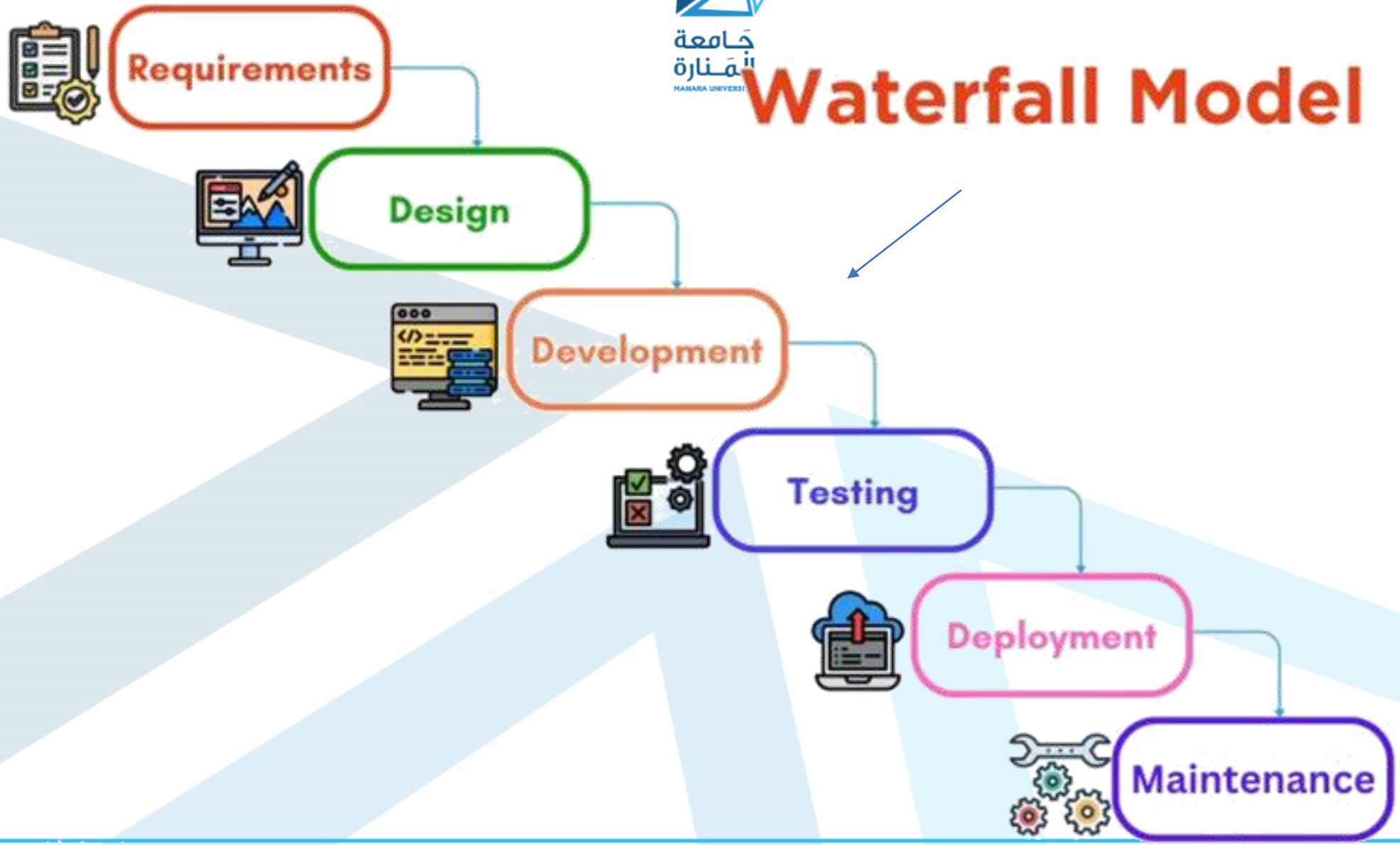
# Software Engineering -2-

## Lecture -3-

## Dr. Inas Laila

الفصل الدراسي الاول ٢٠٢٥ / ٢٠٢٦

# Waterfall Model



**Development:**

The Development phase includes implementation involves coding the software based on the design specifications.

This phase also includes unit testing to ensure that each component of the software is working as expected.

**تشمل مرحلة التطوير**

**التنفيذ (implementation): أي برمجة البرنامج بناءً على مواصفات التصميم**

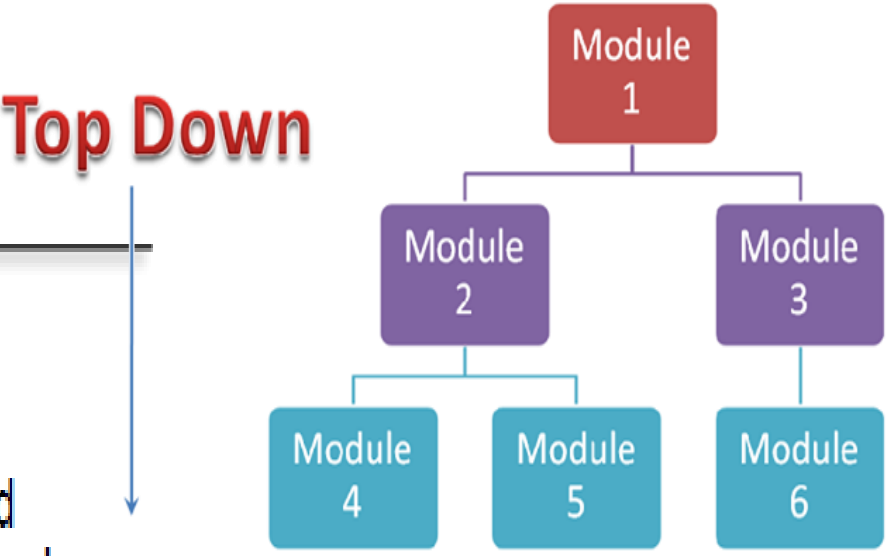
**كما تتضمن اختبار الوحدات (unit testing) لضمان عمل كل مكون من مكونات البرنامج كما هو متوقع.**

**تُعد هذه المرحلة جزءًا أساسيًا من دورة حياة تطوير البرمجيات (SDLC)، حيث تمثل التنفيذ العملي للخطط النظرية وتتويجًا للمراحل السابقة مثل جمع المتطلبات والتصميم.**

# Implementation Order

- Input, process, output (IPO) – a development order that implements input modules first, process modules next, and output modules last
- Top-down development – a development order that implements top-level modules first
- Bottom-up development – a development order that implements low-level detailed modules first
- Use-case driven – select specific use cases and order the development based on selected use cases

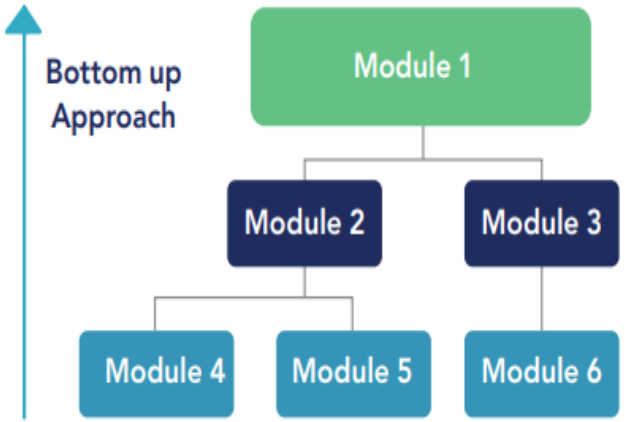
# Top-down Implementation




- A Development order that Implements top-level modules first.
  - Top-Down Approach implementation will implement the View Layer Classes and Methods first, the Domain Layer Classes and Methods next and the Data Access Layer Classes and Methods last

- ADVANTAGES
  - The primary advantage is that there is always a working version of a Program. (A partial or a complete version of a higher level program and Dummy (Stub) version of Lower level programs.)
  - Once the Topmost Classes and Methods are completed, development proceeds downwards to the next level.
  - As each method or Class is implemented, stubs for the Methods or Classes on the next lower level are added.

# Bottom-up Implementation




- Bottom-up a development Order that implements low-level detailed modules developed first and works upward to the top module.
- Bottom-Up is reverses the Top-Down Implementation Order.

- ADVANTAGES
  - The primary Advantage is that many Programmers can be put to work immediately on Classes that support a wide variety of Use Cases.
  - Programmers start writing the most difficult Classes and Methods and hence early development of  those classes allows more time for Development and Testing.

- DISADVANTAGES
  - It will require writing a number of Driver Methods to test Bottom-level Classes and Methods which adds additional complexity to the implementation and Testing process.
  - Integration and Testing for individual Use Cases are delayed because he entire System could not be assembled until the Topmost Classes are written.

# Code Refactoring

نقصد ب refactoring هو إعادة ترتيب code ما , وتنظيفه، أي مثلاً لدينا code معين و أخذنا كل فترة نعدل عليه و نضيف إليه أشياء جديدة و متحولات جديدة نتيجة لذلك ينتج لدينا العديد من المتحولات المعرفة و الغير المستخدمة و كذلك ينتج لدينا code غير مرتب و لذلك نقوم بعملية refactoring لذلك ال code لكن يجب الانتباه أننا لا نغير وظائفه أو ما يقدمه بل فقط نقوم بعملية تهذيب لل code .

فلا نعرف متحولاتنا بأسماء عامة مثل x و y و z .. بل

كل متحول و كل method يكون اسمها يعبر عنها و كذلك نحاول وضع comments داخل ال code لتسهيل عملية الفهم .

إعادة هيكلة الكود هي عملية إعادة هيكلة الكود الحاسوبي الحالي دون تغيير سلوكه الخارجي.
الهدف الرئيسي هو تحسين البنية الداخلية للكود، وسهولة قراءته، وسهولة صيانته، مما يسهل فهمه وتوسيعه وتصحيح أخطائه.
لا تضيف إعادة الهيكلة ميزات جديدة أو تغير وظائف البرنامج بل تساعد إعادة الهيكلة المنتظمة على الحفاظ على نظافة الكود واستدامته بمرور الوقت.

جَـامعة
المَـنارة
MANARA UNIVERSITY

# Deployment (SDLC)

في هندسة النظم، يشير النشر إلى الأنشطة الأساسية التي ينطوي عليها تحويل نظام مُطوّر إلى الاستخدام التشغيلي، وضمان قبوله تشغيليًا، ونقل مسؤوليات تشغيله بفعالية وكفاءة وأمان إلى المالك.

تشمل هذه العملية نقل قدرة النظام إلى المستخدم النهائي، وتسليم مسؤوليات الدعم والصيانة إلى جهة دعم ما بعد النشر. قد يشمل نشر النظام اختبارًا تجريبيًا للموثوقية، والتخلص التدريجي من الأنظمة القديمة التي يحل النظام الجديد محلها.

**Deployment:** Once the software has been tested and approved, it is deployed to the target environment

Software deployment is the process of making software available for use on a system by users and other programs, encompassing activities such as installation, configuration, testing, and ensuring the software runs effectively in its target environment

يتضمن عملية استثمار الـ Application الجديد ضمن الشركة ويتم ذلك بعدة طرق :

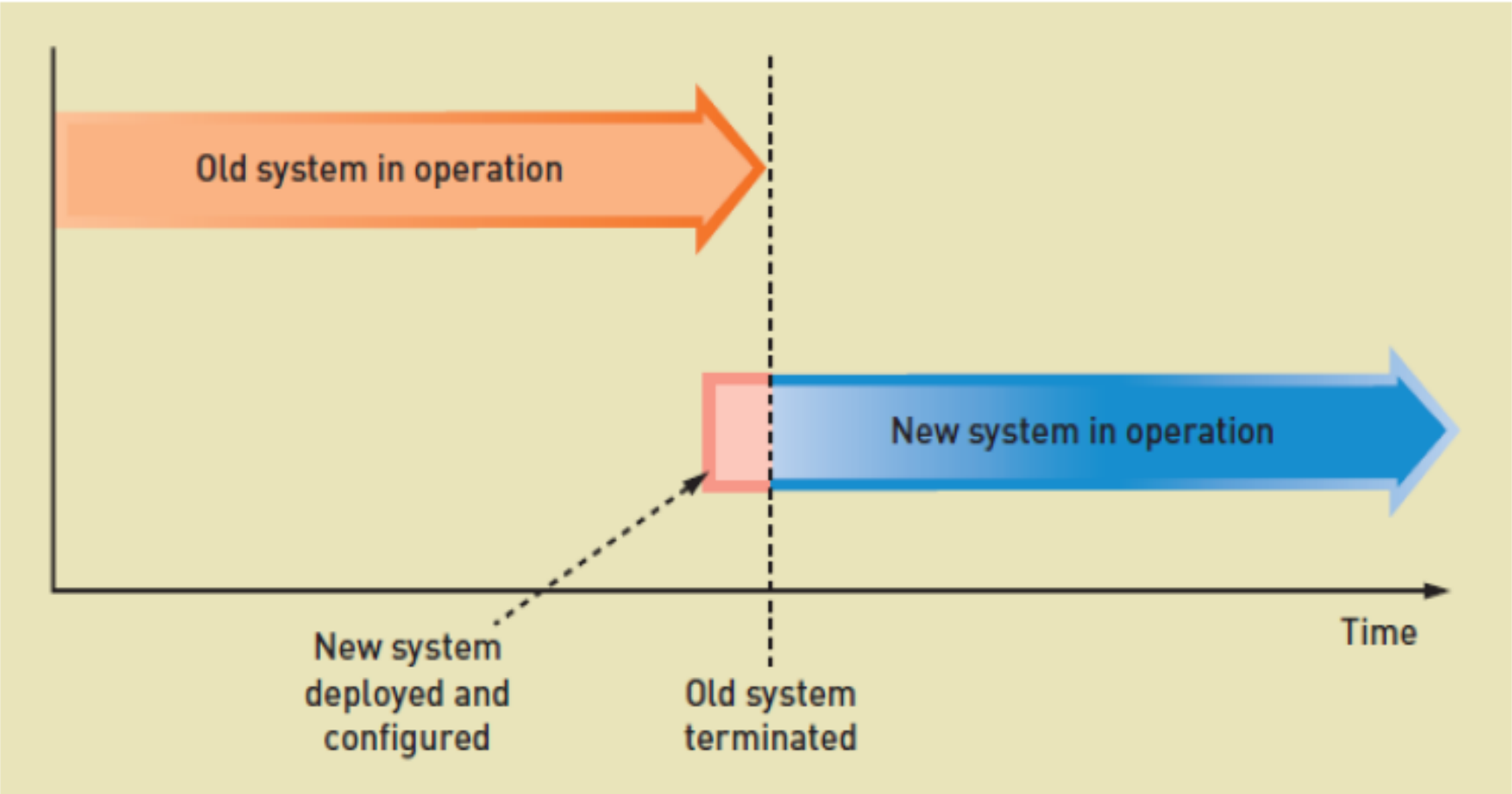
1. Direct Deployment :

نتخلص بشكل نهائي من النظام القديم ونضع النظام الجديد ولذلك العديد من المخاطر حيث أنه إذا اكتشفنا وجود عطل ما فإن ذلك سيعطل الشركة عن العمل وستكون الشركة "بفترة إيقاف" وعدم استثمار للـ Data , ولكن في المقابل تعتبر هي الأرخص لأن الأشخاص الذين يعملون على النظام القديم سيعملون على النظام الجديد .

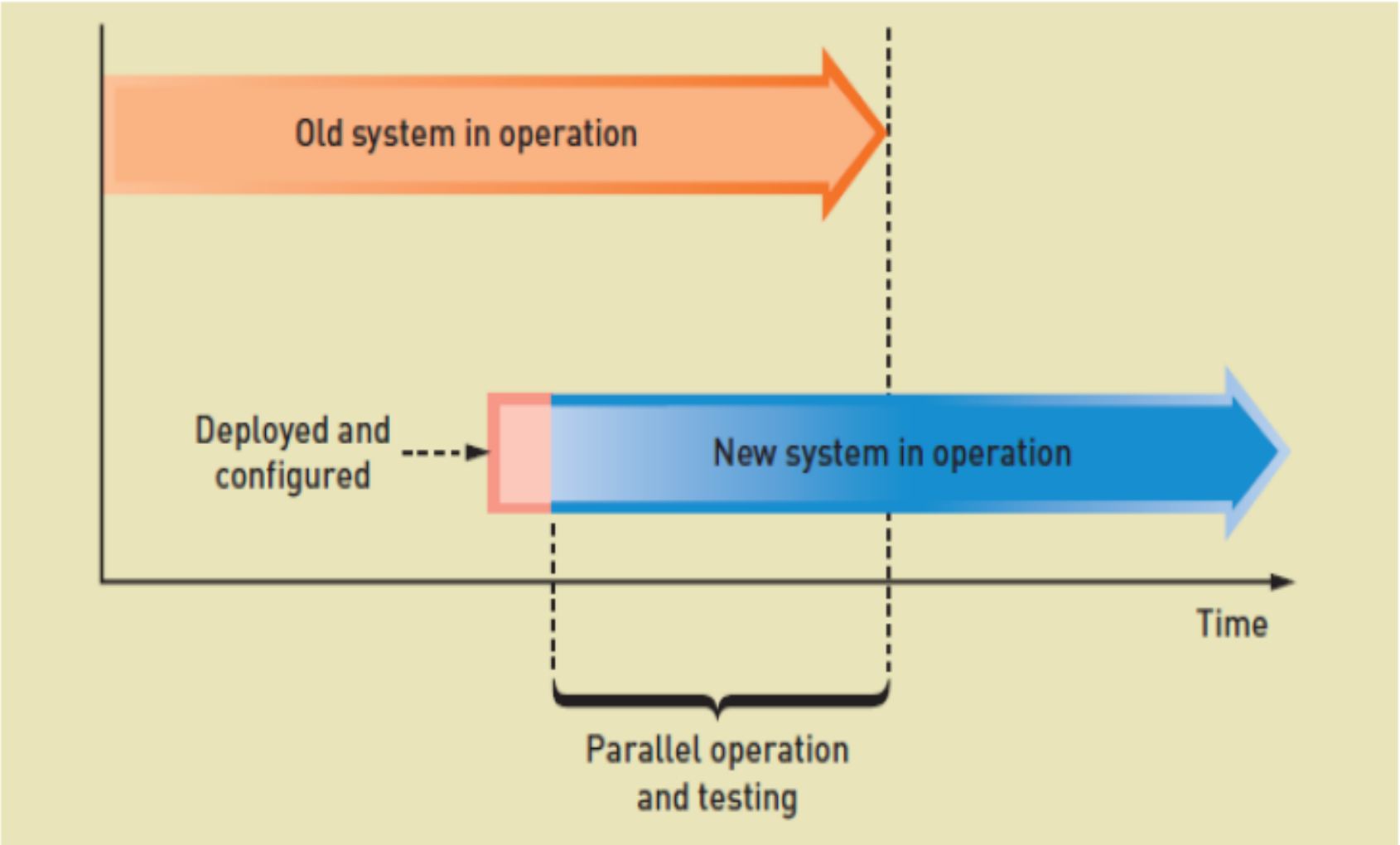
2. Parallel Deployment :

وهو الحل الأضمن لأن النظام القديم موجود ريثما يتم التّحقق من أنّ النظام الجديد يعمل بشكل جيد حينها سنقوم بسحب النظام القديم و وضع الجديد مكانه , وهذا الحل الأقل خطورة لكن بالمقابل ستكون التكلفة عالية لأن النظامان يعملان بالوقت نفسه حيث أنه عند قدوم Request ما فإنه سيأتي على النظام القديم وعلى النظام الجديد بنفس الوقت بالتالي سيكون لدينا شخصان يتعاملان مع الـ Request نفسه بالتالي ذلك سيكلفنا Hardware أكبر وعدد أشخاص أكبر و وقت أكبر وكلفة أكبر .

# Direct Deployment



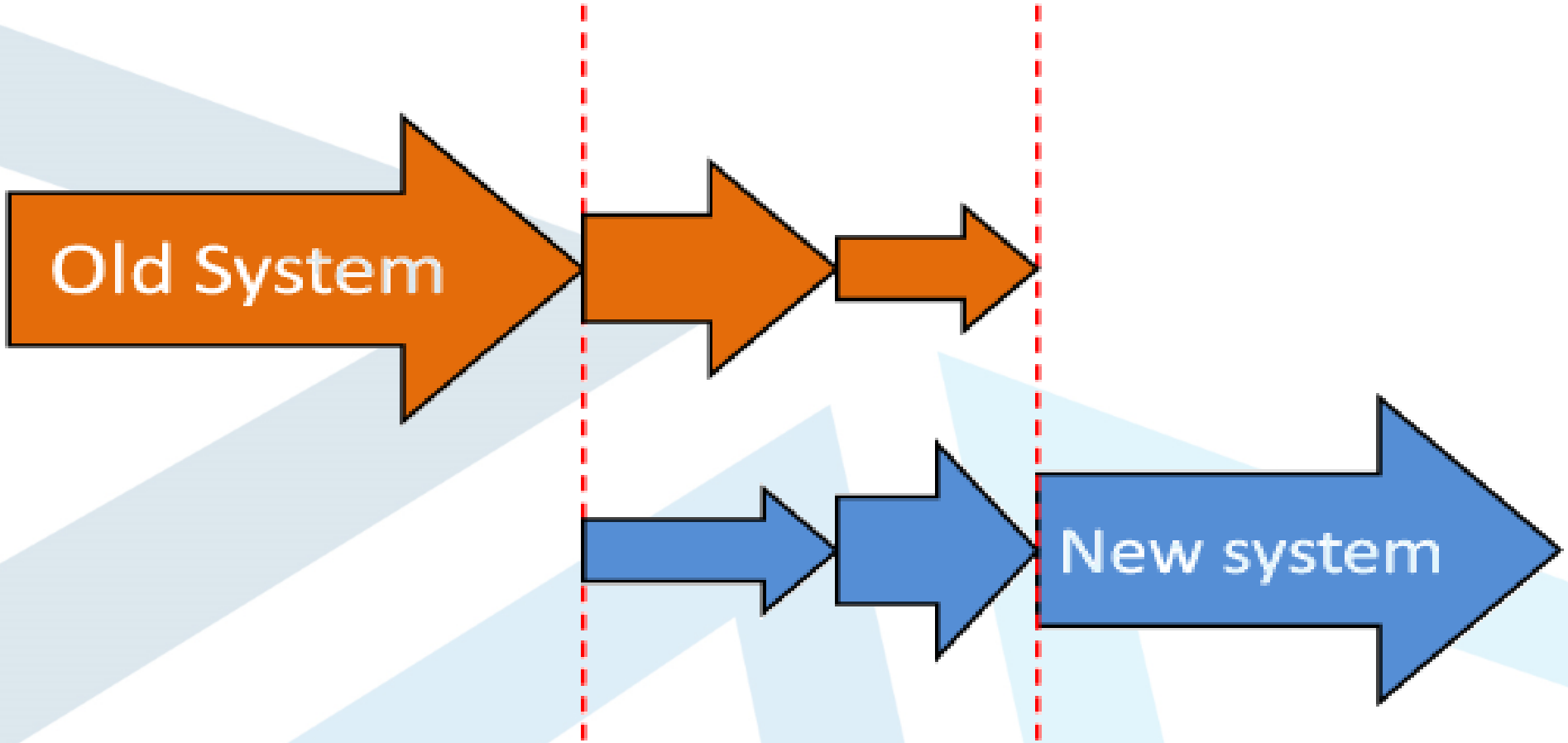
# Parallel Deployment



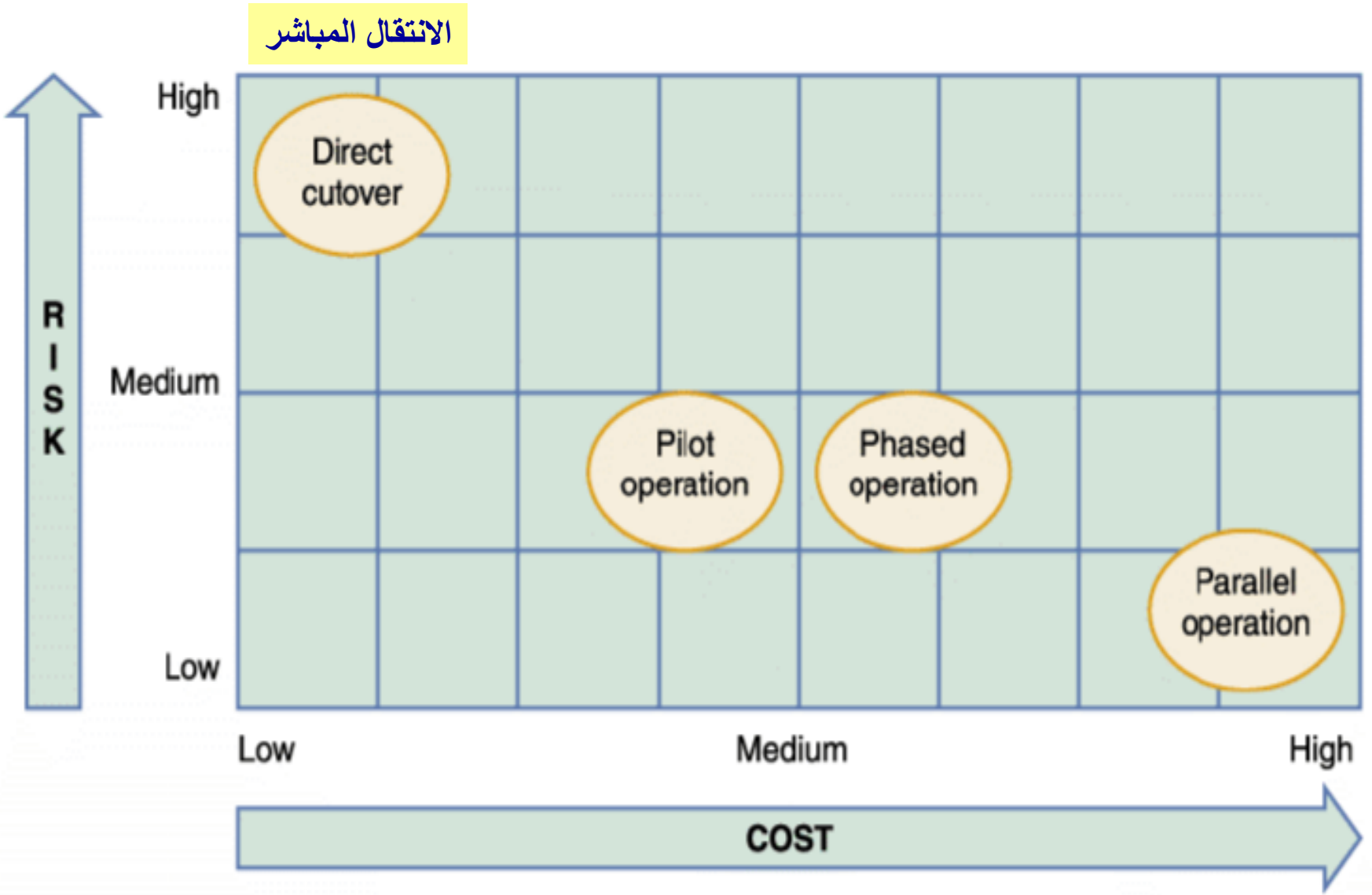
## 3. Phased Deployment (Step By Step)

هنا سيكون لدينا مجموعة وظائف محددة نعطيها لموظفين نموذجيين مهمتهم التأكد من هذا الجزء أنه يعمل بشكل صحيح ثم نقوم بعمل Replacement بين هذه الجزئية والجزئية القديمة "على Data من الشركة " لكن تكمن المشكلة هنا هو بصعوبة الفصل بين الأجزاء , وستأخذ فترة طويلة جداً لاستبدال كامل لكل جزء بين النظامين القديم والجديد , وتكلفة هذه الطريقة تعتبر أقل من الـ Parallel وأكبر من الـ Direct

أي يمكننا القول بأن الـ Risk قليل ومضبوط والتكلفة قليلة لكن الزمن اللازم للحصول على النظام الجديد كبير جداً .


Old System
New system

# Deployment Approaches Summary

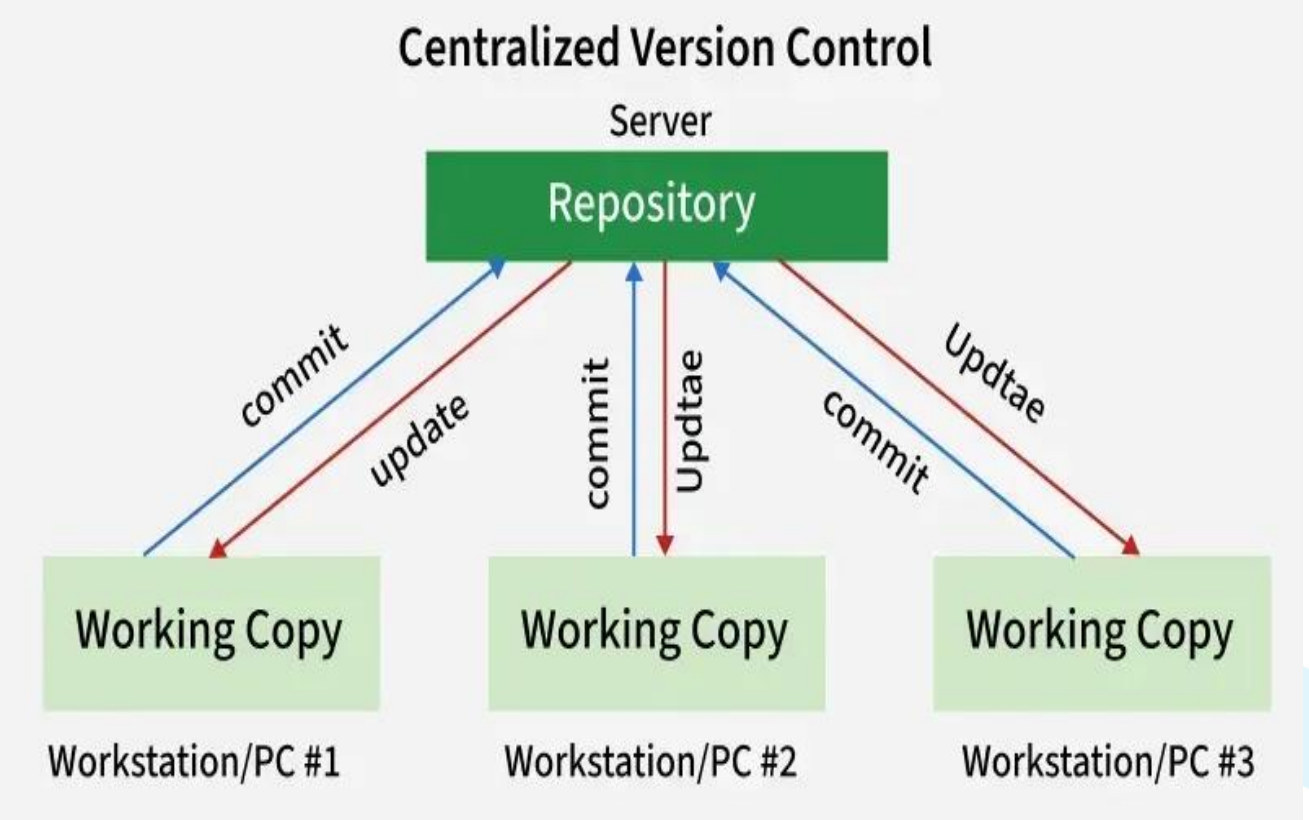



A pilot in software development is **to release built software to a small set of customers in a controlled fashion**. The pilot is the deployment of production software but prior to a full release of the software to all customers.

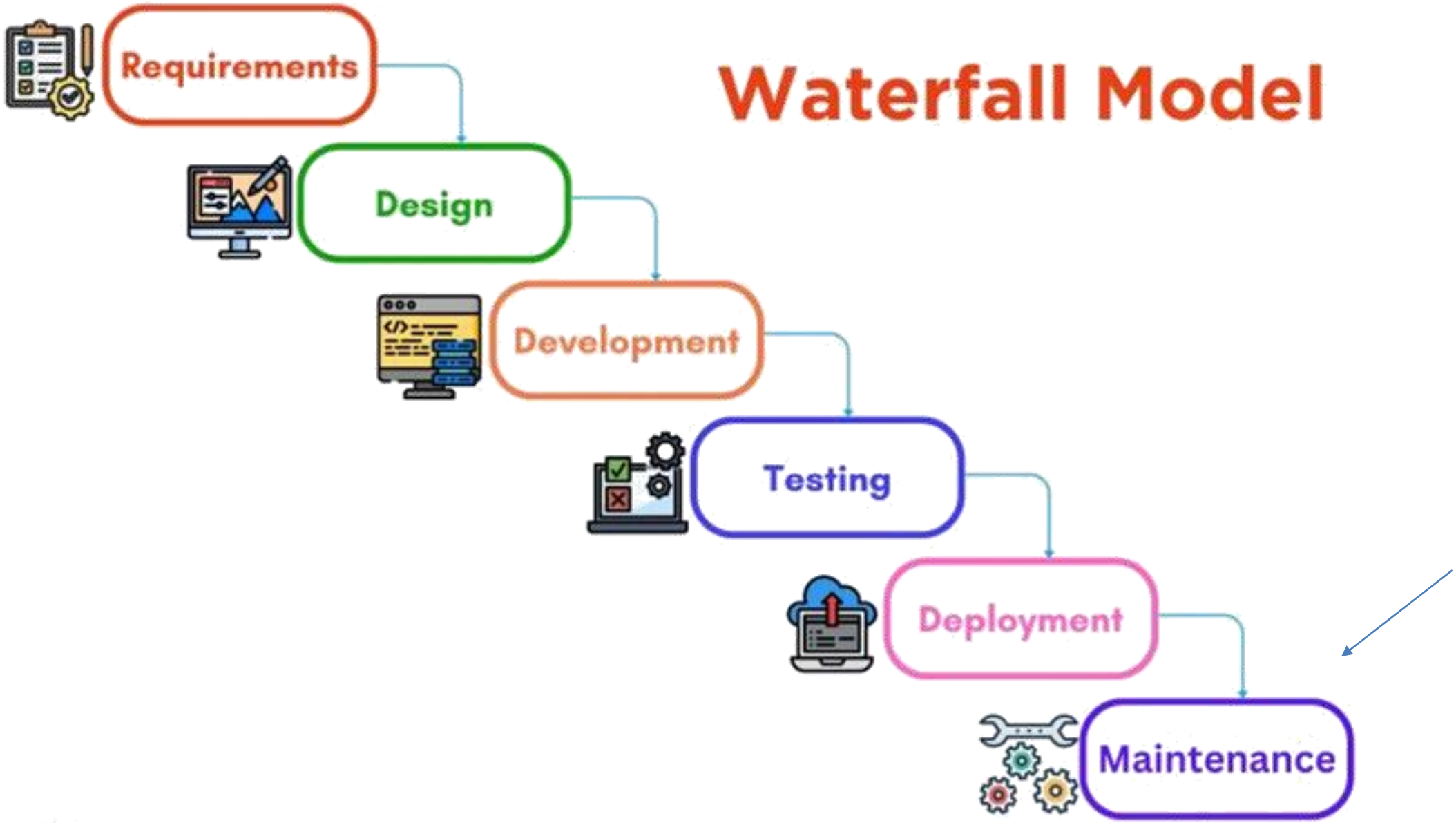
# Release vs Version


In software engineering, "release" and "version" are related but distinct concepts. A version refers to a specific, identifiable state of software, typically labeled with a version number such as 1.0 or 2.1.5, which signifies a particular iteration of the software with specific features, bug fixes, or improvements.

A release is a version of the software that has been formally distributed or made available to users, often after testing and quality assurance.
Not all versions are released; only those deemed stable and ready for public use are considered releases.
For example, a development team might create multiple builds and versions internally, but only the final, tested version that is deployed to users is labeled a release.



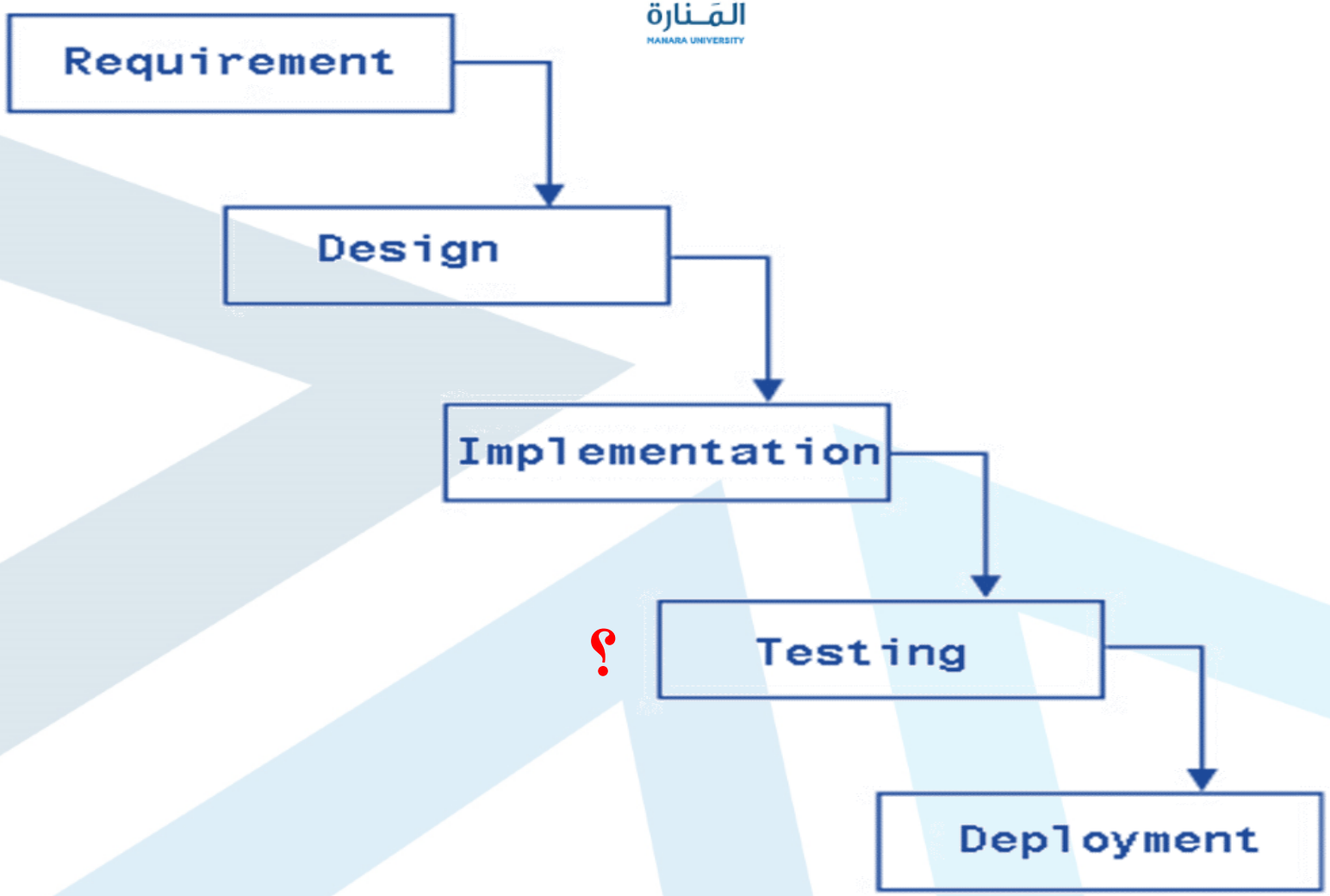
# Waterfall Model

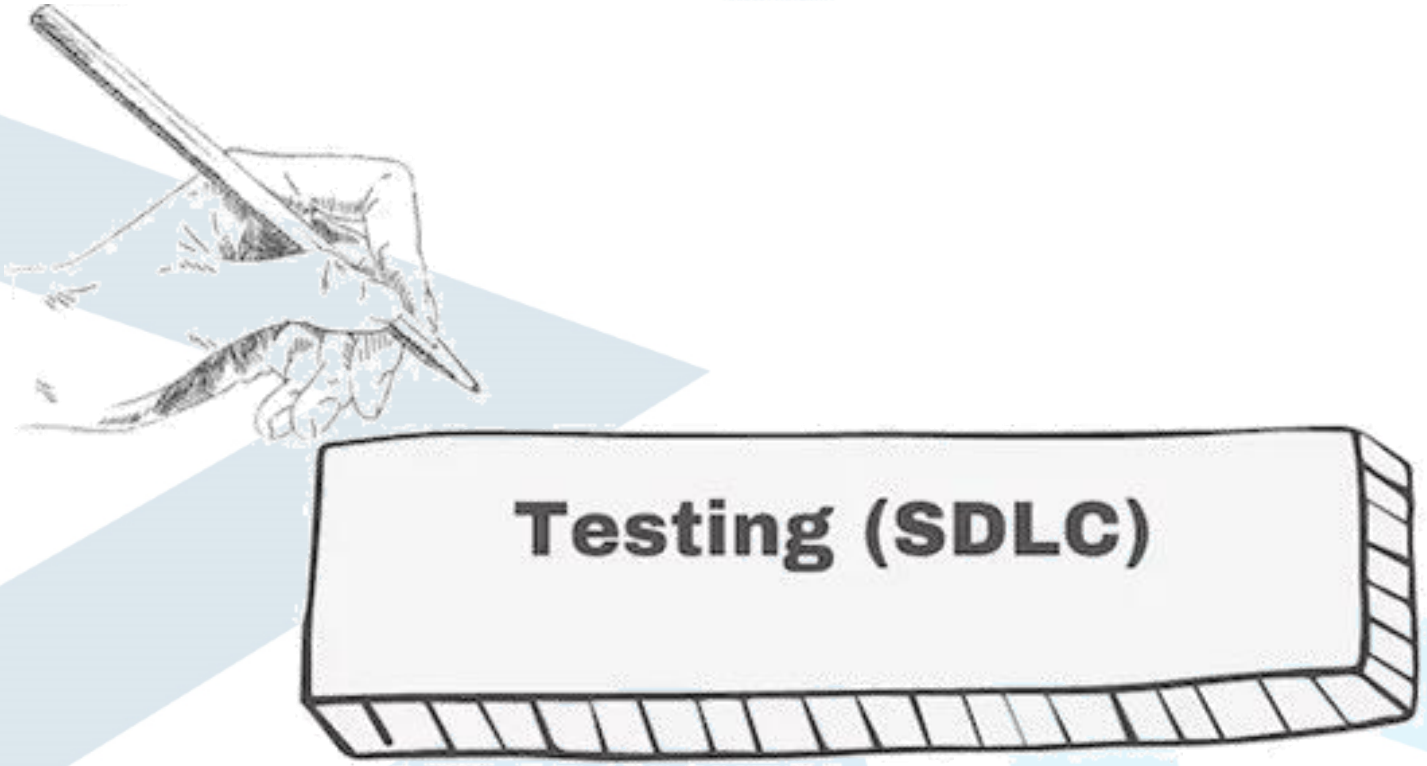


## Maintenance:

The final phase of the Waterfall Model is maintenance, which involves fixing any issues that arise after the software has been deployed and ensuring that it continues to meet the requirements over time.

**الصيانة: المرحلة الأخيرة من نموذج الشلال هي الصيانة، والتي تتضمن إصلاح أي مشكلات تنشأ بعد نشر البرنامج والتأكد من استمراره في تلبية المتطلبات بمرور الوقت.**


Requirement
Design
Implementation
Testing
Deployment

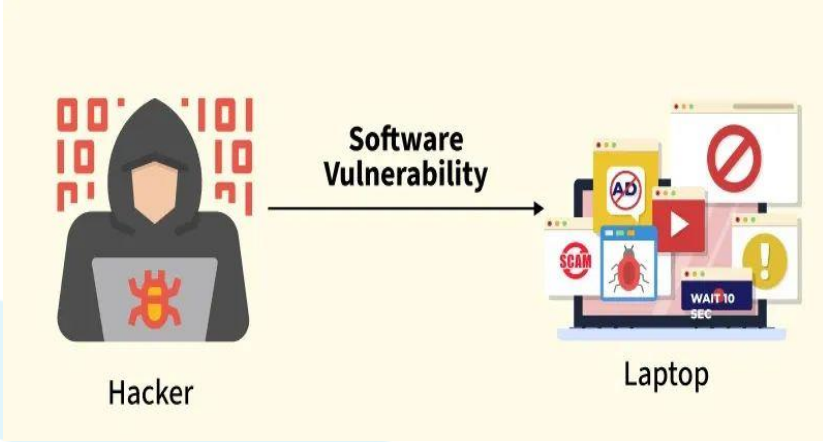



الاختبار: في مرحلة الاختبار، يتم اختبار البرنامج ككل للتأكد من أنه يلبي المتطلبات وخالٍ من العيوب

# Introduction To Software Testing

## Basic Testing Terminologies

1. Defect
2. bug
3. Vulnerability
4. Review




الأخطاء في البرامج التي يمكن أن يستغلها المتسللون

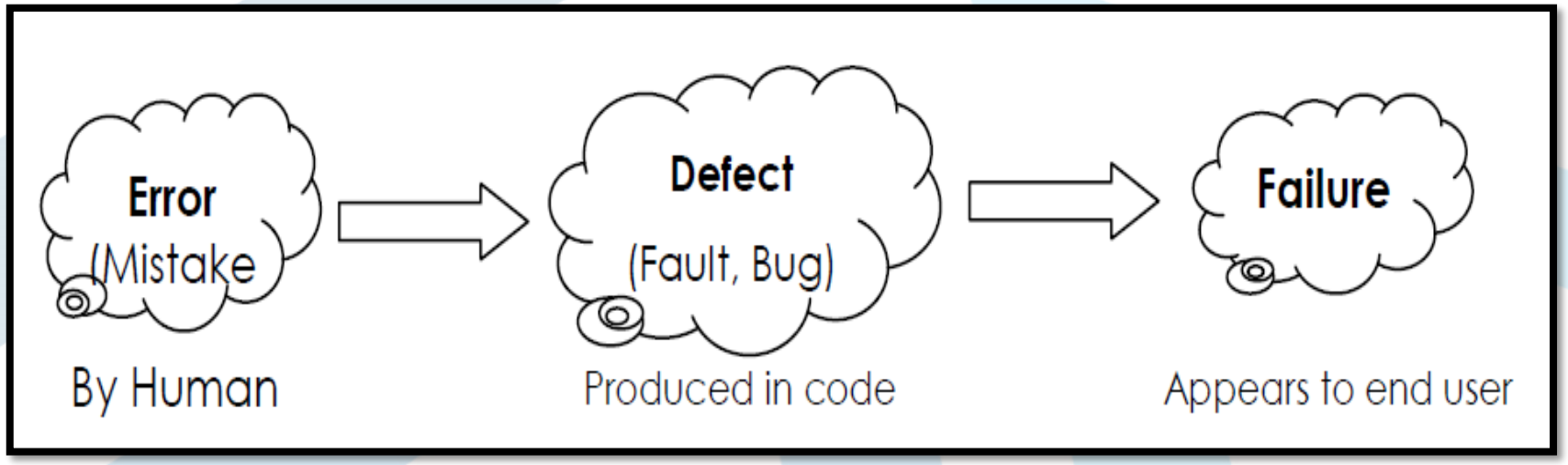
Vulnerability: bugs in software that can be exploited by a malicious actor to gain unauthorized access, disrupt services, or to get sensitive data. It is commonly defined as a weakness in the computational logic (e.g., code) found in software components that, when exploited, results in a negative impact to confidentiality, integrity, or availability.

# Software Defects

A human being can make an error (mistake), which produces a defect (fault, bug) in the program code, or in a document.

If a defect in code is executed, the system may fail to do what it should do (Or do something it shouldn't), causing a failure.

Defects in software, systems or documents may result in failures, but not all defects do so.



مثال(1) لتوضيح ال errorو fault و failure:

ليكن لدينا الكود التالي المكتوب بلغة الباسكال

| LOC | Code |
|---|---|
| 1 | program double (); |
| 2 | var x,y: integer; |
| 3 | begin |
| 4 | read(x); |
| 5 | y := x * x; |
| 6 | write(y) |
| 7 | end |

البرنامج الذي نريده هو حساب ضعف قيمة ما أي القيمة * ٢ ، لكن لاحظ المبرمج ماذا كتب بالسطر الخامس y:=x * x
أي أنه أخطأ في فهم البرمجية المطلوبة واوجد مربع قيمة عوضا عن ضعفها ،إن هذا الخطأ الذي ارتكبه المبرمج يعتبر error، وعند إسناد قيمة x*x إلى y يعتبر ذلك fault في البرنامج وهو الخطأ في الـ code الناتج عن الـ error الذي ارتكبه المبرمج
أما عند تنفيذ البرنامج والحصول على قيمة غير منطقية عندها يسمى ذلك failure أي فشل بالنظام
لاحظ هنا أننا لو أدخلنا قيمة ٢ نجد أن الناتج سيكون صحيحاً ولن ينتج لدينا Failure ولذلك قلنا أنه يجب علينا تجريب أكثر من عملية testing بأرقام و data مختلفة للتأكد تماماً من عدم وجود Failures في النظام

```
public static int numZero (int [ ] arr)
{  // Effects: If arr is null throw NullPointerException
   // else return the number of occurrences of 0 in arr
   int count = 0;
   for (int i = 1; i < arr.length; i++)
   {
      if (arr [ i ] == 0)
      {
         count++;
      }
   }
   return count;
}
```

بهذا المثال نحن نقوم بإدخال عناصر مصفوفة ونقوم بعد الأصفار الموجودة في المصفوفة فلاحظ أن المبرمج قد أخطأ فبدل من أن يقوم بالبدأ بمسح المصفوفة من الـ index (0) بدأها من الـ index (1) .

ويعتبر هنا الخطأ الذي أخطأه المبرمج هو error ويعتبر تعليمة الإسناد i=1 هي الـ fault الكامن في البرنامج ، وعند تنفيذ البرنامج وإدخال data معينة (إذا كانت أول قيمة في المصفوفة ليست صفراً فالنتائج ستكون صحيحة ولن يعطي Failure لكن لو كانت أول قيمة في المصفوفة صفراً عندها لن يعطي النتائج الصحيحة وسيعتبر Failure)

نلاحظ أن ال fault هو خطأ كامن في البرنامج أما لـ failure فهو ما نلمسه عند التنفيذ لكن ربما test_case معين سيء لم يستطع أن يكشفه لنا وسنتحدث فيما يلي عن الـ test- case الجيد وما هو ؟

وضوحاً مما سبق فإن الـ test_case الجيد هو الذي يستطيع كشف الأخطاء والعيوب داخل البرنامج أو داخل النظام.

# What Is Software Testing

تُعد مرحلة اختبار البرمجيات في دورة حياة تطوير البرمجيات (SDLC) مرحلةً حاسمةً لضمان الجودة، تحدث بعد مرحلة التطوير وقبل النشر، وتُعدّ بمثابة البوابة الرئيسية لجودة البرمجيات.
عادةً ما يهدف اختبار البرمجيات إلى تشغيل برنامج أو تطبيق بهدف اكتشاف ثغرات برمجية (أخطاء أو عيوب أخرى).
ويعني اختبار البرمجيات تشغيلها في ظروف مُتحكم بها، لـ

- Verify that it behaves “as specified” ,
- To detect Defects / Errors ,
- To validate that what has been specified is what the user actually wanted.

**إن مهمة الاختبار هي عملية تكرارية، حيث عندما يتم إصلاح خطأ واحد، فإنه يمكن أن يؤدي إلى تسليط الضوء على أخطاء أخرى أعمق، أو حتى إنشاء أخطاء جديدة.**

# Why Do we need testing

**Software is every ware controlling our as :**

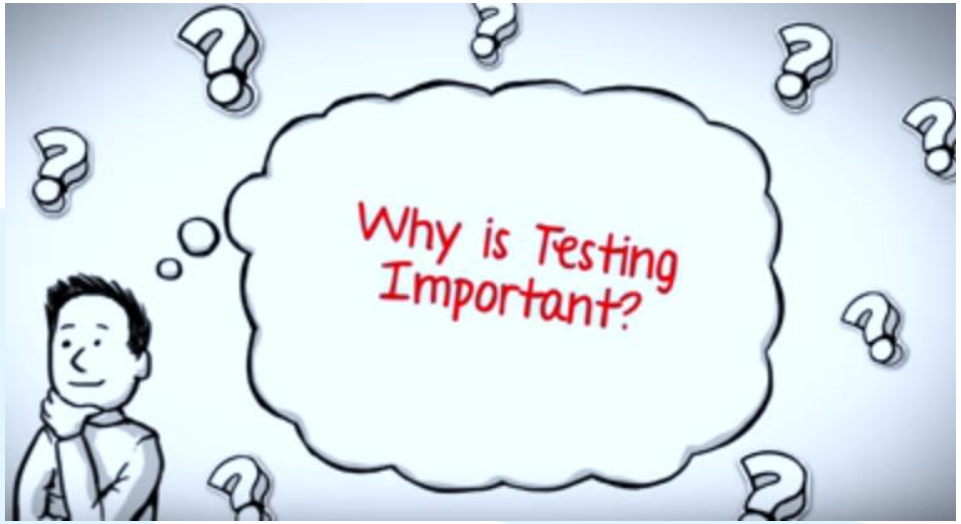
**Work :** PCs , OSs ,DBs , Servers , …

**Social Life :** FB , Twitter , Mobiles , **…**

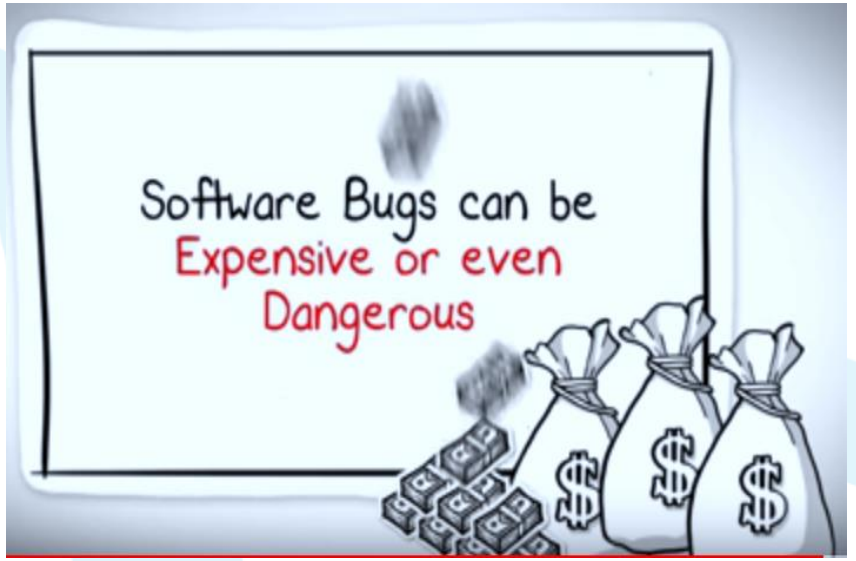
**Entertaining :** Games , Play Pad , …

**Medical :** Medical tools , medical tracking systems , …

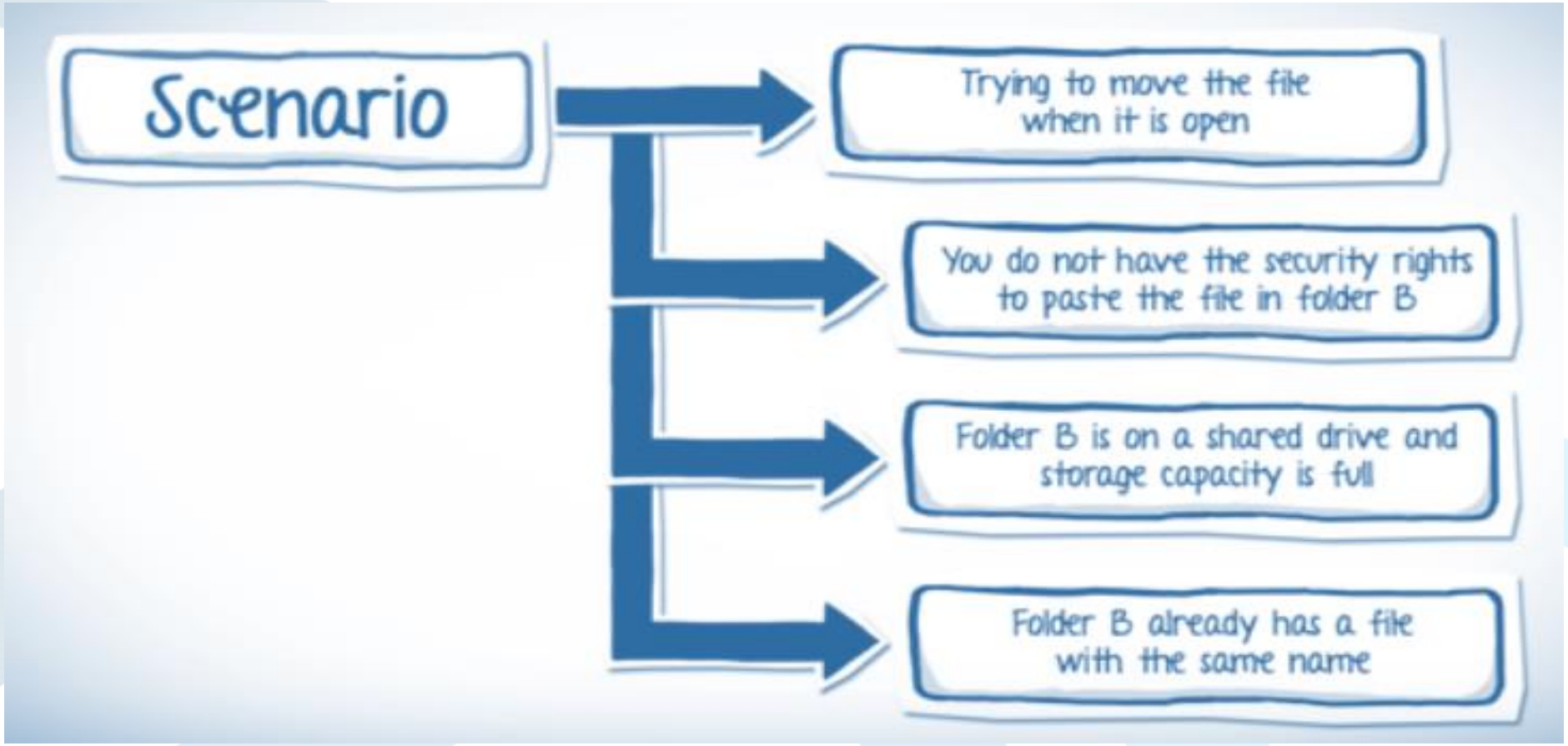
**Other :** Cars , Metro , Plains , trains , …




يمكن أن يؤدي البرنامج الذي لا يعمل بشكل صحيح إلى العديد من المشاكل، بما في ذلك خسارة المال أو الوقت أو سمعة العمل، وقد يؤدي حتى إلى الإصابة أو الوفاة.

# Example : Move file from folder A to folder B




HAVE ANY
QUESTIONS AND DOUBTS