



СОФТУЕРНИ АРХИТЕКТУРИ

**АТРИБИТИ ЗА
ОЦЕНКА НА
КАЧЕСТВОТО**

ВЪВЕДЕНИЕ

- Атрибутите за **качество** на софтуера са **показатели**, които описват предвиденото поведение на системата в средата, за която е предназначена.
- Атрибутите за качество осигуряват средства за измерване на възможностите на даден продукт.
- *Софтуерната архитектура влияе дълбоко върху повечето качества по един или друг начин, а атрибутите за качество на софтуера влияят на архитектурата.*

<http://www.softwarearchitectures.com/>

ФУНКЦИОНАЛНИ ИЗИСКВАНИЯ

- В софтуерното инженерство **функционалното изискване определя функция на дадена система или нейн компонент**. Под функция на система се разбира точна спецификация на нейното поведение по време на работа ѝ.
- Функционалните изисквания могат да включват **специфични изчисления, технически детайли за манипулиране и обработка на данни, както и други специфики, които определят какво трябва да изпълнява системата**.
- Изискванията към поведението на системата описват всички сценарии, заложи в нейните функционални изисквания. Те се описват чрез use case диаграми.
- Функционалните изисквания се допълват от **нефункционални изисквания (известни също като "изисквания за качество")**, които налагат ограничения върху проектирането и разработката на системата (като изисквания за производителност, сигурност или надеждност).

ФУНКЦИОНАЛНИ ИЗИСКВАНИЯ

- Обикновено функционалните изисквания се представят в следния вид - **„системата трябва да изпълнява изискването X или изискването Y“**.
- *Планът за изпълнение на функционалните изисквания е подробно описан в проектната документация (дизайна) на системата.*
- Определянето и промяната на функционалните изисквания на системата се описват чрез следния процес:

заявка на потребител/заинтересована страна за нова функционалност → анализ → създаване на use case сценарий → реализиране на изискването

- Заинтересованите страни правят заявка за нова функционалност; системните инженери анализират и обсъждат аспектите за нейната реализация; създават use-case диаграми, диаграми за взаимодействието между обектите, участващи в новата функционалност; ако се одобри, тя се документира и се подава за реализация към разработчиците

НЕФУНКЦИОНАЛНИ ИЗИСКВАНИЯ

- Нефункционалните изисквания определят критерии, които се използват за оценка на цялостната работа на дадена система, а не за конкретна нейна функция. Те са в контраст с функционалните изисквания, които определят конкретно поведение или функция.

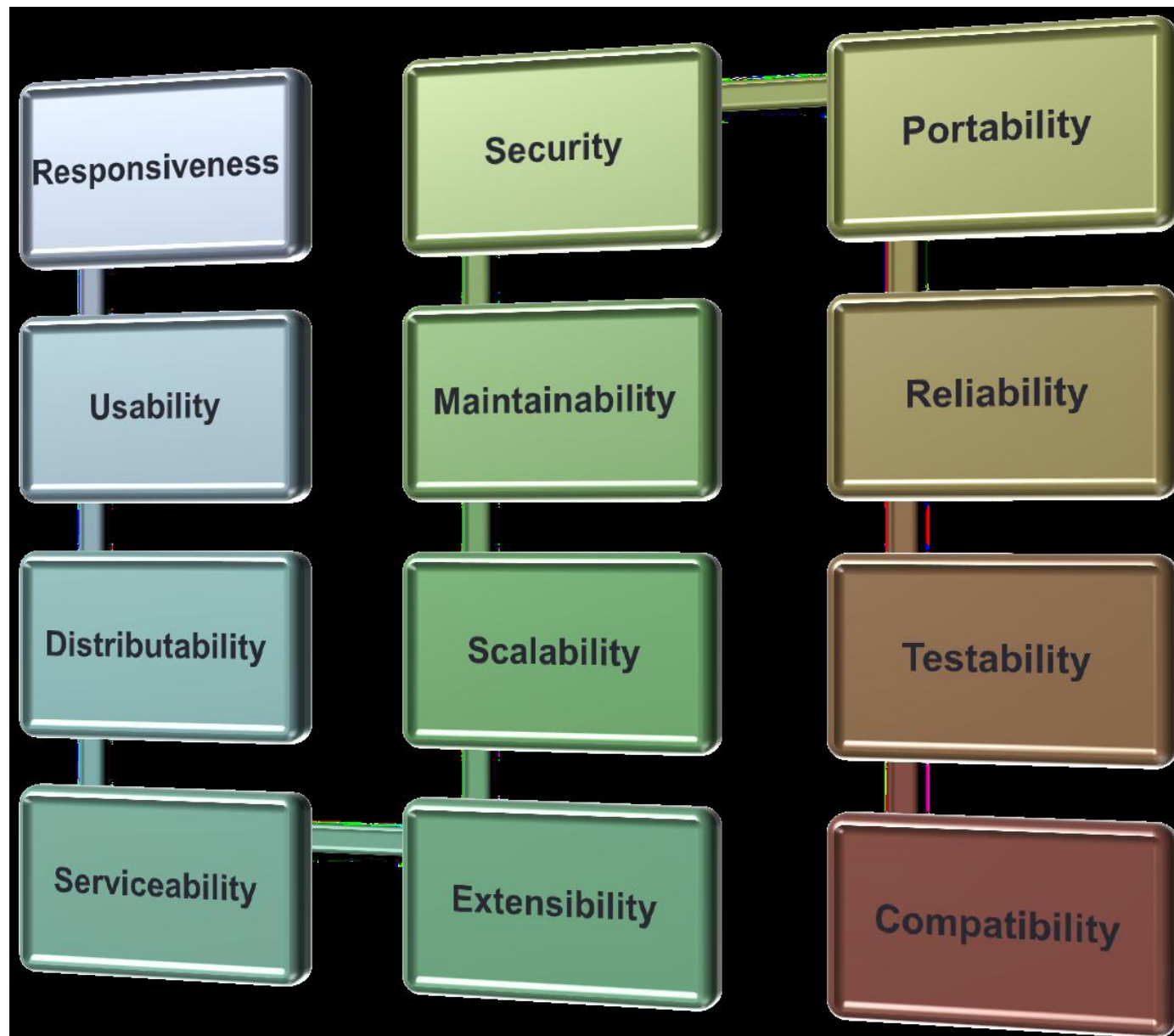
Планът за изпълнение на нефункционални изисквания е подробно описан в архитектурната документация на системата. Това е така, тъй като обикновено нефункционалните изисквания са архитектурно значими изисквания.

- Нефункционалните изисквания се дефинират по следни начин: „системата трябва да бъде <изискване> (сигурна, бърза, лесна за поддръжка и т.н.)“.
- Често това дали поставените нефункционални изисквания са реализирани, определя дали разработваната система е успешен или не толкова успешен проект.

АТРИБУТИ ЗА КАЧЕСТВО

- Как да оценим дали даден софтурен продукт е качествен или не? От какво зависи качеството на даден продукт?
 - Дали е с достатъчно добро бързодействие?
 - Дали е лесен за използване от потребителите?
 - Дали съхранява и обменя данни в защитен вид?
 - Дали лесно могат да се правят промени по неговата функционалност?
 - Какви са усилията по неговата поддръжка?
- Атрибутите за качество вземат отношение към нефункционалните изисквания на софтуера.

АТРИБУТИ ЗА ОЦЕНКА НА КАЧЕСТВОТО

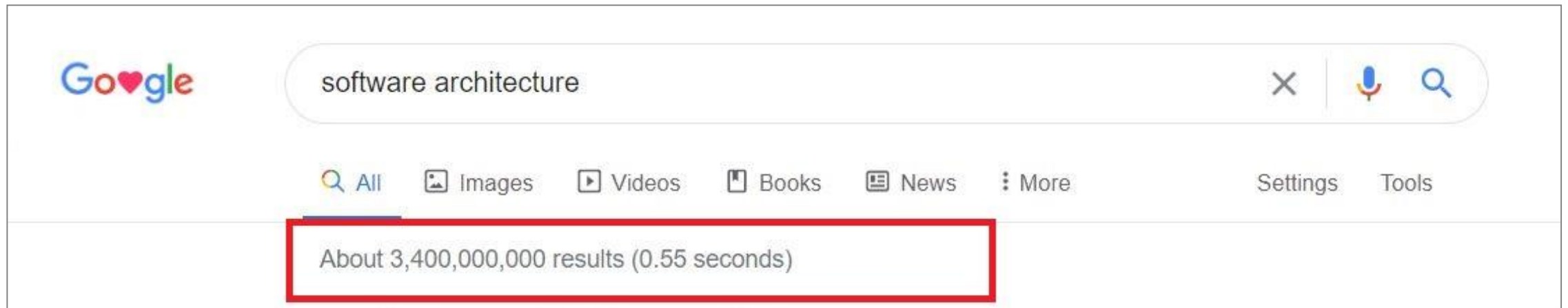


ВРЕМЕ ЗА РЕАКЦИЯ НА СИСТЕМАТА

RESPONSIVENESS

Този атрибут се дефинира като бързина на системата да реагира на потребителската заявка.

- Дългите закъснения могат да бъдат основна причина за неудовлетвореност на потребителя. Той може да сметне, че системата е повредена или че поставената от него команда е игнорирана. *Този атрибут не се припокрива с ефективността.*



ИЗПОЛЗВАЕМОСТ *USABILITY*

Използваемостта е способността на клиента да изпълни работата си със системата по ефикасен и приятен начин.

Работата със системата трябва да се основава на умения, които потребителят вече има, и да не изисква нови или уникални знания, за да използва системата.

Всяка нова функция, с която се сблъсква потребителят, трябва да следва подобен модел, така че след като потребителят е научил една функция, другите да могат да се научат интуитивно (консинстенност).

КОМПОЗИЦИЯ НА ИЗПОЛЗВАЕМОСТТА

Способност за учене (Learnability)

- Колко лесно е за потребителите на една система да изпълняват основни задачи (да я използват) при първата си среща със системата.

Ефективност (Efficiency)

- След като потребителите се запознаят основите на потребителския интерфейс, колко бързо могат да изпълняват задачи.

Способност за запоняне (Memorability)

- Когато потребителите се върнат към системата след период, в който не са я използвали, колко лесно могат да възстановят умението си.

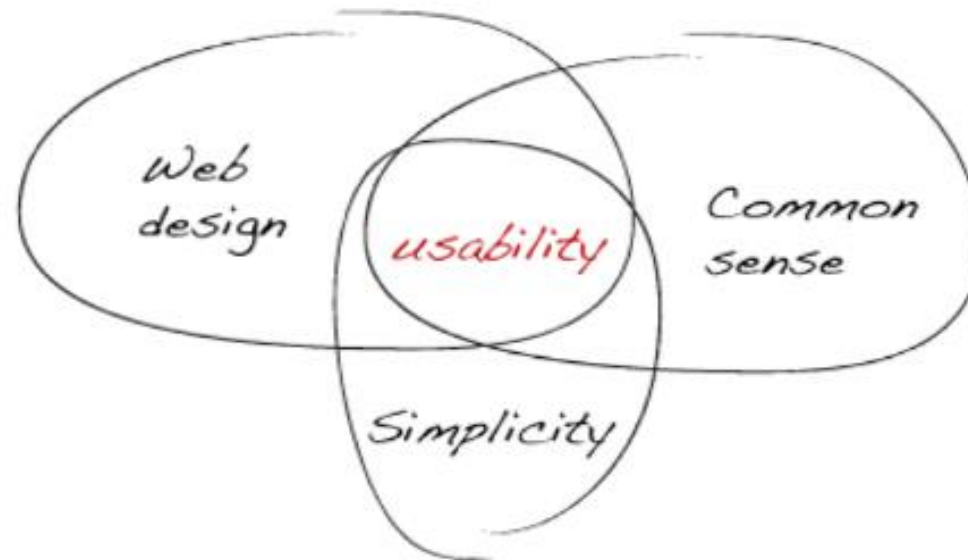
Допускане на грешки (Errors)

- Колко грешки правят потребителите, колко тежки са тези грешки и колко лесно могат да се възстановят от направените грешките.

Удовлетвореност (Satisfaction)

- Колко е приятно да работите със системата.

ИЗПОЛЗВАЕМОСТ *USABILITY*



ПРАВИЛО 1: НЕ МЕ КАРАЙ ДА МИСЛЯ?

<https://www.amazon.com/Dont-Make-Me-Think-Usability/dp/0321344758>

ПРАВИЛО 2: ОПОЗНАЙ СВОИТЕ ПОТРЕБИТЕЛИ?

ПРАВИЛО 3: БЪДИ ПОСТОЯНЕН!

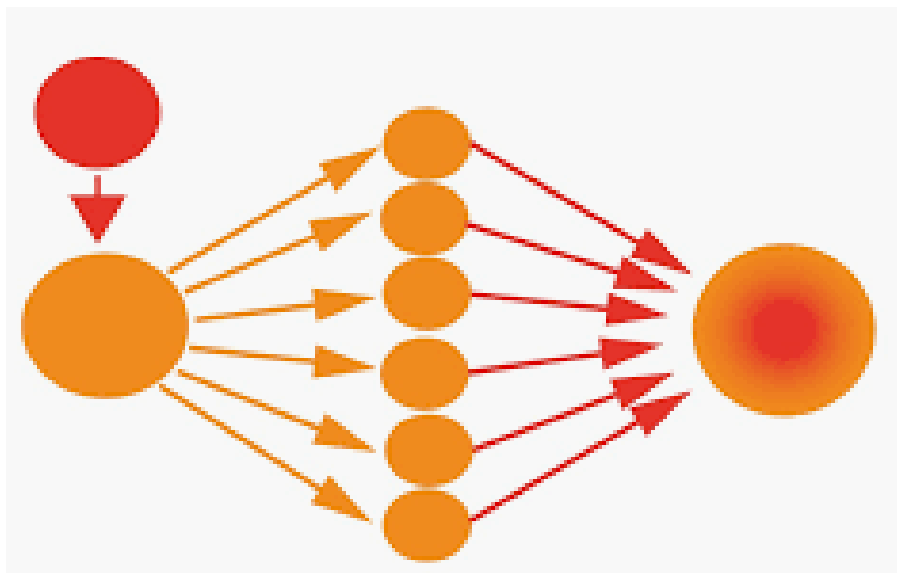
ДИСКУСИЯ ...

<https://www.softwaretestinghelp.com/best-usability-testing-tools/>

РАЗПРЕДЕЛЕННОСТ *DISTRIBUTABILITY*

Разпределеността се дефинира като способността да се управлява група от системи като едно цяло и да се извършва управление от всяко място в мрежата.

Трябва да се определи дали даден компонент ще се изпълнява в основния процес или в самостоятелен процес същата машина или на отдалечена машина (сървър).



ВЪЗМОЖНОСТ ЗА СЕРВИЗНО ОБСЛУЖВАНЕ *SERVICEABILITY*

Отнася се до способността за наблюдение на системите по време на тяхната работа, идентифициране на проблеми, извършване на анализ на основни причини за тези проблеми.

Основната цел е откриване и решаване на проблем, т.е. възстановяване на системата във функциониращо състояние.



ВЪЗМОЖНОСТ ЗА СЕРВИЗНО ОБСЛУЖВАНЕ *SERVICEABILITY*

Този атрибут се отнася до *способността за техническа поддръжка, идентифициране и отстраняване на грешки без системите да се извеждат от работоспособност.*

Проблеми могат да бъдат докладвани от клиенти или представители на техническо обслужване.

Всяка система трябва да притежава модул, поддържащ журнал за грешки (Error log). При необходимост този модул се конфигурира да генерира съобщения за грешки с различна степен на детайлност. Модулът може автоматично да известява отговорните за поддръжката на система лица за възникването на проблем чрез e-mail или друг комуникационен канал.

ВЪЗМОЖНОСТ ЗА РАЗШИРЯВАНЕ

EXTENSIBILITY

Разширяемостта е принцип в софтуерното инженерство и проектиране на системи, който осигурява бъдещо развитие или растеж.

Разширяемостта е мярка за способността за разширяване на системата и нивото на усилията, необходими за прилагане на разширението.

Разширенията могат да бъдат чрез добавяне на нова функционалност или чрез промяна на съществуваща такава.

Принципът предвижда подобрения, без да се нарушават съществуващите системни функции.

МЕХАНИЗМИ ЗА РАЗШИРЯВАНЕ

White-Box – извършва се при условие, че е наличен source (изходния) код на системата с или без неговата модификация.

- **Open-Box** – подход с модификация на изходния код. Най-гъвкав подход.
- **Glass-Box** – подход без директна подмяна на оригиналния изходен код,

Black-Box – разширение без достъп до изходния код на системата, използва се документация на налични програмни интерфейси и добавяне на нови модули

Gray-Box – компромисен вариант, който не разчита напълно на достъп до изходния код. Използва част от кода, описващ най-вече програмните интерфейси

МАЩАБИРУЕМОСТ *SCALABILITY*

Мащабируемостта е способността на системата да се справи с увеличаването на натоварването без да намалява производителността си.

Необходимост от мащабируемост може да възникне при:

- Увеличаване на броя потребители на системата
- Увеличаване на обема на обработваните данни



ВИДОВЕ МАЩАБИРУЕМОСТ

Хоризонтална (Scale Out) – представлява добавяне/внедряване на допълнителни модули от същия тип, изпълнявани най-често на отделни сървъри (Load Balancing).

Вертикална (Scale up) – добавяне на ресурси към вече съществуващ изчислителен възел (сървър) – добавяне на процесор, памет, пропускателна способност на мрежата.

Мащабируемост на база данни

ВЪЗМОЖНОСТ ЗА ПОДДРЪЖКА *MAINTAINABILITY*

Този атрибут е свързан с лекотата, с която даден продукт може да се поддържа, с възможност за:

- коригиране на дефекти или тяхната причина;
- поправяне или заменяне на дефектни или износени компоненти, без да се налага да се подменят работещи части;
- предотвратяване на неочаквани условия на работа;
- увеличаване на полезния живот на продукта;
- максимална ефективност, надеждност и безопасност;
- отговаряне на новите изисквания;
- улесняване на бъдещата поддръжка или справяне с променена среда.

ВЪЗМОЖНОСТ ЗА ПОДДРЪЖКА *MAINTAINABILITY*

КАЧЕСТВО НА СОФТУЕРЕН
ПРОДУКТ
ISO 25010

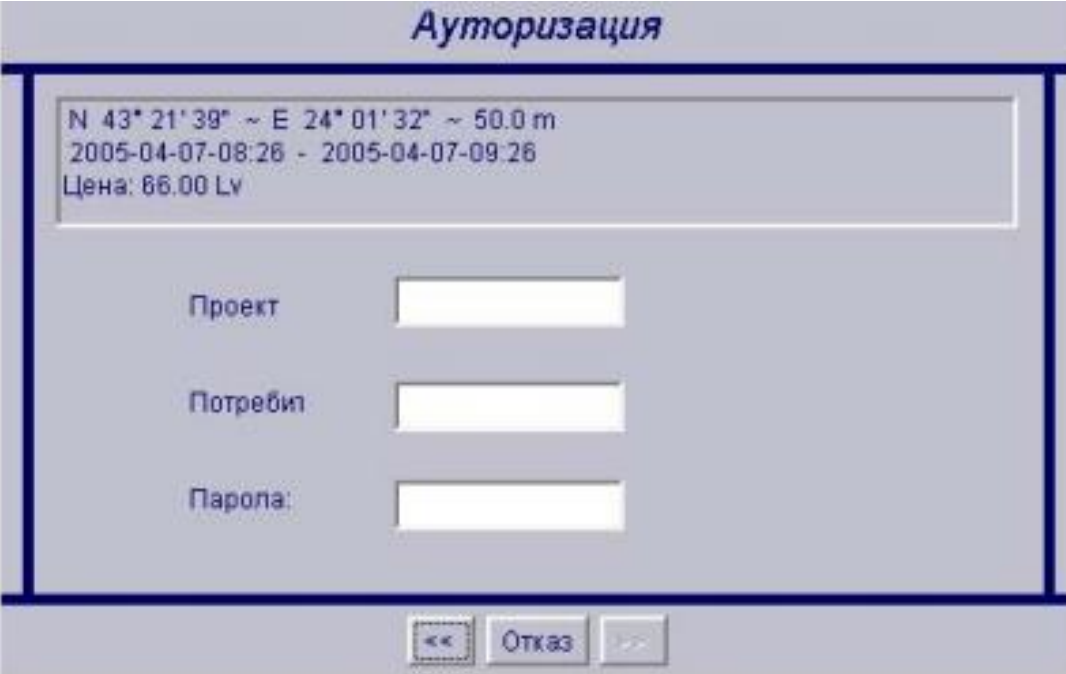


СИГУРНОСТ *SECURITY*

Сигурността е атрибут, свързан със степента на защита на данните от неправилен достъп до тях, осигуряване на идентификация на потребители

Сигурността се свързва с:

- *ИДЕНТИФИКАЦИЯ (Identification)*
- *АВТЕНТИКАЦИЯ (Authentication)*
- *ОТОРИЗАЦИЯ (Authorization)*
- *ПРОВЕРКА (Audit)*
- *КРИПТИРАНЕ (Encryption)*



Ауторизация

N 43° 21' 39" ~ E 24° 01' 32" ~ 50.0 m
2005-04-07-08:26 - 2005-04-07-09:26
Цена: 66.00 LV

Проект

Потребител

Парола:

«« Отказ »»

СИГУРНОСТ *SECURITY*

Идентификация – процес, при който потребителят заявява своята личност – чрез потребителско име или идентификационен код

Автентикация – процес, чрез който потребителят, доказва своята самоличност – чрез въвеждане на порала, пин код, пръстов отпечатък

Оторизация – процес на предоставяне на права за извършване на дейности след като потребителите са автентикирани

ПРЕНОСИМОСТ *PORTABILITY*

Преносимостта на високо ниво е използваемостта на един и същ софтуер в различни среди.

Предпоставка за преносимост е обобщената абстракция между логиката на приложението и системните интерфейси.

Когато се произвежда софтуер със същата функционалност за няколко компютърни платформи, преносимостта е основният проблем за намаляване на разходите за разработка.

ПРЕНОСИМОСТ ***PORTABILITY***

ПРЕНОСИМОСТТА НА СОФТУЕРА ВКЛЮЧВА:

Прехвърляне на инсталирани програмни файлове на друг компютър със същата архитектура или ОС.

Преинсталиране на програма от файлове за разпространение на друг компютър с подобна архитектура или ОС.

Изграждане на изпълними програми за различни платформи от изходния код; това е, което обикновено се разбира под „преносимост“.

ПРЕНОСИМОСТ *PORTABILITY*



ПОДХАРАКТЕРИСТИКИТЕ ЗА ПРЕНОСИМОСТ СА:

Приспособимост (*Adaptability*)

Възможност за инсталация (*Installability*)

Съвместно съществуване (*Co-Existence*)

Заменяемост (*Replaceability*)

Съответствие за преносимост (*Portability Compliance*)

Този атрибут трябва да се има предвид от архитектите. Самата система и използваните технологии в продуктите често се променят в бъдеще. Трябва да се изграждат системи по начин, който да може да позволява промени на подсистемите без много работа.

НАДЕЖДНОСТ *RELIABILITY*

Надеждността е вероятността за безпроблемна работа на софтуера за определен период от време в определена среда с определено натоварване.

Тя е важен фактор, влияещ върху удовлетвореността на потребители при работата със системата.

Високата сложност на софтуера е основната причина за проблемите с надеждността на софтуера.

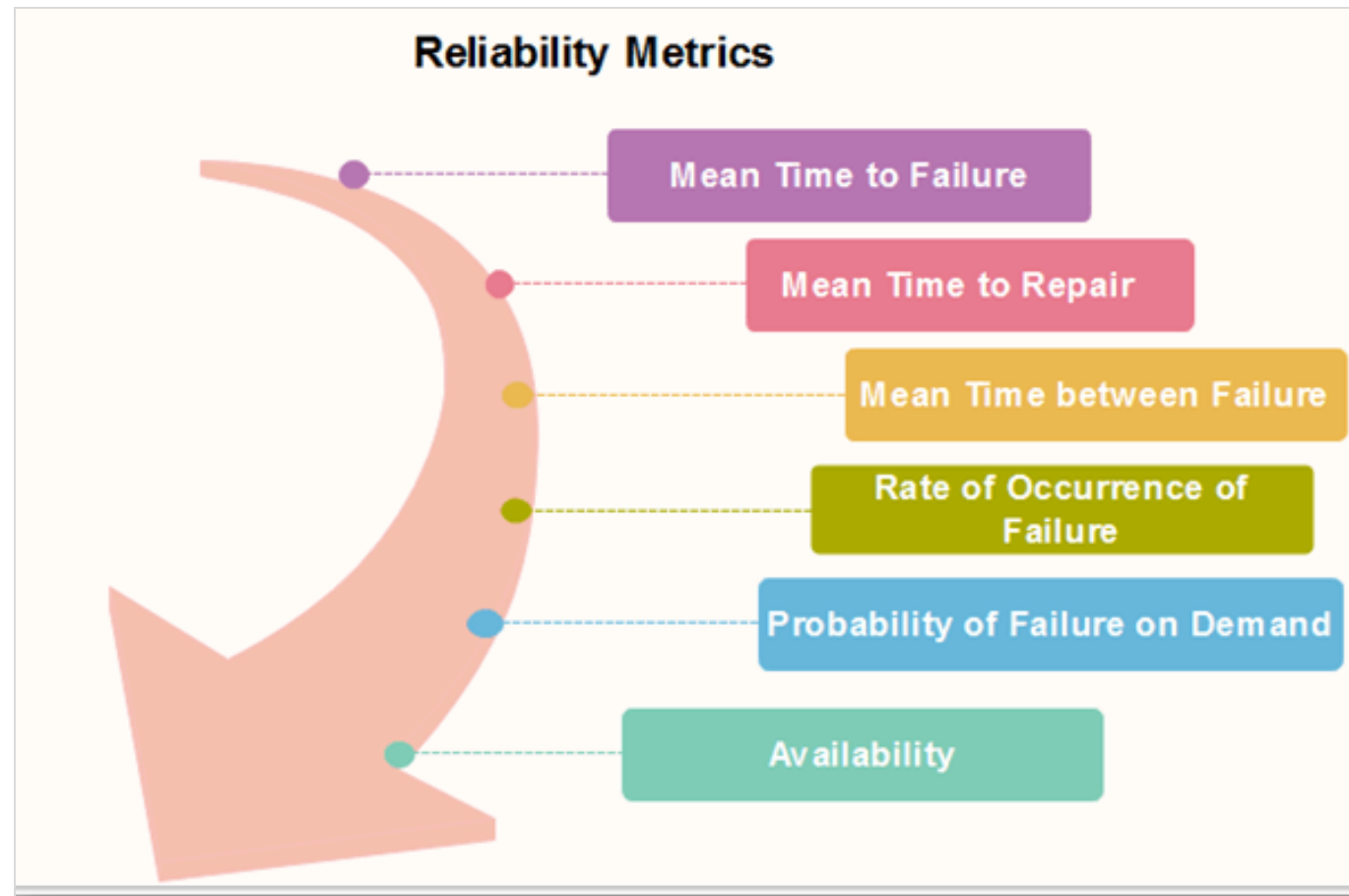
Надеждността на системата може да се провери чрез различни видове тестове – load тестове (за натоварване), тестове за отказоустойчивост и др.

Пример: производство на процесори

НАДЕЖДНОСТ *RELIABILITY*

Показателите за надеждност се използват за количествено изразена надеждност на софтуерния продукт.

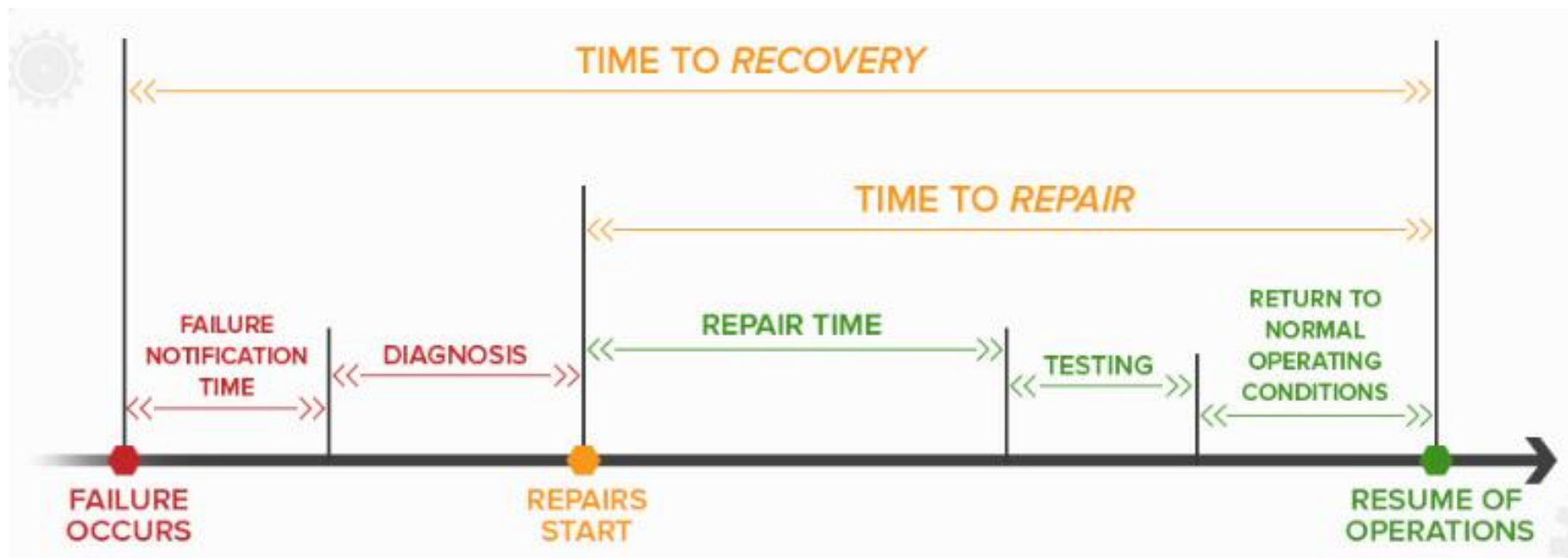
Вариантът на използването на определен показател, *зависи от типа система, към която се прилага, и от изискванията на областта на приложението.*



НАДЕЖДНОСТ *RELIABILITY*

Показателите за надеждност, които могат да бъдат използвани за количествена оценка на надеждността на софтуерния продукт, са следните:

Средно време до отказ/грешка (Mean Time to Failure - MTTF) - описва се като интервал от време между двата последователни отказа.



НАДЕЖДНОСТ *RELIABILITY*

Средно време за ремонт (Mean Time to Repair - MTTR) – при възникване на грешка, е необходимо известно време за нейното отстраняване.

MTTR измерва средното време, необходимо за проследяване на грешките, причиняващи повредата и за тяхното отстраняване.

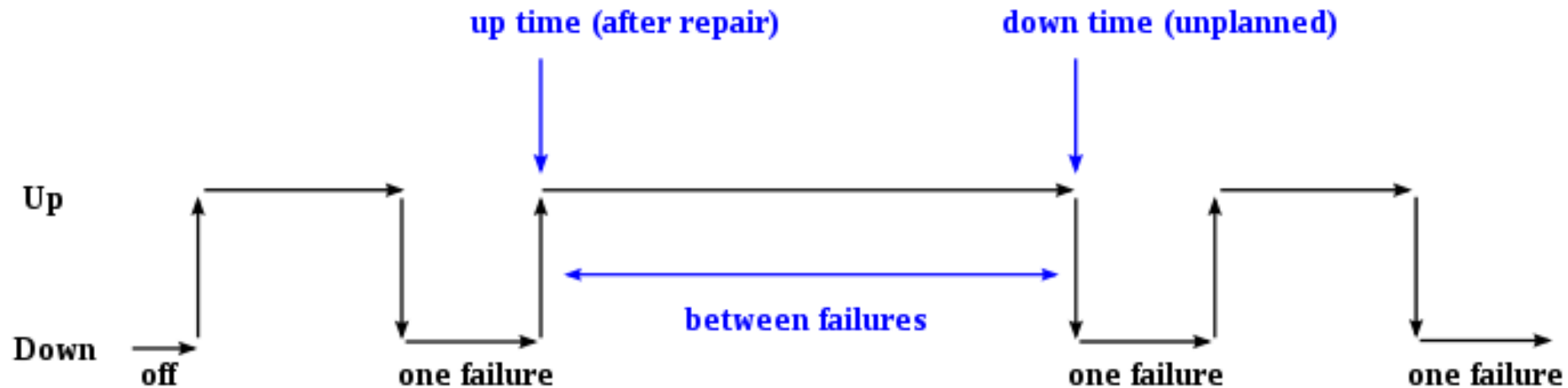
Изразено математически, това е общото време за коригираща поддръжка за повреди, разделено на общия брой коригиращи действия за поддръжка за повреди през даден период от време.

НАДЕЖДНОСТ *RELIABILITY*

Средно време между откази (Mean Time Between Failure - MTBR)

$$\text{MTBF} = \text{MTTF} + \text{MTTR}$$

MTBF може да се изчисли като средноаритметично (средно) време между отказите на системата.



$$\text{Time Between Failures} = \{ \text{down time} - \text{up time} \}$$

ВЪЗМОЖНОСТ ЗА ПРОВЕРКА *TESTABILITY*

Проверка на софтуера е степенята, в която софтуерният артефакт (т.е. софтуерна система, софтуерен модул, изисквания или документ за проектиране) поддържа тестване в даден тестов контекст.

Ако изпитателността на софтуерния артефакт е висока, тогава намирането на неизправности в системата (ако има такива) чрез тестване е по-лесно.

ВЪЗМОЖНОСТ ЗА ПРОВЕРКА *TESTABILITY*

Усилието и ефективността на софтуерните тестове зависи от множество фактори, включително:

Свойства на софтуерните изисквания

Свойства на самия софтуер (като размер, сложност и възможност за тестване)

Свойства на използваните методи за тестване

Свойства на процесите на разработване и тестване

Квалификация и мотивация на лицата, участващи в процеса на тестване



СЪВМЕСТИМОСТ **COMPATIBILITY**

СЪВМЕСТИМОСТ НАПРЕД (FORWARD COMPATIBILITY)

Това е съвместимост на дизайна, която позволява на системата да приема вход, предназначен за по-късна версия на самата нея.

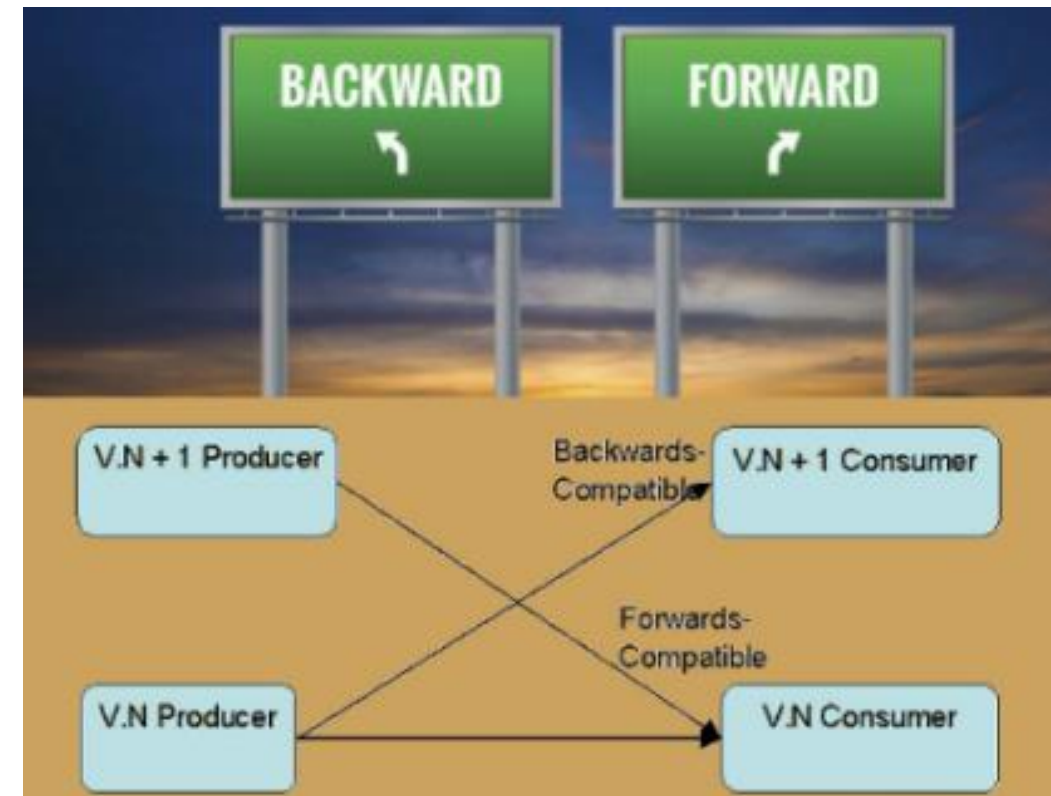
Концепцията може да се прилага за цели системи, електрически интерфейси, телекомуникационни сигнали, протоколи за комуникация на данни, файлови формати и езици за програмиране.

Дизайнът, който е съвместим с този атрибут, обикновено има пътна карта за съвместимост с бъдещите стандарти и продукти.

СЪВМЕСТИМОСТ COMPATIBILITY

ОБРАТНА СЪВМЕСТИМОСТ (BACKWARD COMPATIBILITY)

Това е свойство на система, продукт или технология, която позволява оперативна съвместимост с по-стара наследена система или с вход, предназначен за такава система, особено в телекомуникациите и изчислителната техника.



АТРИБУТИ ЗА КАЧЕСТВО ОТ БИЗНЕС ГЛЕДНА ТОЧКА

Време за пускане на продукт на пазара

Разходи и ползи

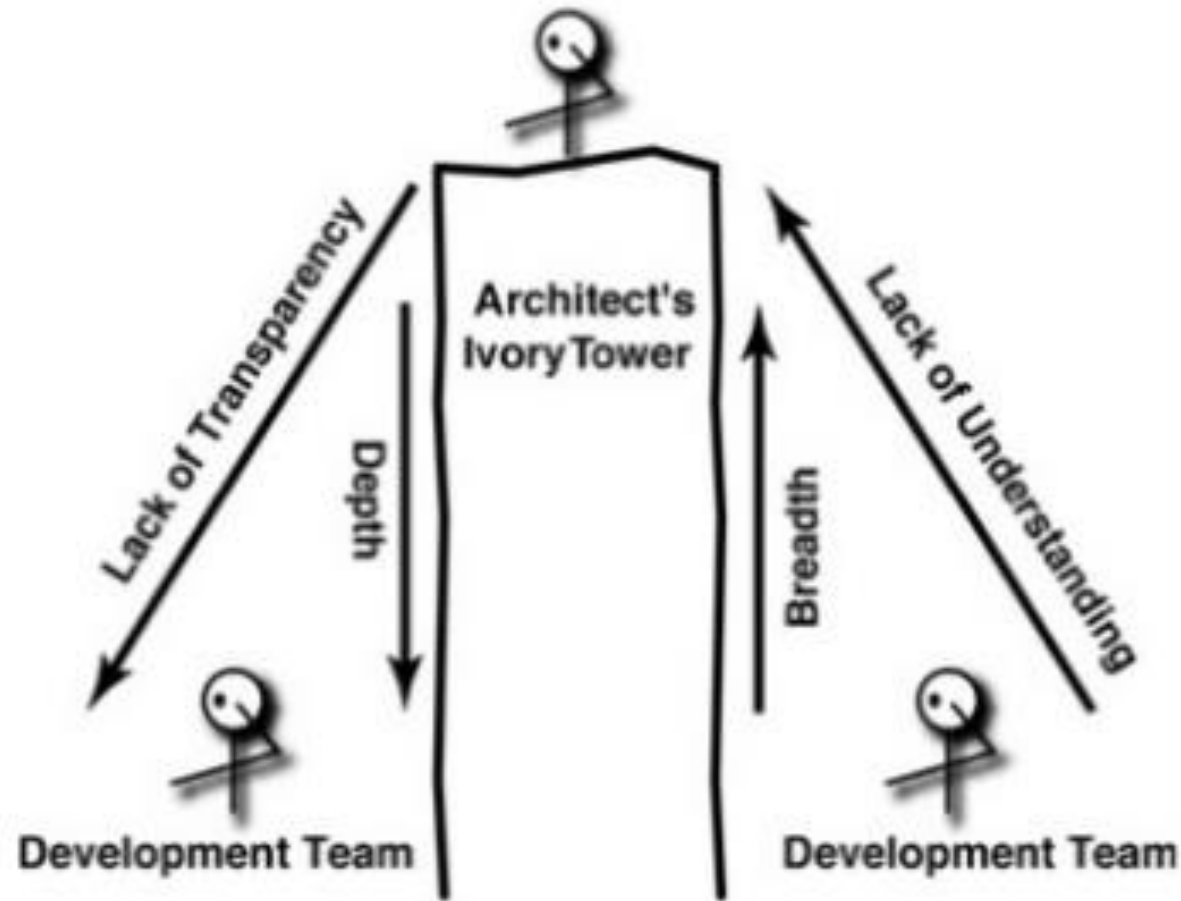
Продължителност на проекта

Целеви пазар

График за заместване на стар софтуер

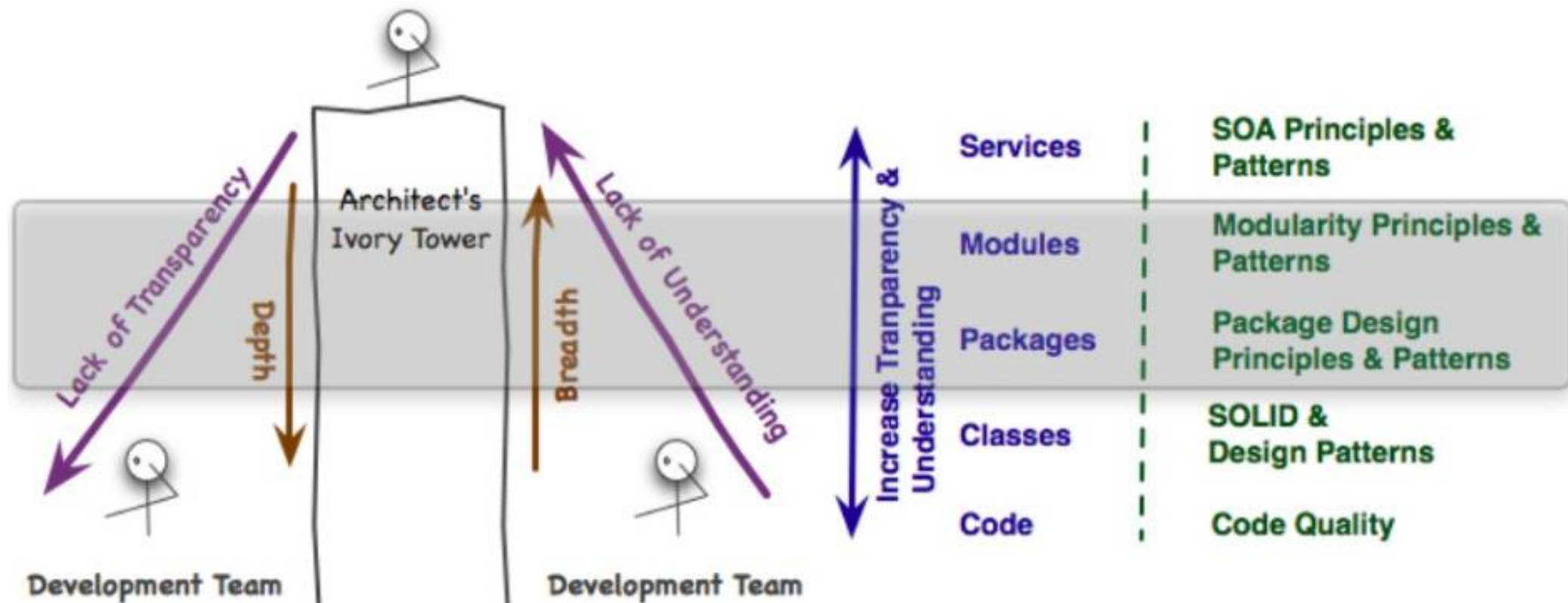
Интеграция с наследни системи

ЛОША КОНЦЕПЦИЯ НА СОФТУЕРНА АРХИТЕКТУРА



ПО-ДОБРА КОНЦЕПЦИЯ НА СОФТУЕРНА АРХИТЕКТУРА

Architecture All the Way Down



БЛАГОДАРЯ ЗА ВНИМАНИЕТО!