Извличане на изискванията – прототипиране. Анализ на изискванията

Лекция 5

- **Техники за извличане на изискванията (продължение)**

  - User story
  - Прототипиране

- **Анализ на софтуерните изисквания**

- **SMARTT изисквания**

8. Потребителска история (User story)

- **Кратки, прости, общи описания на функция, разказана от конкретна гледна точка**.
- Гледната точка е на човек - потребител, клиент или друга заинтересована страна, който желае функцията.
- Една потребителска история трябва да уточнява крайната цел (ползата, стойността), която заинтересованата страна ще получи.
- Артефакт, който определя работата на разработването на софтуерната система.

Цели на потребителските истории

- Способства за деформализиране на описанието на изискванията.
- **Променя фокуса на работата от описание на изискванията към дискусия за тях**.
- Потребителската история запазва фокуса върху потребителя и напомня, че има действителна цел, която трябва да бъде постигната чрез прилагане на функционалност.

8. Прототип

Дефиниция: **Прототипът е (*начална) версия на система*, която може да се използва за експериментиране**.

- ✓ Хардуерни системи
- ✓ Софтуерни системи

Прототипите в подкрепа на дейности на инженеринга на изискванията

Прототипирането може да се прилага при два процеса на ИИ (Boehm, 1984 г.).

- **извличане на изискванията и**
- **валидиране на изискванията**

1. Прототипите са полезни за *идентификацията на изискванията*, тъй като потребителите могат да експериментират със системата и да посочат нейните силни и слаби страни. Те имат *нещо конкретно*, което да критикуват.

2. Чрез прототип при *валидиране на изискванията* лесно може да се коригира/оптимизира първоначалния вид на системата.

Цели и роля на прототипирането в различни етапи на софтуерната разработка

- **Изяснява и допълва (завършва) изискванията**
- **Може да прерасне в цялостен краен продукт** (в определени случаи) посредством последователност от работни цикли.
- **Изследва проектни решения и алтернативи** като начини на
  - взаимодействие с потребителя,
  - оптимизация на използването на софтуера
  - оценка на технически подходи.

Прототипите в дейностите на ИИ - особености

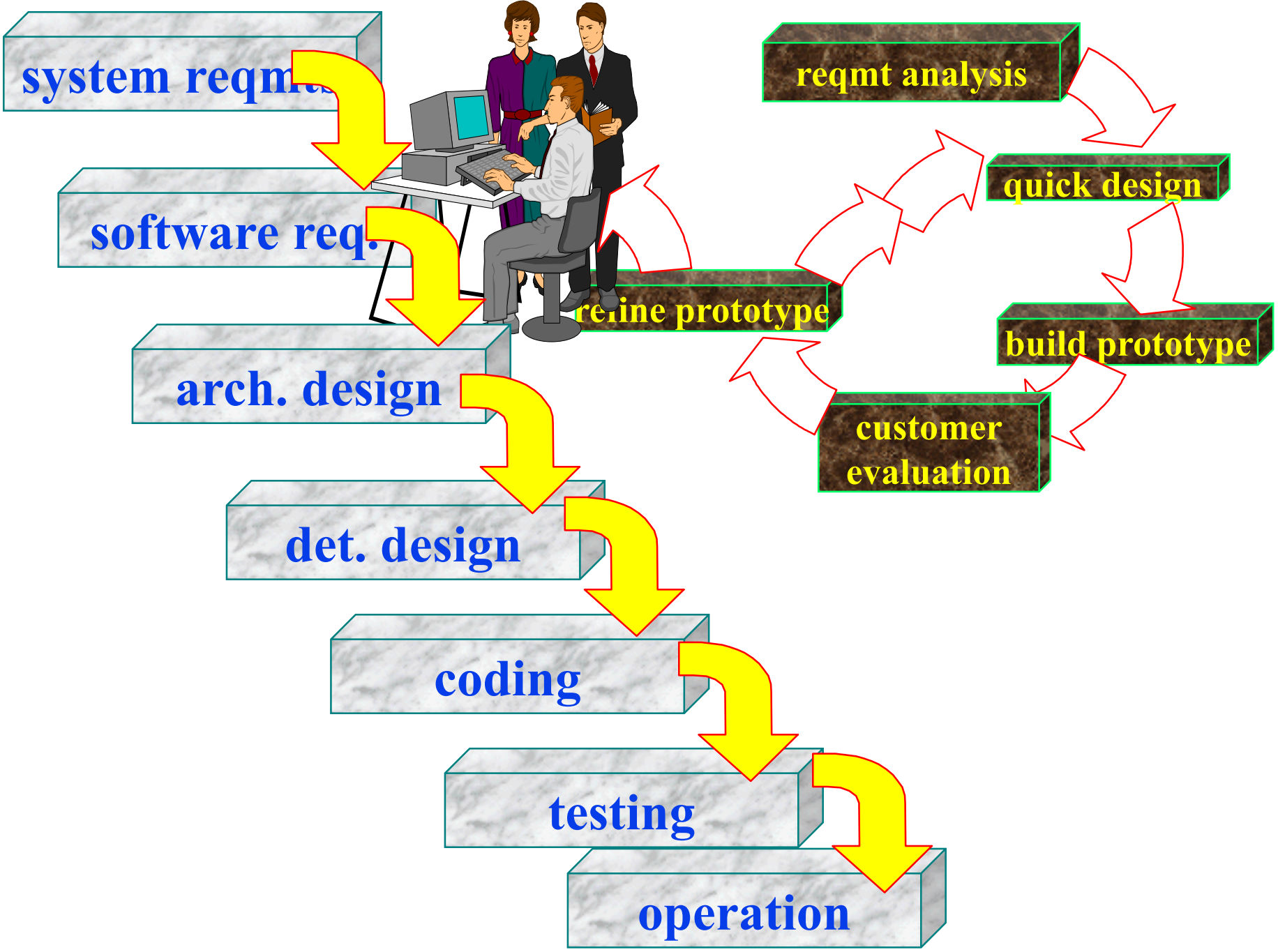
- ***Бързото разработване*** на прототипа е много важно, за може да бъде наличен *в ранния етап* на процеса на идентификация.

- *За целта*:

  - реализират се само *определени функционалности*

  - *не се реализират* някои *качествени изисквания* (RT, memory, error handling…)

  - *може да не се реализират някои основни процеси* на „класическата“ разработка (управление, тестване …);


system reqmt.
software req.
arch. design
det. design
coding
testing
operation
reqmt analysis
quick design
build prototype
customer evaluation
refine prototype

Видове прототипи – 1

- **Throw-away прототип**

  - подпомага специфицирането на изискванията

  - бърза разработка и кратко “съществуване”, което обуславя занижено качество на изпълнение

  - преизползване на отделни компоненти

  П*риложение*: хардуерни системи; за големи системи; Wizard of Oz, mock-ups;

- **Еволюционен прототип** – не се предоставя детайлна спецификация, но изисква прецизност в изработката.

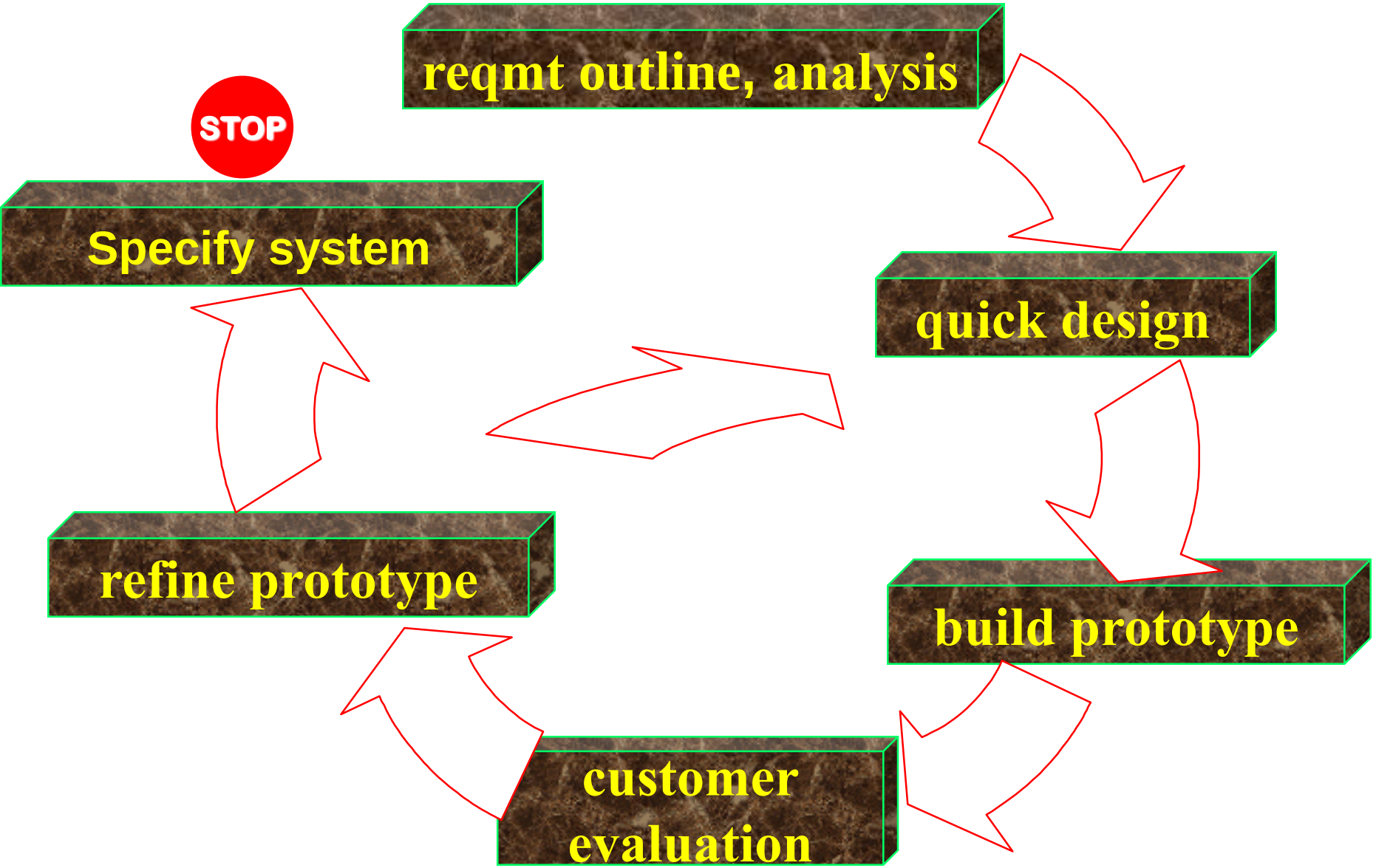
  - основна техника в съвременните софт. технологии- защо?

  - за малки и средни по големина системи, но не и за големи проекти (защо?)

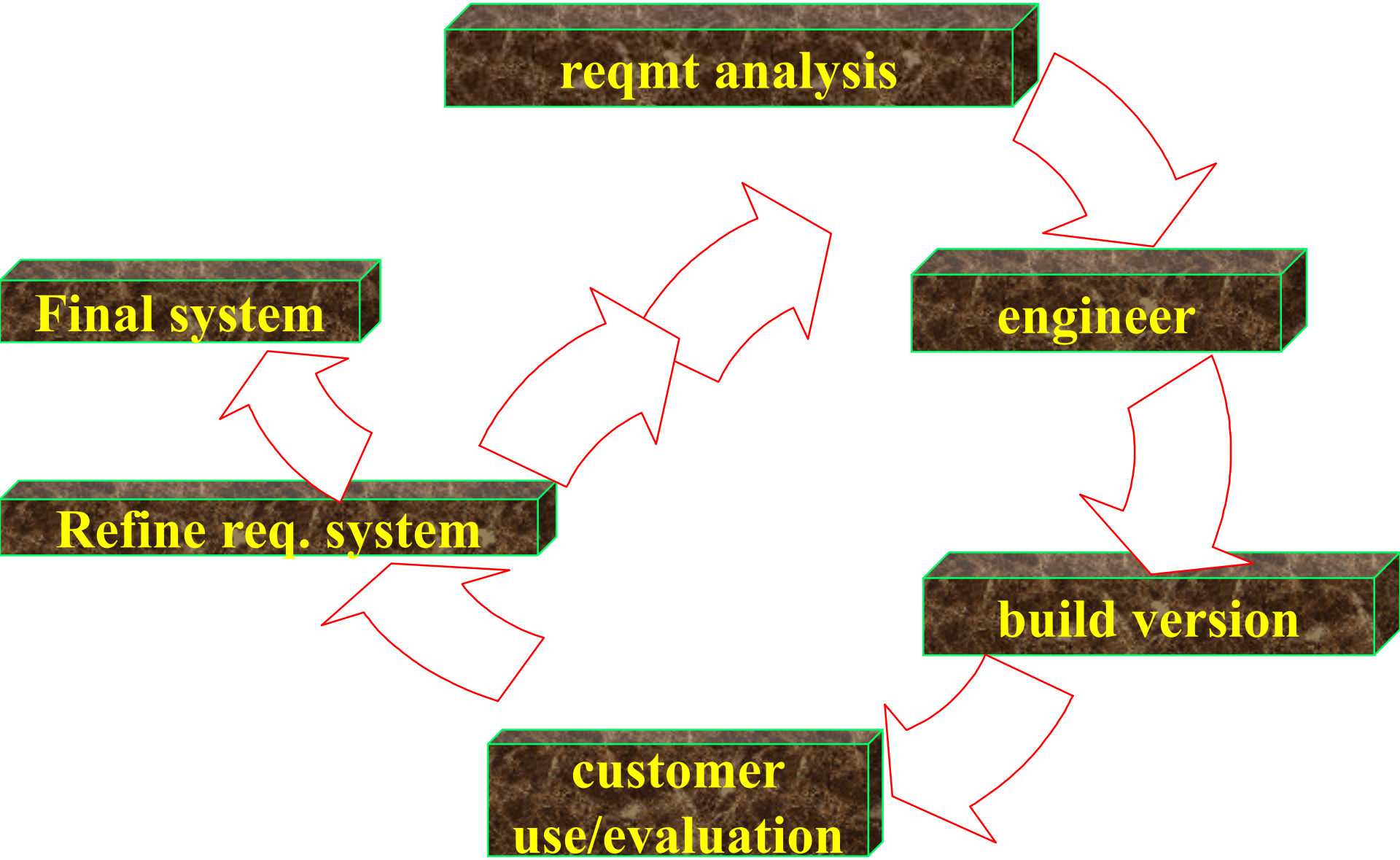
  Приложение - e-commerce приложения, web-site разработване

Boundaries …?

Модел на Throw-away прототипиране



Модел на еволюционното прототипиране



Видове прототипи и видове изисквания, които те представят - 1

- **Throw-away прототипиране**

  - идентифицира/реализира изисквания, които *създават най-много затруднения на клиентите и които са най-трудни за разбиране*.

  - обикновено се ползва *само* в помощ на *идентифициране* на изискванията.

  - бърза реализация.

Видове прототипиране и видове изисквания, които те представят 2

- **Еволюционно прототипиране**

  - целта му е *бързо* да предостави *работеща система* на клиента.

  - Затова изискванията, които трябва да се поддържат от началните версии са тези, които са *добре разбираеми* и които могат да предоставят полезна функционалност за крайните потребители.

Vague boundaries …

- **Хоризонтален (по поведение) прототип or mock-up**

  Не се “гмурка” във всички слоеве на архитектурата и не реализира реална работа.

  Първично описание на част от *интерфейса*; изследва *специфични* страни на бъдещата система.

  На фокус е *общата визия*; *широк* кръг изисквания и работни потоци

Пример: Western movie.

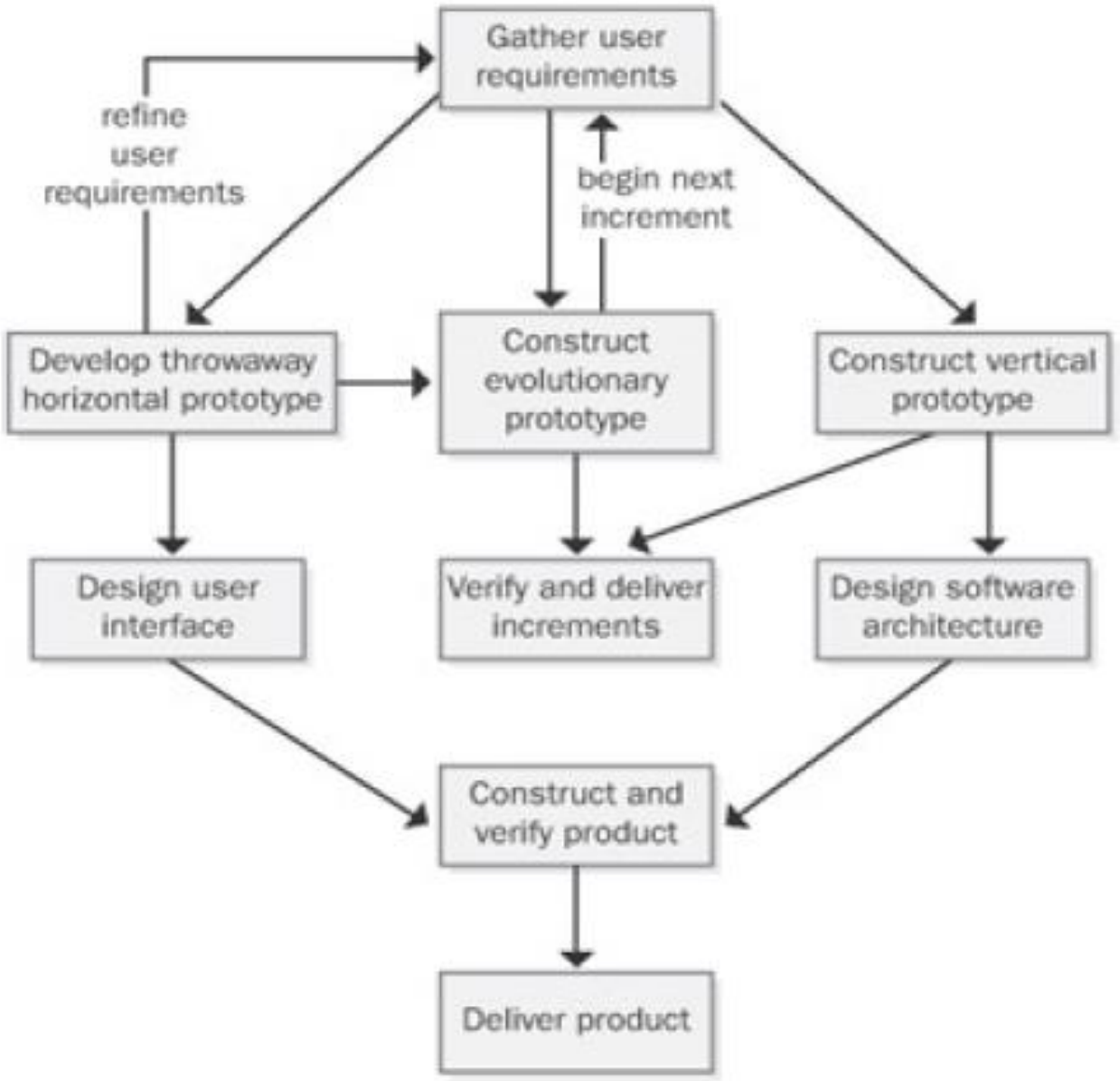
- **Вертикален (structural or proof of concept) прототип**

  Реализира отделна част от системата в цялостност на интерфейс и технически услуги.

  Прилага се в случай на неизяснени въпроси от *архитектурата*, *оптимизиране* на алгоритъм, *схемата на БД*, ограничения за време...

Пример (Karl E.Wiegers): „Вертикален прототип, който имплементира само част от потребителския интерфейс и съответната сървърна функционалност (на UNIX), ни позволи да оценим комуникационните компоненти, производителността и надеждността на предложената от нас клиент/сървър архитектура. Експериментът беше успешен, както и реализацията, базирана на тази архитектура.“

Възможни начини за включване на прототипирането в разработването на софтуера (Karl E.Wiegers, 2003)



Typical applications (1)

	Throwaway	Evolutionary
Horizontal	• Clarify and refine use cases and functional requirements. • Identify missing functionality. • Explore user interface approaches.	• Implement core use cases. • Implement additional use cases based on priority. • Implement and refine Web sites. • Adapt system to rapidly changing business needs.

Typical applications (2)

	Throwaway	Evolutionary
Vertical	• Demonstrate technical feasibility.	• Implement and grow core client/server functionality and communication layers. • Implement and optimize core algorithms. • Test and tune performance.

Други ползи от прототипирането

- Позволява *детайлно разглеждане* на изискванията като *открива* несъгласуваност и пропуски.
- *Оценява осъществимостта* на системата и използваемостта ѝ **преди да се правят големи разходи** за разработката.
- (единствения) **ефективен** начин за разработване на *потребителския интерфейс.*
- Но ! *допълнително* може да се използва за:
  - създаване на *тест* на системата
  - за съставянето на *документация*
  - за *обучение*

Подходи за *бързо* прототипиране

- **Прототипиране на хартия**

  създава се хартиен mock-up на системата и се използва за системни експерименти

  (Mok-up - to create user interfaces that show the end user what the software will look like without having to build the software or the underlying functionality. Software UI mockups can range from very simple hand drawn screen layouts, through realistic bitmaps, to semi functional user interfaces developed in a software development tool)

- **'*Wizard of Oz*' прототипиране**

  отговорите на системата за определени входни потребителски данни се симулират от човек

- **Автоматизирано прототипиране**

  използва се език от четвърто поколение или друга среда за *бързо* разработване, за да се създаде изпълним прототип

Кога ще използвате всеки един от тези подходи?

Разработване на изпълним прототип

- ✓ Езици от четвърто поколение, базирани на системи за бази от данни.
- ✓ Езици за визуално програмиране като Visual Basic или ObjectWorks, в които има *готови обекти*.
- ✓ Интернет базирани прототипи, основани на уеб браузъри и езици като Java – *готови интерфейси, сегменти (аплети*), които се стартират автоматично при зареждане в браузера.

Прототипирането на интерактивни системи е много по-лесно отколкото прототипирането за критични системи и системи в реално време (защо?).

Проблеми и разходи при прототипирането

- **Разходи за обучение** – разработването на прототипи може да изисква *специални инструменти*

- **Разходи за разработката** – зависят от вида на прототипа. Разходите в началото на софтуерния процес, но ги намалява в по-късните етапи (защо?).

- **Разширен график за разработка** –времето за прототипиране може да се възвърне в последствие, тъй като се избягва преработката на софтуер.

- **Незавършеност**: 1) Прототипът не е завършена система!

2) **Не е** възможно да се прототипират *критични* системни изисквания;

3) **Не е** възможно да се прототипират някои качествени изисквания;

Оценяване на прототипа

Специфични (**what**) въпроси:

- Дали прототипът реализира функционалността?
- Липсва ли някоя функционалност?
- Има ли някакви състояния на грешки, които прототипът не засяга?
- Има ли ненужни функции?
- Колко логична и завършена е навигацията?
- Бяха ли много сложни някои от задачите, реализирани с него?

Правилните хора оценяват прототипа в подходящи перспективи:

- Включат се ***опитни, но и неопитни*** представители на потребителската група.

Ограничения при разработването на прототип

Работещата версия има примамлив вид.

но !

- Не правете един throw-away прототип по-сложен отколкото е необходимо, за да бъдат постигнати прототипните цели.
- Не се поддавайте на натиска от страна на потребителите.

Същевременно

Не е необходимо да изхвърляте прототип, а да бъде запазен за повторно използване.

Той обаче няма да бъде включен в крайния продукт. Поради тази причина може да го наричате *no releasable* прототип.

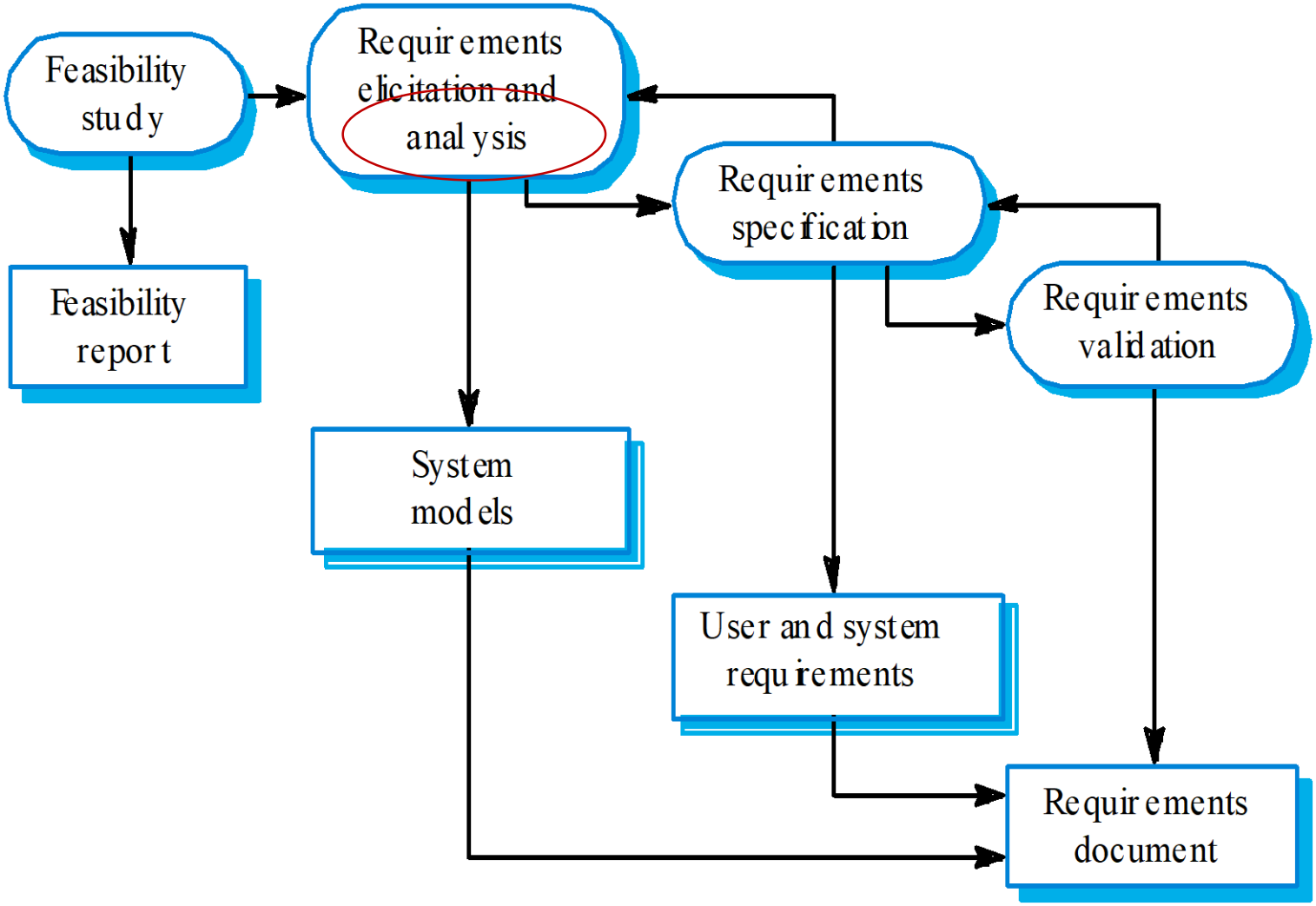
Коя е най-добрата техника за извличане на изискванията?

Анализ на изискванията

Лекция 5

- **Анализ на изискванията**
- **Процес на анализ на изискванията**
- **Техники за анализ на изискванията**
- **Договаряне на изискванията**

Процес на инженеринг на изискванията



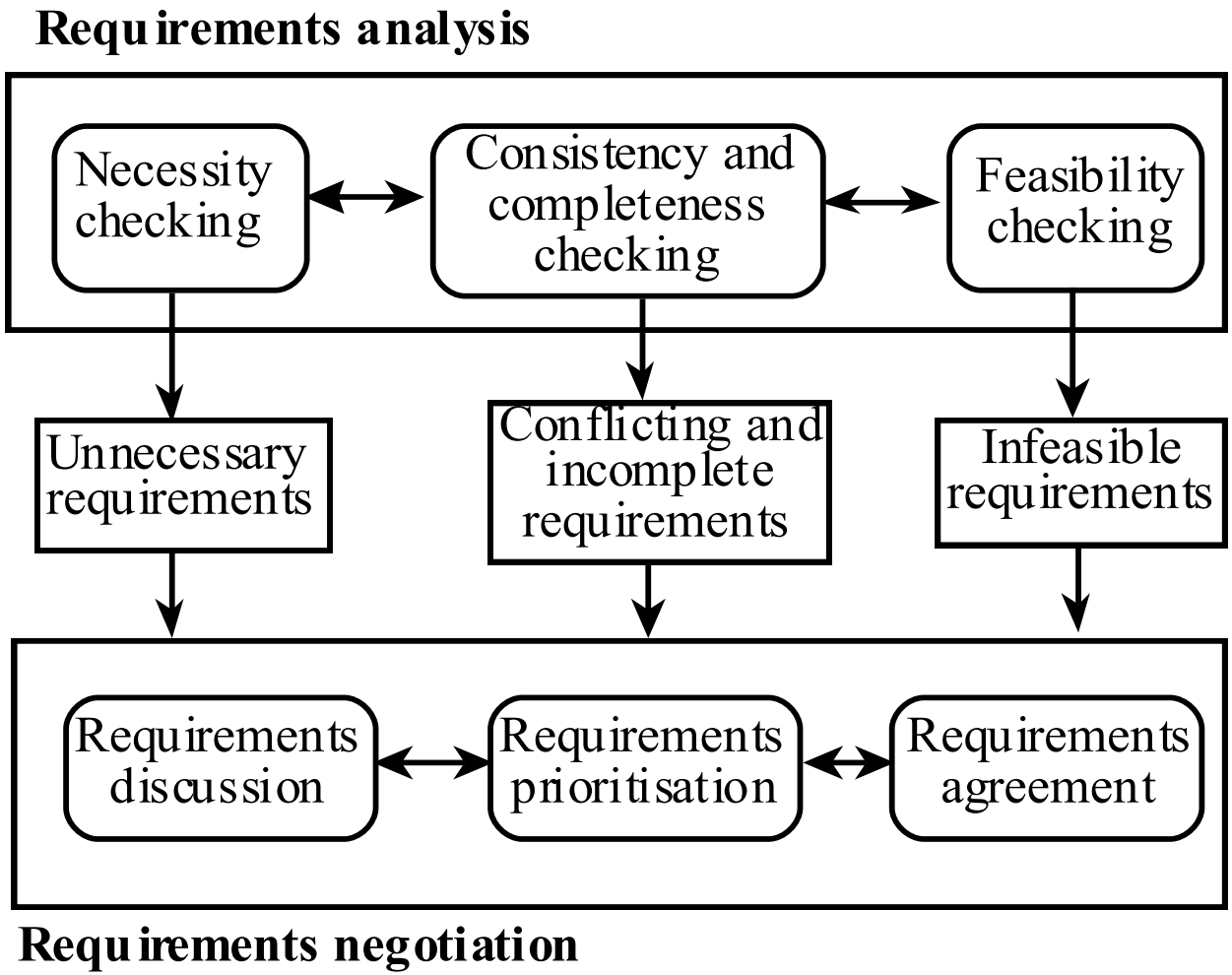
Анализ на изискванията

- Целта на анализа е да **открие проблеми** като *незавършеност и несъгласуваност* в идентифицираните изисквания, както и да обсъди **и разреши** със заинтересованите лица в процеса *на преговорите.*

- Анализ и извличане на изискванията – взаимна зависимост (*interleave)*

- Анализът е зависим от опита и експертизата на хората, които са включени в процеса на анализ.

Процеси на анализ и преговори на изискванията



Проверки на анализа

- Проверка за **необходимост**

  за да не допускаме изисквания, които *не допринасят за постигането на бизнес целите* на организацията или не спомагат за решаването на специфичния проблем, адресиран от системата.

- Проверка за **съгласуваност и завършеност**

  Съгласуваност означава, че изискванията не трябва да си *противоречат*; *завършеност означава*, че никои от необходимите услуги и ограничения не са пропуснати.

- Проверка за **осъществимост**

  Дали изискванията *са изпълними в контекста на бюджета* и *плана* за разработването на системата.

Анализ на изискванията

Каква е разликата между анализ на изискванията и валидиране на изискванията?

Анализ на изискванията - фактори

Функционалните изисквания се представят *на съответното ниво на детайлност*.

Пример:

1) Един уеб сайт, който се изгражда на стъпки (инкрементално) от малък, добре синхронизиран екип, може да бъде и с ограничена документация.

2) За една сложна вградена система, която ще бъде outsourced отдалечено, е необходима точна и детайлна SRS.

Техники за анализ на изискванията

1. Списък (Checklist) (1)

- Списък от ключови и достатъчно **общи** въпроси за оценка на всяко едно изискване.

  Въпроси (*колко?*) за **a)** “стандартни” системи?

  b) критични системи?

- Систематичен подход, опит.

- Списък (таблица):

  Requirement x Checks (limited number) + comments

Списък (примерен) за анализа 1

- **Premature design**

  Включва ли изискването предварителна информация за проектиране или за реализацията?

- **Обединени изисквания (Combined)**

  Дали описанието на едно изискване описва единствено изискване или може да се раздели на няколко различни изисквания?

- **Ненужни изисквания (Unnecessary)**

  Дали изискването е ‘gold plating’? Т. е., дали изискването е козметична добавка към системата и в действителност не е необходимо.

- **Използване на нестандартен хардуер**

  Изискването предполага ли използването на нестандартен хардуер или софтуер?

Списък (примерен) за анализа 2

- **Следване на бизнес целите**

  Дали изискването е в съгласие с бизнес целите, дефинирани в уводната част на документа с изискванията?

- **Неяснота на изискванията (ambiguity)**

  Може ли да се прочете по различен начин от различните хора? Какви са възможните интерпретации на изискването?

- **Реалистичност на изискванията (realism)**

  Изискването реалистично ли е при дадената технология, която ще се използва за реализацията на системата?

- **Възможност за тестване на изискванията**

  Може ли изискването да се тества, т. е. дали е изразено по такъв начин, че QA инженерите могат да съставят тест, който да покаже дали системата изпълнява това изискване?

Техники за анализ на изискванията:

SMART(T) Изисквания

(„Software Engineering Notes“, vol.20, n.2,1995, pp. 42-47)

- **Specific (Специфични)**

  Отделни, т. е. множество елементи (изисквания/стъпки/и др.) не са обединени в един и не се дублират

- **Measurable (Измерими)**

  Могат да се определят количествено по някакъв начин (дори и да е само с TRUE/FALSE за една или повече качествени мерки)

- **Attainable (Достижими)**

  Могат да бъдат постигнати в рамките на физическото съществуване на системата (реалистичност на функционални и нефункц. изисквания)

- **Realisable (Реалистични)**

  Могат да бъдат реализирани в рамките на съществуващите ограничения (време, ресурси, и др.)

- **Traceable (Проследими)**

  Възможност за проследяване на изискването от неговото замисляне, спецификация до неговия дизайн, реализация и тестване; връзка с други.

- **Testable (Възможност за тестване)**

  Може да се провери чрез съответните средства за тестване

Характеризиране на добрите изисквания - 1

Характеристики на **добрите изисквания** – различни според различните автори. Тези характеристики могат да послужат и за приоритизиране на изискванията. Следните характеристики обаче са общоприети

Неделимо (Cohesive)	Изискването се отнася само за едно единствено нещо.
Пълно	Изискването е пълно изказано без липсваща информация .
Съгласувано	Изискването не противоречи на никое друго изискване и е напълно съгласувано с цялата външна документация.
Без връзки (**атомарни**)	Изискването е *атомарно*, т.е., не съдържа връзки.
Проследимо	Изискването отговаря на определена бизнес нужда или част от нея според искането на заинтересованите лица и е достоверно документирано.

Характеризиране на добрите изисквания - 2

Текущо Изискването не е остаряло с времето.

Осъществимо Изискването може да се реализира според ограниченията на проекта.

Недвусмислено Изискването е изказано сбито без използване на технически жаргон. То изразява конкретни факти. Интерпретира се по един единствен начин. Отрицателните твърдения са *забранени*.

Задължително Изискването представя характеристика, дефинирана от заинтересованите лица.

Проверимо Реализацията на изискването може да се определи чрез *един от четири възможни метода*: изследване, демонстрация, тест или анализ.

***Пример*:**

"Background Task Manager ще предоставя съобщения за статуса на редовни интервали не по-малки от всеки 60 секунди.“

- Какви са съобщенията за статус?
- При какви условия и по какъв начин се предоставят на потребителя?
- Ако се показват, колко дълго остават видими?
- Времевият интервал не е ясен и фразата “всеки" обърква израза.

Тези крайни интерпретации са съгласувани с първоначалното изискване, но определено не са това, което потребителят е имал предвид.

Поради тези проблеми, изискването не е проверимо.

Следователно …

***Решение*:**

Един начин за решение е да се пренапише предишното изискване, след като се получи повече информация от клиента:

1. Background Task Manager (BTM) ще показва съобщенията за статуса в **предназначената за целта област от потребителския интерфейс**.

1.1 Съобщенията ще се обновяват **на всеки 60 плюс** секунди след като започне обработката на background задачата.

1.2 Съобщенията ще остават **видими непрекъснато**.

1.3 Когато комуникацията с background процеса е възможна, BTM **ще показва какъв процент** от background задачата е завършен.

1.4 BTM ще показва съобщение “Завършен“, когато background задачата е изпълнена.

1.5 BTM ще показва съобщение, ако background задачата е спряла (stalled).

1. Защо изискването е разделено на множество изисквания?

2. Преработените изисквания не уточняват, как ще бъдат показвани съобщенията за статуса. Защо?

Това е въпрос на дизайна; ако бъде зададен върху изискването, ще ограничи дизайна и работата на разработчика.

Прибързаните ограничения върху опциите за дизайна пречат на програмистите и могат да доведат до suboptimal дизайн на продукта.

НО!

- *analysis paralysis*:
Не можете да удължавайте да усъвършенствате изискванията прекомерно.

- Целта е да се напишат изисквания, които са ***достатъчно добри***, за да позволят на екипа да продължи с дизайна и конструкцията при допустими нива на риска.

2. Взаимодействия между изискванията

- Една много важна цел на анализа на изискванията е да открие ***взаимодействията*** между изискванията и да посочи ***конфликтите*** *и* ***припокриванията*** между изискванията

- ***Матрицата на взаимодействие*** на изискванията показва как изискванията си взаимодействат помежду си. Изискванията се подреждат по редовете и по колоните на матрицата

  - ✓ За изисквания, които си противоречат, се попълва 1
  - ✓ За изисквания, които се припокриват, се попълва 1000
  - ✓ За изисквания, които са независими, се попълва 0

Матрица на взаимодействието

Requirement	R1	R2	R3	R4	R5	R6
R1	0	0	1000	0	1	1
R2	0	0	0	0	0	0
R3	1000	0	0	1000	0	1000
R4	0	0	1000	0	1	1
R5	1	0	0	1	0	0
R6	1	0	1000	1	0	0

За тази матрица дефинирайте:

a) взаимодействията на R1

b) независими изисквания

c) открийте броя на конфликтите за от всеки тип за всяко изискване. Задайте подходящи формули.

Взаимодействие на изискванията

- Когато не може да се вземе еднозначно решение за конфликт, се попълва 1. Защо?
- Какво означава съществуването на голям брой припокривания и/или конфликти?
- Размерът на матрицата е ограничен – (вероятно максимум 200 изисквания). Защо?

Приоритизиране на изискванията с използване на MoSCoW

- **MoSCoW** е техника за приоритизиране, която се използва за постигане на съвместно споразумение със заинтересованите лица за важността, която те поставят върху доставянето на всяко от изискванията

First developed by Dai Clegg of Oracle UK Consulting; in CASE Method Fast-Track (RAD)

M – ЗАДЪЛЖИТЕЛНО ТРЯБВА да го има.

S – ТРЯБВА да го има, ако въобще е възможно.

C – МОЖЕ да го има, ако не влияе на нещо друго.

W – НЯМА да го има този път, но БИХА искали да го има в бъдеще.

3. Договаряне на изискванията - 1

- **Обсъждане на изискванията**

  Изискванията, които са били отбелязани като проблематични, се обсъждат. Участващите заинтересовани лица представят своите възгледи за изискванията.

- **Приоритизиране на изискванията**

  Спорните изисквания се приоритизират, за да се идентифицират критичните изисквания и да се подпомогне процеса по вземане на решение.

- **Споразумение за изискванията**

  Откриват се решенията на проблемите при изискванията и се уговаря компромисен набор от изисквания. Обикновено това включва извършване на промени в някои от изискванията.

***Бележки*:** Разпространение на копие от резюмето на дискусията.
Ръководителят на срещата е независим човек !

Договаряне на изискванията - 2

- **Договарянето на изискванията е процес на обсъждане на конфликтите между изискванията и постигане на споразумение, с което всички заинтересовани страни са съгласни:** *организационни нужди, специфични потребителски изисквания, ограничения, бюджет на проекта и планиране*.

- **Разногласията за изискванията *са неизбежни***, когато една система има много заинтересовани страни. Конфликтите не са '*неуспехи*', а отразяват нуждите и приоритетите на различни заинтересовани лица

- При планирането на процеса за инженеринг на изискванията е важно да се остави *достатъчно време за преговори*. Откриването на приемлив компромис може да отнеме време

Срещи за преговори - стъпки

1. *Информационен етап*, на който се обяснява същността на проблемите, свързани с едно изискване.
2. *Етап на дискутиране*, на който участващите заинтересовани лица обсъждат как могат да се решат тези проблеми.

   Всички заинтересовани лица, които са засегнати от изискването, трябва да имат възможност да коментират. *На този етап могат да се назначат приоритети за изискванията.*

3. *Етап на разрешаване*, на който се уговарят действията, отнасящи се до изискването.

   Тези действия могат да бъдат изтриване на изискването; да се предложат специфични промени в изискването или да се извлече допълнителна информация за изискването.

Преговори за изискванията

- Скъп и времеотнемащ процес. (Защо?)

- Различните аналитици могат да имат **различни решения** на проблемите. Защо?

- Невъзможност тези процеси да се структурират, систематизират и автоматизират. Изискват се самостоятелни решения, опит..., дипломатичност

- Преговорите за изискванията понякога са повече въпрос на **политически решения**

Пример: Изискване на привилегирован достъп на мениджърския състав до локални данни. Същевременно, персоналът от охраната може да иска единствено мениджърът да има такъв достъп. Конфликтът би се решил в преговори.