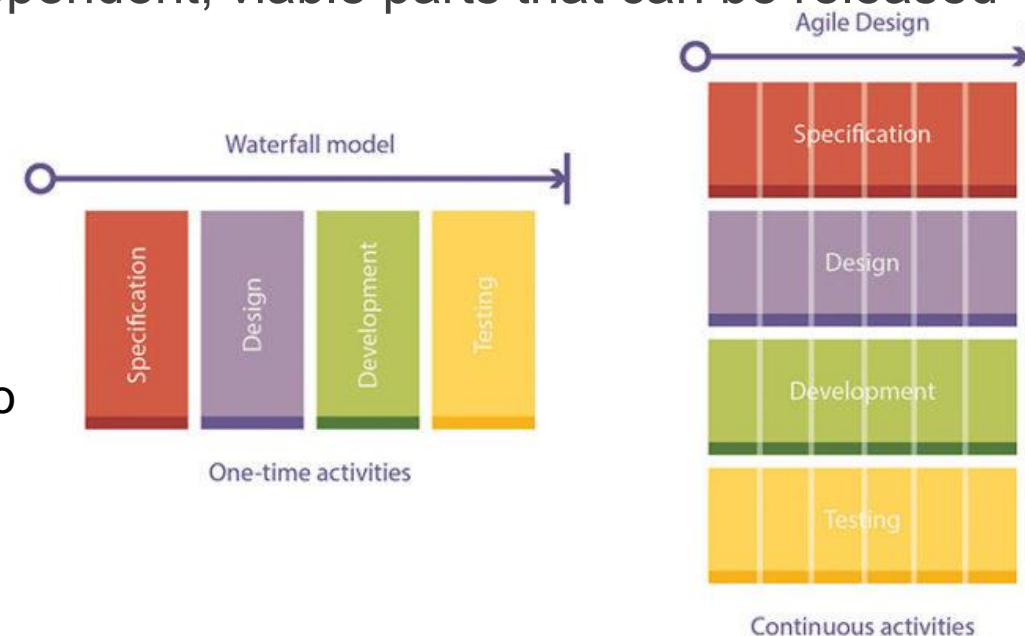
# Build design in the: Agile development processes (1)

□ In an agile environment the phases run parallel instead of following each other.

- We design, develop and test at the same time.
- We divide the product into smaller, independent, viable parts that can be released individually.

An **agile design process** allows us to use

- **iterative** as well as
- **incremental** approach to deliver design to our customer.





# Build design in the: Agile development processes (2)

---

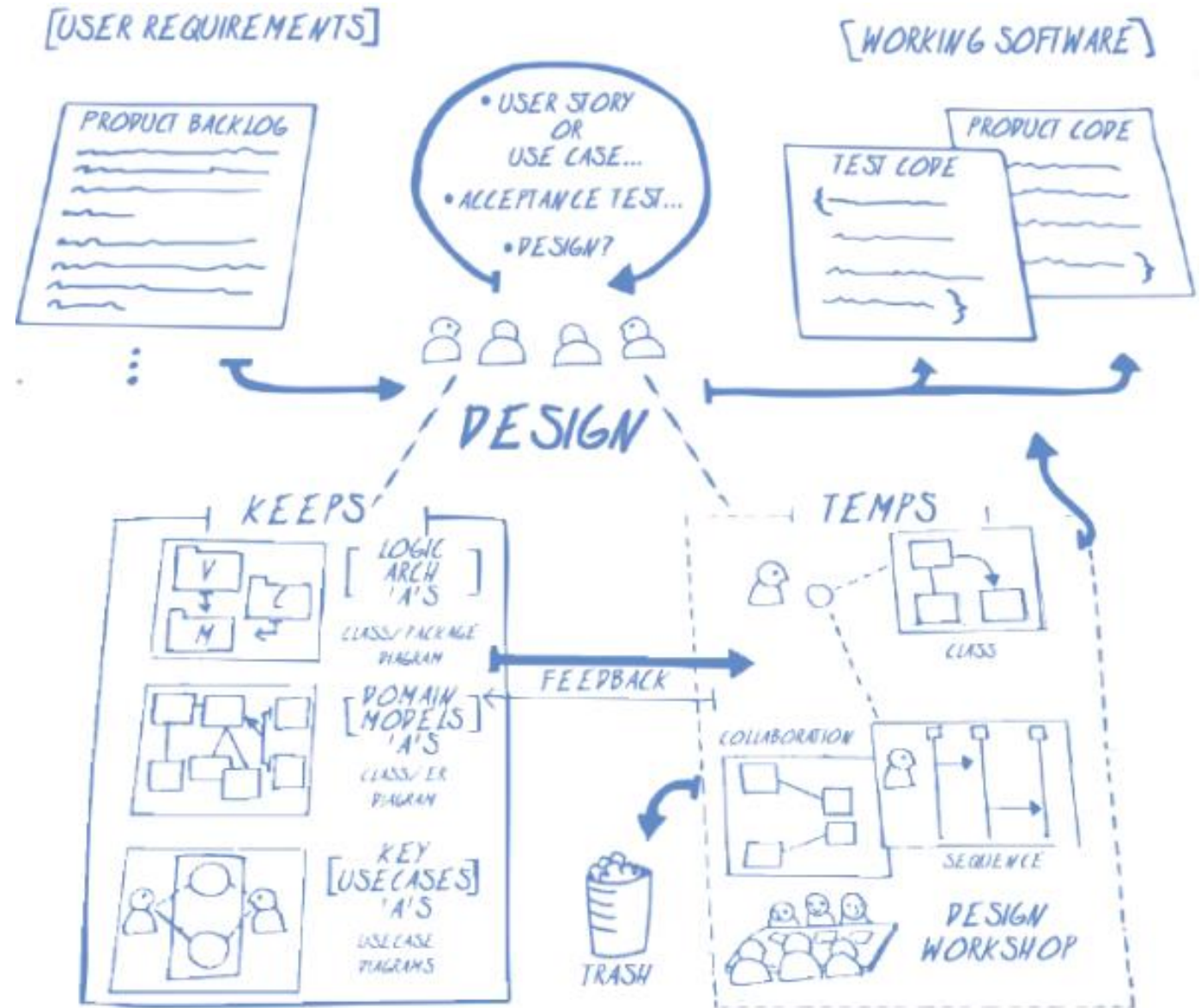
□ Recommendation of the “Big Picture” models to keep and maintain consists of:

- “**Architecture**” of the system for the team to get a rough idea of the whole system structure.
  - Architecture as: Package, Component or Deployment Diagrams
- “**Domain Model**” to help the team to understand the concepts used in the problem domain.
  - Domain Model as: Domain class or Class Diagrams
- “**Key Use Cases**” to understand the typical users of the system, and how they benefit from the system.
  - Key Use Cases as: Use Case Diagrams + Sequence/Communication Diagrams



Agile modeling with:

- 'Keeps and
- 'Temps'



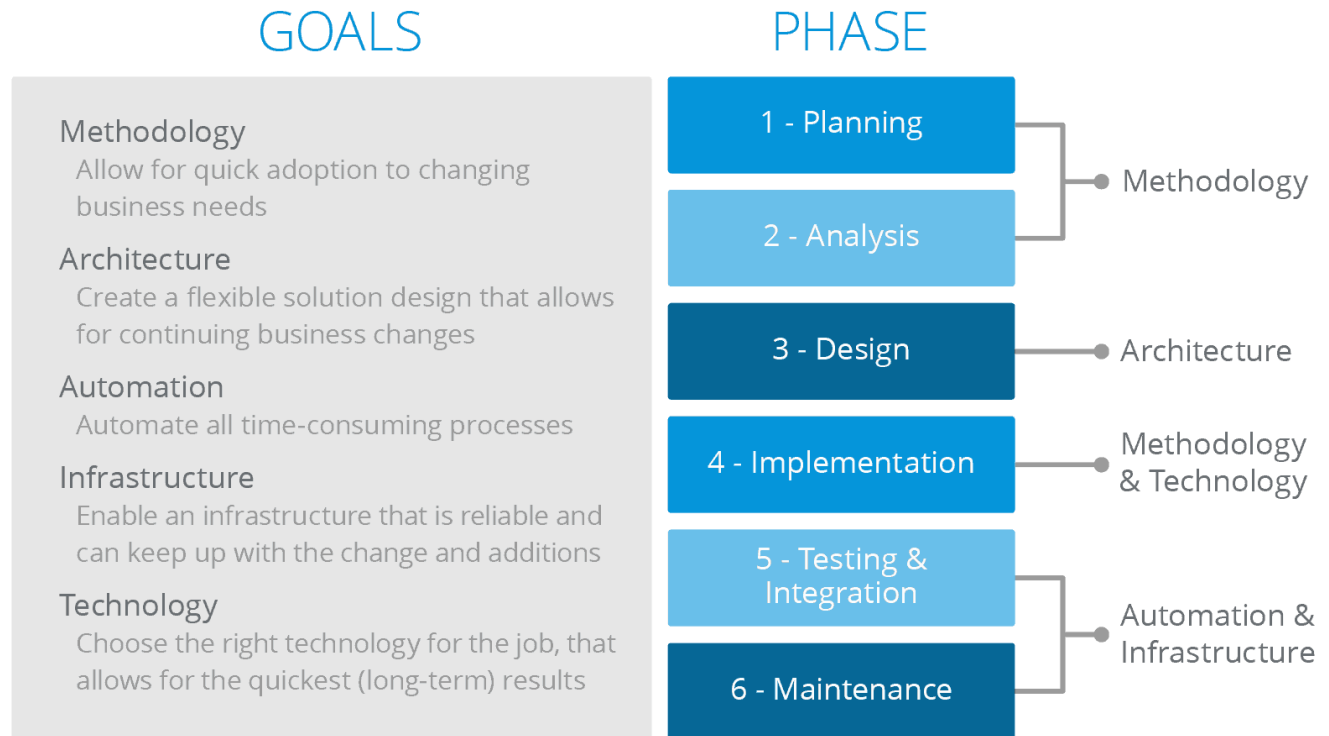


# Becoming an Agile Organization?

- ❑ Each phase has a different goal, to achieve that goal we need to focus on different areas.

We can identify **five key** areas that can enable speed and agility:

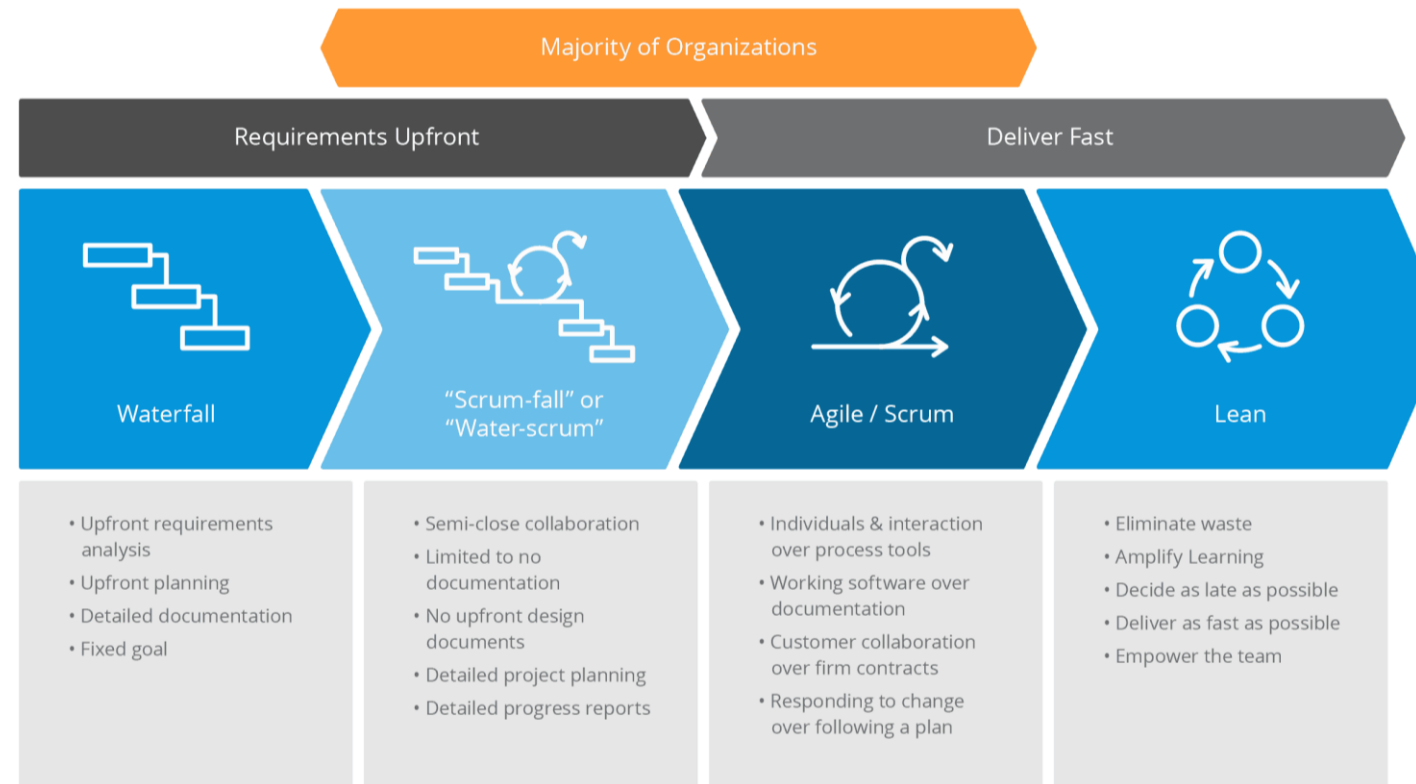
1. Methodology,
2. Architecture,
3. Automation,
4. Infrastructure, and
5. Technology.



# Methodology

- In this area, make sure to improve collaboration between business and IT and allow for quicker discovery of the requirements to ensure a quicker time to value.

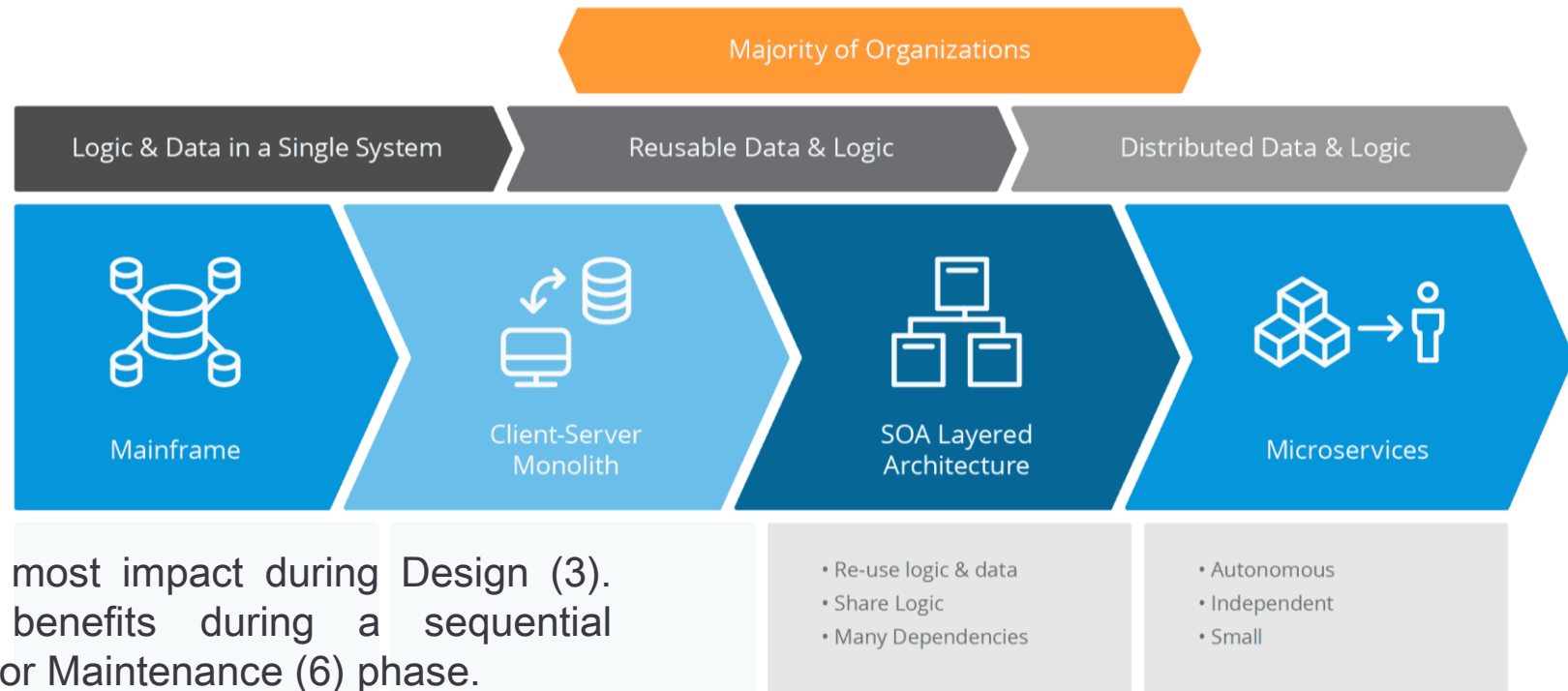
Allowing for quick adoption to changing business needs improves speed and agility during Planning (1), Analysis (2) and Implementation (4).





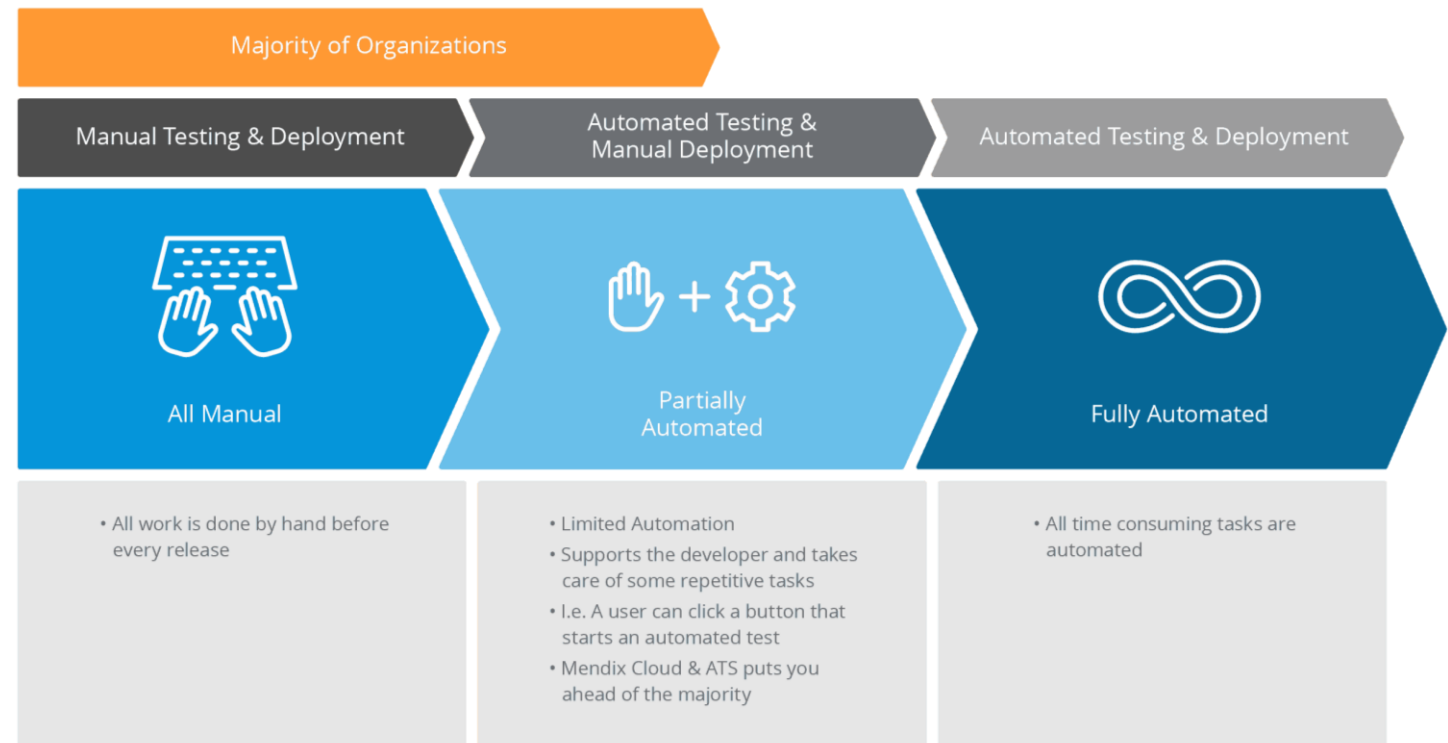
# Architecture

□ Design our solutions using the principles from a *component-based*, *MVC*, *SOA*, *microservices* architecture to create *autonomous* and *independent* systems, allowing for flexibility.



# Automation

❑ Automation reduces risk and improves speed during Testing & Integration (5).

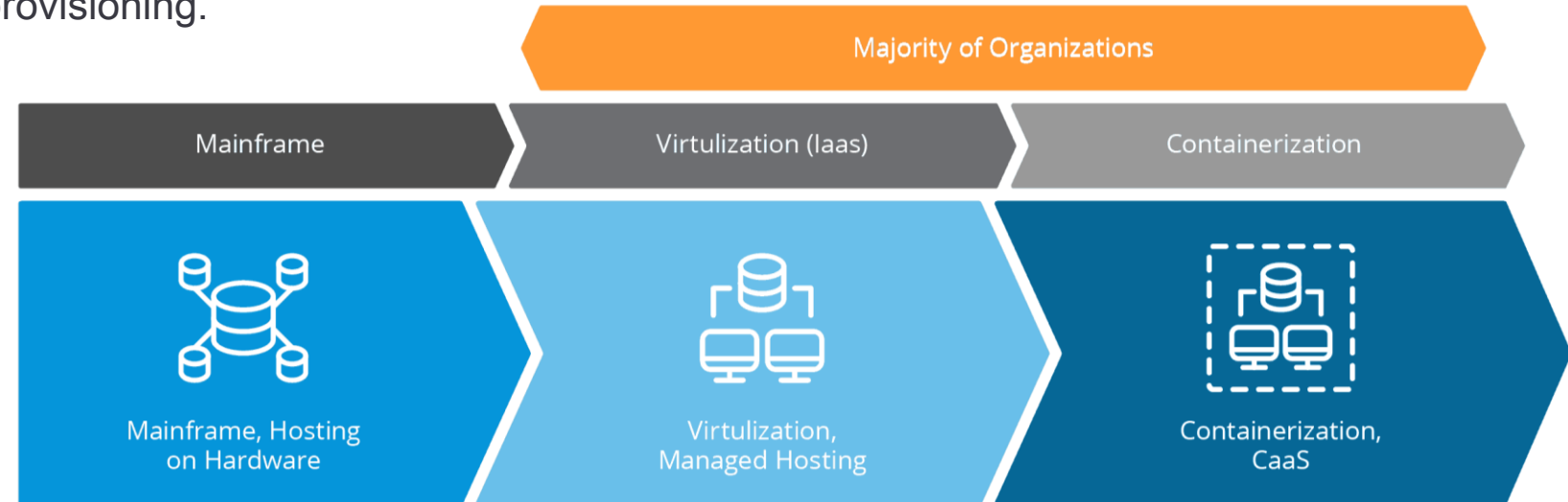


# Infrastructure

□ Flexibility in infrastructure is a must to keep up for the continuing change.

- IaaS (Infrastructure as a Service) providers already allow for immediate provisioning of servers.

Containerization which, in combination with automation, can lead to self-service and fully automated provisioning so System Operations can focus on improving rather than provisioning.



This improves speed and agility during Testing & Integration (5) and Maintenance (6).



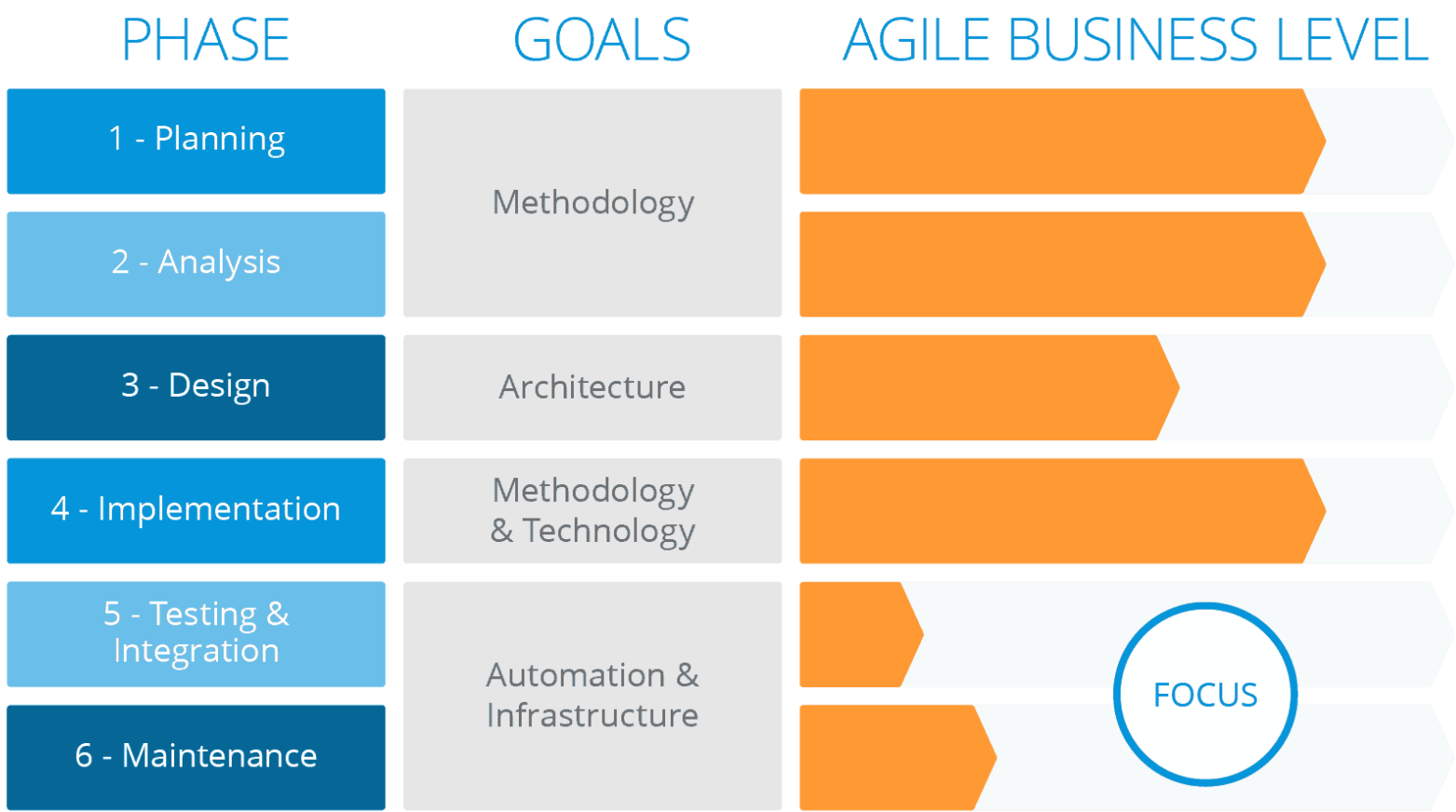
# Technology

---

- ☐ Technology is what ultimately allows your organization to be agile.
- ☐ Choose technologies that are open and can easily increase your agility, but are also capable of growing with you.



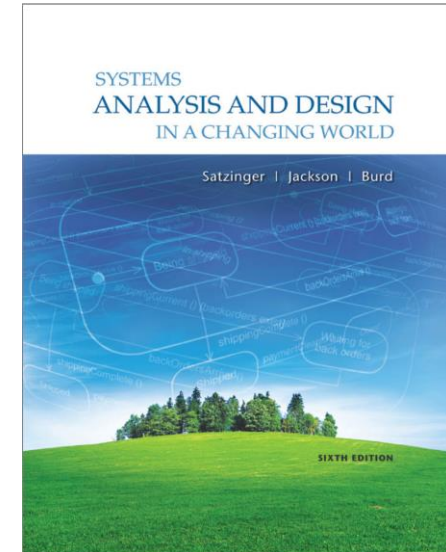
# What is Your Agility and Your Focus?





## □ Kapitulli 6: SYSTEMS ANALYSIS AND DESIGN IN A CHANGING WORLD

John W. Satzinger  
Missouri State University  
Robert B. Jackson  
RBJ and Associates  
Stephen D. Burd  
University of New Mexico



□ <https://www.mendix.com/blog/pursuing-a-full-agile-software-lifecycle/>