1.

Egy kis vegyesbolt vezetője felkéri, hogy készítsen egy számítógépes programot, amely a bolt raktárkészletét tartja nyilván. Határozza meg a fejlesztendő szoftver funkcionális és minőségi követelményeit, valamint az alkalmazás szerkezeti felépítését!

– Mutassa be a programtervezés folyamatát!
– Nevezze meg és jellemezze a tervezés során készítendő dokumentumokat!
– Határozza meg a fejlesztendő szoftver funkcionális követelményeit!
– Határozza meg a fejlesztendő szoftver minőségi követelményeit!
Kulcsszavak, fogalmak (a vizsgakérdést tartalmazó lapon ezek nem szerepelnek, de a vizsgáztatók lapján igen, tehát kérdezhetik őket**):**
– Programtervezés: követelmények meghatározása, specifikáció, tervezés.
– Rendszerkövetelmények, felhasználói követelmények.
– A követelményfeltárás módszerei.
– Követelményspecifikáció.
A tételhez segédeszköz nem használható.

A programtervezés folyamata

A programkészítés lépései:

1.) A feladat megfogalmazása
2.) A feladat modellezése
3.) Az algoritmus meghatározása
4.) A szükséges hardver és szoftver környezet megállapítása
5.) Az algoritmus kódolása
6.) A program tesztelése, hibakeresése, javítása
7.) A hatékonyság növelése, optimalizálás
8.) A dokumentációk elkészítése
9.) A működő program karbantartása, felügyelete

1. A feladat megfogalmazása

A feladat megfogalmazása a megoldás első lépése, amelynek során egyértelműen rögzíteni kell (írásban) az elérendő célt a lehetséges pontossággal, a figyelembe vehető feltételekkel, különböző korlátok megadásával. Ezt a szakaszt analízisnek is nevezik.

A program megrendelője (a leendő felhasználó) megfogalmazza elvárásait, igényeit köznapi vagy a megrendelő szakmai nyelvén. Ideális esetben a megrendelővel a rendszerszervező tárgyal, akinek feladata a megrendelő szakmájának olyan szintű ismeret, hogy eldönthesse a feladatról, hogy az mennyire számítógépesíthető, milyen hardverigényű, létezik-e a probléma megoldására már kész szoftver. Továbbá milyen átalakításokat igényel a hagyományos folyamat a gépesítéshez. A rendszerszervező feladata az is, hogy közelítőleg megbecsülje a szoftver elkészítési idejét és költségét. Gyakori, hogy ebben a szakaszban a megrendelő még a kitűzendő feladat tekintetében is tanácsokat kér.

A program legfőbb részeit valamilyen módon meg kell terveznünk. E terv leírása különösen fontos a későbbi teszteléshez és a dokumentáláshoz. A program tervezője ekkor állítja össze a menüszerkezetet, elkészíti a képernyőterveket, meghatározza az adatábrázolási módokat.

2. A feladat modellezése

Annak ellenére, hogy a számítógép felhasználásával megoldott feladatok nagy része nem matematikai problémaként fogalmazódik meg, az eredmény előállításához legtöbbször matematikai és logikai eljárások sorozatos alkalmazására van szükség. A megfogalmazott feladatokhoz ezért általában matematikai modellt kell keresni.

Ezek a modellek - mint az összes egyéb modell is - a valóság többé-kevésbé hű másai, esetünkben matematikai eszközökkel megvalósított másolatok. Egy modell sohasem tükrözi a valóság minden apró részletét, hiszen mindig valamilyen céllal alkotjuk meg. A modellben azokra a tulajdonságokra koncentrálunk, amelyek a cél elérése szempontjából fontosak.

Kérdés, hogy tudunk-e a modellről olyan leírást készíteni, amelyet mindenki azonosan értelmez. Teljes általánosságban a kérdésre nagyon nehéz lenne válaszolni, de beérjük kevesebbel is, megelégszünk azzal, hogy csak azok értsék azonosan, akik a rendszer létrehozásában valamilyen módon részt vesznek. A kérdésre igen választ csak akkor kaphatunk, ha a leírás (reprezentáció) egyértelmű, pontos és kimerítő. Ezt úgy érhetjük el, hogy azok, akiknek a leírást értelmezniük kell, megegyeznek bizonyos tartalmi és formai szabályok szigorú betartásában, azaz a leírást **formalizálják.** *Formálisnak nevezzük az olyan reprezentációt, amely csak pontosan definiált fogalmakat, szerkezeteket és műveleteket használ, és a definíciók megadásának formáit is rögzíti.*

3. Az algoritmus meghatározása

Az algoritmus meghatározása, a megoldási folyamatnak az eredmény előállításáig vezető logikai felépítése igen fontos lépése a probléma-megoldásnak.

Az algoritmusoknak feltétlenül rendelkezniük kell néhány alapvető tulajdonsággal. Ezek:

Általánosság

A megadott algoritmus nem csak a szóban forgó speciális feladat esetén vezessen a probléma megoldására, hanem a hasonló jellegű feladatok széles körére szolgáltasson helyes eredményt.

Teljesség

A választott algoritmus a feladatkör minden lehetséges megoldását szolgáltassa. Ehhez az szükséges, hogy a bemenő adatok minden szóba jöhető értékére (értelmezési tartomány) felkészült legyen az algoritmus. A teljesség és az általánosság követelménye egymásnak ellentmondó ezért célszerű a két tulajdonságnak a megoldás szempontjából optimális kielégítését kitűzni.

Végesség

Az eredmény véges számú lépésben elérhető legyen.

Egyértelműség

Azonos körülmények között (ugyanolyan bemenő adatokkal) az eljárást megismételve azonos eredményeket kapjunk.

Értelmezettség

Legyen pontosan körülhatárolt az a bemenő adathalmaz, amelyre az algoritmus megoldást szolgáltat.

4. A szükséges hardver és szoftver környezet megállapítása

Hardver környezet: meghatározza, hogy milyen típusú számítógép, minimálisan milyen részegységekből (pl. memória, háttértár kapacitás) épüljön fel, esetleg milyen speciális kiegészítőkre van szükség a program futtatásához.

Szoftver környezet: az adott hardveren milyen egyéb programok futtatását igényli. A legfontosabb, hogy milyen operációs rendszer (és milyen verziójú) szükséges a futtatáshoz.

5. Az algoritmus kódolása

Kódolás: az algoritmus alapján a kiválasztott programozási nyelv utasításainak, szimbólumainak felhasználásával a program elkészítése. A programozási nyelvet célszerűen az adott feladathoz illeszkedően „illik” kiválasztani. Ma már az a gyakoribb, hogy olyan programnyelven készül el a program, amelyet ismer a fejlesztő. Ez nem mindig vezet a legjobb megoldáshoz. Minden programozási nyelvnek megvannak a sajátosságai, amelyek bizonyos mértékig behatárolják, hogy hol és milyen feladatokat célszerű az adott nyelven megvalósítani.

6. A program tesztelése, hibakeresése, javítása

Ez a fázis igényli a legtöbb figyelmet. Ha ugyanis a programozó nem járja végig az összes lehetséges végrehajtási utat, amit a programfutás során el lehet képzelni, akkor az esetleges hiba csak a program használata során derül ki. Ez viszont adatvesztéshez, akár anyagi kárhoz is vezethet. Először meg kell vizsgálni, hogy a program az elvárt funkcióknak eleget tesz-e. Ez a **tesztelés** folyamata.

Ha ennek során hibajelzést kapunk, hibás működést tapasztalunk, akkor az ezt okozó hibát meg kell keresnünk. Ez a **hibakeresés**.

Ha megtaláltuk a hibát, akkor azt javítanunk kell, ez a **hibajavítás**.

Ezután ismételten meg kell győződnünk a program működésének helyességéről, s mindaddig folytatni az előbbiekben leírtakat, míg tökéletesen nem fut a programunk.

7. A hatékonyság növelése, optimalizálás

A programozási tételek, alapalgoritmusok csupán a program előkészítésének lehetőségét biztosítják, a program eredményessége mindig a programozótól függ. A hatékonyságot befolyásolja a végrehajtási idő, a program és adatainak helyfoglalása (memóriában, lemezen), valamint a program bonyolultsága (számításigénye, processzorterhelése).

A tervezés során készítendő dokumentumok

8. A dokumentációk elkészítése

Minden emberi használatra előállított terméket dokumentálni kell.

A dokumentáció **célja**, hogy a termék életének különböző eseményeire
- tervezésre,
- gyártásra,
- értékesítésre,
- felhasználásra
- és karbantartásra

vonatkozóan a termékkel kapcsolatban álló ember számára a szükséges információt tartalmazza.

A szoftverfejlesztéssel kapcsolatban kétféle dokumentáció létezik, egyik a felhasználó számára készül, míg a másik a programozóknak, fejlesztőknek szól.

8.1 Fejlesztői dokumentáció

A programozóknak szól és a program fejlesztésével párhuzamosan készül, a továbbfejlesztéshez nagy segítséget jelent. Minden szoftverfejlesztési fázisnak meg van a maga dokumentációja. A **szoftver fejlesztését végigkísérő dokumentációk összességé**t nevezik fejlesztői dokumentációnak.

A fejlesztői dokumentáció részei:

- a feladatspecifikáció (a feladat megfogalmazása, pontosítása, általánosítása)
- a programterv (az algoritmus részletes leírása)
- hardver és szoftver feltételek
- a programban használt változók megadása
- az alprogramok (függvények, eljárások) hierarchiáját (és szerkezetét) megadó táblázat
- az alprogramok nevei és feladatai
- a tesztadatok listája
- a program fejlesztési lehetőségei és feltételei
- a program teljes forráskódja, valamint egy háttértáron őrzött példánya (Pl. CD-n mellékelve)
- a kész program

Ezeket a dokumentációkat a szoftver életciklusa során mindvégig meg kell őrizni, hiszen csak a dokumentáció segítségével lehet változtatást, javítást végezni a szoftveren.

A szoftverkövetelmények dokumentuma

A szoftverkövetelmények dokumentuma nem más, mint **a rendszerfejlesztőkkel szemben támasztott elvárások hivatalos leírása.**

8.2 Felhasználói dokumentáció

Ez a felhasználó számára készül. Önálló, a kész programmal együtt átadandó szöveges áttekintés, mely megad minden a program használatához szükséges információt.

A felhasználói dokumentáció célja, hogy **a termék rendeltetésszerű használatához szükséges valamennyi információt tartalmazza.** Ezért a szoftver használata előtt a felhasználónak ajánlott áttekinteni ezt a dokumentációt.

A különböző csoportba tartozó felhasználók és a felhasználói szakértelem különböző szintjeinek kiszolgálása érdekében a szoftverrendszerrel együtt legalább öt dokumentumot kell elkészíteni:

1. **Funkcionális leírás**: Megadja a program feladatát, leírja a rendszer által biztosított szolgáltatásokat. A bevezető kézikönyvet, amelyet elolvasva a felhasználó képes lesz eldönteni, hogy az adott rendszerre van-e szüksége.
2. **Telepítési dokumentum**: részletezi a rendszer telepítésének menetét. Leírja a lemezeket, amelyen a rendszer leszállításra került, az ezeken található állományokat, és a minimálisan szükséges hardverkonfigurációt és a szoftver feltételeket.

   Tartalmaznia kell telepítésre vonatkozó utasításokat is, valamint azt, hogy hogyan kell a konfigurációfüggő állományokat beállítani.
3. **Bevezető kézikönyv**: egy kötetlen bevezetést tartalmaz, leírva a rendszer betöltését, indítását, „szabályos” használatát, kezelését. Le kell írnia, hogy hogyan lehet elkezdeni a rendszer használatát, és hogy a végfelhasználók hogyan vehetik hasznát a rendszer megszokott lehetőségeinek. Tartalmaznia kell egy tipikus futtatás teljes leírását. A kézikönyvnek sok szemléltető példát kell bemutatnia, és információt kell tartalmaznia a hibajelzésekről, a hibák kijavításának és a hasznos munka újrakezdésének módszereiről.
4. **Referencia kézikönyv**: a rendszer lehetőségeit és azok használatát adja meg, leírja a hibaüzenetek listáját, a lehetséges okokat, valamint leírja, hogy adott hibák esetén mit kell tenni.
5. Egyes rendszerekhez **adminisztrátori kézikönyv** is csatolható. Ez a rendszer egy másik rendszerrel kölcsönhatásba lépésekor létrejövő üzeneteket írja le, és azt, hogy hogyan kell azokra reagálni. Ha a rendszer hardveréről is szó van, ez a kézikönyv elmagyarázhatja, hogy hogyan lehet a hardverrel kapcsolatos problémákat felismerni és megjavítani, hogyan kell új perifériát csatlakoztatni stb.

   Tartalmazza a program fejlesztési lehetőségeinek, illetve ennek feltételei leírását.

9. A működő program karbantartása, felügyelete

A felhasználónak éreznie kell, hogy az új program megvásárlása után nem hagyták magára. Tudnia kell, hogy ha gondja, problémája van, mindig van valaki, aki segít. Ezt túlzásba sem szabad vinni, mindig az igényekhez

Követelmények

A szoftverfejlesztés első lépése a specifikáció, vagy más néven a követelménytervezés, amelynek célja, hogy meghatározzuk milyen szolgáltatásokat követelünk meg a rendszertől, és hogy a rendszer fejlesztésének és működtetésének milyen megszorításait alkalmazzuk.

A követelménytervezés lépései:

- Követelmények feltárása, elemzése (cél a rendszer jobb megértése, követelmények elemzése)

- Követelményspecifikáció (az elemzési tevékenység során összegyűjtött információk alapján egységes dokumentum készítése)
- Követelményvalidáció (ellenőrzi, hogy mennyire valószerűek, következetesek és teljesek a követelmények)

Követelmény típusok

A követelményeknek két nagy csoportja van:

- **felhasználói követelmények:** diagramokkal kiegészített természetes nyelvű kijelentések arról, hogy mely szolgáltatásokat várunk el a rendszertől, és annak mely megszorítások mellett kell működnie. Az ügyfelek és a fejlesztők képviselői számára készülnek, akik nem rendelkeznek részletes technikai ismeretekkel a rendszerről.
- **rendszerkövetelmények:** a rendszerszolgáltatásokat és megszorításokat jelöli részletesen. A rendszerkövetelmények dokumentumának - amelyet funkcionális specifikációnak is hívnak - pontosnak kell lennie. Ez szolgálhat alapul a rendszer vásárlója és a szoftverfejlesztő közötti szerződésnek. Ezzel a vezető technikai személyzetet és a projektvezetőket ajánlott megcélozni. Mind az ügyfél, mind a vállalkozó alkalmazottai használni fogják.

Rendszerkövetelmények

A szoftver rendszerkövetelményeket gyakran felosztják funkcionális, illetve nem funkcionális követelményekre.

Funkcionális követelmények

Leírják, hogy a rendszernek

- milyen funkciókkal kell rendelkeznie
- hogyan kellene működnie (pl. aktualizálások, lekérdezések, jelentések, kimenetek, adatok, más rendszerekkel való kapcsolat)
- hogyan kell reagálnia bizonyos bemenetekre
- hogyan kell viselkednie bizonyos szituációkban. Itt gyakran azt is definiálják, hogy mit nem szabad tennie a rendszernek.

Ezek a követelmények a fejlesztett szoftver típusától és a leendő felhasználóitól függenek.

A funkcionális követelményt leíró specifikáció legyen teljes és ellentmondásmentes! A teljesség azt jelenti, hogy a felhasználó által igényelt összes szolgáltatást definiáljuk. Az ellentmondás mentesség azt jelenti, hogy ne legyenek benne ellentmondó megállapítások.

A gyakorlatban azonban a nagyméretű, összetett rendszerek fejlesztésénél gyakorlatilag lehetetlen a követelmények teljességét és ellentmondás-mentességét elérni. Ennek egyik oka, hogy nagyméretű, összetett rendszerek specifikációinak írásakor könnyű kifelejteni dolgokat, illetve hibát elkövetni. Másik ok, hogy a különböző kulcsszemélyeknek eltérő – és gyakran ellentmondó – igényeik vannak.

Egy rendszer funkcionális követelményei leírják, hogy a rendszernek milyen funkciókkal kell rendelkezni, hogyan kellene működnie (például aktualizálások, lekérdezések, jelentések, kimenetek, adatok, más rendszerekkel való kapcsolat). Ezek a követelmények a fejlesztett szoftver típusától, a szoftver leendő felhasználóitól függenek. Természetesen itt is megjelenik a követelmények másik megközelítése, miszerint kifejthetők mind felhasználói követelményekként, melyek rendszerint egészen absztrakt leírások, mind pedig rendszerkövetelményként, amelyek a rendszerfunkciókat részletesen írják le.

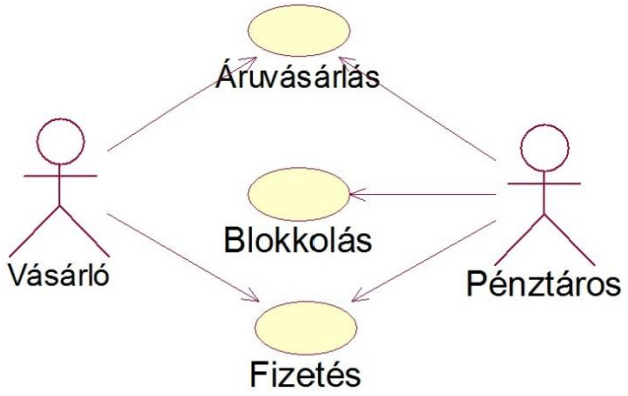
Nagyon fontos, hogy a rendszer funkcionális követelményt leíró specifikációjának tartalmaznia és definiálnia kell a megrendelő által igényelt összes szolgáltatást, azaz teljesnek és ellentmondásmentesnek kellene lennie. Az ellentmondás-mentesség szintén fontos, amely azt jelenti, hogy ne legyenek ellentmondó meghatározások a leírásokban. A gyakorlatban azonban a nagyméretű, összetett rendszerek fejlesztésénél gyakorlatilag

lehetetlen a követelmények teljességét és ellentmondás-mentességét elérni. Ennek egyik oka, hogy nagyméretű, összetett rendszerek specifikációinak írásakor könnyű kifelejteni dolgokat, illetve hibát elkövetni. Másik ok, hogy a különböző kulcsfiguráknak eltérő – és gyakran ellentmondó – igényeik vannak.

Funkcionális követelmények felderítése

Az fejlesztés következő lépése a funkcionális követelmények feltárása, erre bevált módszer az UML használati eset diagramjának elkészítése. Az ábráról gyorsan és egyértelműen leolvashatóak a kapcsolatok, azonban szöveges leírás szükséges a modell elemek leírására. Egyes UML szerkesztő alkalmazások a diagramok és a modell alapján képesek dokumentációt generálni, ami igen hasznos, sok munkát takarít meg.

Példa: egy áruvásárlás során zajló eseményeket összefoglaló használati eset diagram



Ennek a létrehozása több lépésben szokott elkészülni, az nem baj, ha elsőre nem jut eszünkbe minden funkció (a módszerek is csak ajánlásokat tesznek, mikor legyen kész a teljes funkciólista bizonyos része). A diagram pont arra jó, hogy a megrendelő, vagy kollégánk a diagram alapján megérti elképzelésünket és ki tudja egészíteni a listánkat. A diagram egyszerűsége miatt a megrendelő is tudja használni azt. A dokumentációnak egységesnek kell lennie minden azonos típusú elemre. A használati esetekről tudnunk kell, hogy mennyire fontos, milyen előfeltételei, utóhatásai vannak. Érdemes a használati eseteket összefoglalni, azaz definiálni a teljes listát.

A fejlesztendő raktárkészlet-nyilvántartó rendszer funkcionális követelményei

A létrehozandó alkalmazás jellegét tekintve egy nyilvántartó program, tehát alapvetően adatbázis-kezelési funkciói lesznek:

- adatbázis-kapcsolat felépítése (saját adatszerkezet, vagy DBMS-en keresztüli – pl. Access)
- adatfelvitel (pl. áruk és jellemzőik)
- adatmódosítás
- adatok törlése
- adatok listázása, kinyerése (a legjellemzőbb funkció adatkezelő programoknál)

Ezek konkrét kifejtése a létrehozandó vegyesbolti raktárnyilvántartó programhoz például ilyen lehet:

- adatfelvitel: a program képes legyen megjeleníteni egy megfelelően kialakított felhasználói felületet (lehetőleg grafikusat – GUI), majd ezen a felvitelhez szükséges elemeket megjeleníteni, melyek segítségével a felhasználó megadhatja a rögzítendő adatokat (pl. termékkód, terméknév, kategória, gyártási idő stb.), ezeket a program legyen képes ellenőrizni, hiba esetén a felhasználót tájékoztatni és segíteni, helyes adatok esetén pedig azokat az adatbázisba rögzíteni
- adatmódosítás, törlés, listázás: az előző funkció megfogalmazáshoz hasonlóan kifejthető
- a program funkciója az adatfájl, vagy adatfájlok kezelése is: vagy közvetlenül, vagy pl. Access adatbázis-motor segítségével – a dolog miként tét a gyakorlaton áttekintett programokból idézzétek fel (pl. Access vagy MySQL adatbázis-kezelő esetén lekérdezés SQL parancsokkal)!
- további funkció lehet pl. jelentések, kivonatok készítése, nyomtatása, exportálása, áru rendelés előkészítése, távlati becslések nyújtása az áruk eladási statisztikája alapján stb.

Nemfunkcionális követelmények (a minőségi követelmények megadásakor, illetve a felelet kiegészítéseként fordulhat elő**)**

A nemfunkcionális követelmények a funkcionális követelményekkel ellentétben nem közvetlenül a rendszer által biztosított specifikus funkciókkal foglalkoznak, hanem inkább a rendszer egészére vonatkozó eredő rendszertulajdonságokra koncentrálnak. Mit is jelentenek ezek? Példaként néhányat említve: megbízhatóság, válaszidő, tárfoglalás, rugalmasság, robosztusság, hordozhatóság, stb.

A követelmények ezen csoportja gyakran kritikusabb, mint az egyedi funkcionális követelmények csoportja. Ha nem teljesítjük ezeket, vagy nem kerültek megfogalmazásra konkrét követelményként, gyakran a teljes rendszert használhatatlanná tehetik. Példaként említhetünk egy banki rendszert, amely nem teljesíti a vele szemben támasztott összes biztonsági követelményt.

Néhány fontos tényező, amely felvetheti a nemfunkcionális követelményeket:

- felhasználói igények,
- költségvetési megszorítások,
- a szervezeti szabályzat,
- más szoftver- vagy hardverrendszerekkel való együttműködés igénye
- külső tényezők, pl. biztonsági szabályozások, adatvédelmi rendelkezések

Mit is jelent a szoftverminőség?

A szoftvertermékek esetén a minőség általános definiálása nehéz és sokszor nem egyértelmű. Következik ez abból is, hogy különböző termékek célja, funkcionalitása, felhasználási körülményei nagyon eltérőek egymástól, és a termék használata során bekövetkezett hibák is nagyon sokféle hatást fejthetnek ki. Mást és mást jelent a minőség a szoftverek gyártója, kereskedője, üzemeltetője, felhasználója számára is. A funkcionális megfelelőségen túlmenően sokszor nagy szerepük van az ún. nem-funkcionális jellemzőknek (pl. megbízhatóság, üzemeltethetőség, terhelhetőség, biztonság, hordozhatóság, stb.) is.

A fejlesztendő rendszer minőségbeli követelményei

Célszerű ebből az ábrából kiindulva felvázolni a követelményeket:

- funkcionalitás: alkalmasság - mennyire alkalmas a feladatára (elsősorban a megrendelő aspektusából, másodsorban a többi lehetséges felhasználó szempontjából), biztonság – mennyire védetten kezeli az adatokat, a hozzáférési szinteket jól el tudja-e különíteni, funkcionális megfelelőség – a funkcionális követelményeknek teljes mértékben eleget tesz-e
- megbízhatóság: hibatűrő képesség - az esetleges felhasználói hibás adatbevitelt hogyan kezeli le, futási idejű hibáktól védett-e, operációs rendszer felől érkező jelzésekre reagál-e (pl. megtelt a lemez, fájl olvasási/írási hiba, víruskereső program vakriasztása esetén sem engedi törölni az

adatbázisát stb.), helyreállíthatóság – program, vagy rendszer összeomlás esetén újra működőképes-e, pl. az adatbázisról tárol-e biztonsági mentést
- használhatóság: mennyire könnyen tanulható, kezelhető a felhasználók szempontjából
- hatékonyság: erőforrás igény, kellően gyors működés
- karbantarthatóság: stabilitás, módosíthatóság (későbbi fejlesztések esetén)
- hordozhatóság: adaptálhatóság – később megjelenő operációs rendszereken is megbízhatóan működőképes lesz-e, telepíthetőség – az összes, a specifikációjában megadott rendszeren gond nélkül telepíthető, együttélés – a napi használat során nem okoz problémákat, nem akad össze más programokkal, op. rendszer frissítésekkel, illetve ilyen gond esetén figyelmezteti-e a felhasználót egy tényleges hibát megelőzendő, kiválthatóság – pl. ha a program maga valamiért nem használható, az adatbázisa másik programmal is kezelhető-e stb.

Követelmények feltárása, elemzése

Az elemzés, analízis, más néven a követelmények megfogalmazása a szoftverfejlesztés első lépcsőfoka, ami a követelmény specifikációt eredményezi majd.

A követelményelemzés tevékenységei:

- felhasználói igények elemzése
- rendszerelemzés
- megvalósíthatóság vizsgálata

A követelményelemzés tevékenységei: felhasználói igények elemzése

- követelmények összegyűjtése
- osztályozás (a követelmények strukturálatlan gyűjteményét összefüggő csoportokba szervezi),
- ellentmondások feloldása (a különböző kulcsemberek számára más és más dolog lehet fontos),
- fontossági sorrend felállítása
- követelményellenőrzés (kiderüljön, hogy a követelmények teljesek-e, ellentmondásmentesek-e, és összhangban vannak-e azzal, amit a kulcsfigurák a rendszertől elvárnak).

Az elemzés eszközei, módszerei:

- dokumentumok begyűjtése, tanulmányozása, elemzése,
- ismeretszerzés kérdőívek segítségével (kérdőívek készítése, kitöltése, kiértékelése),
- ötletbörzék, “brain stormingok” szervezése,
- egyéni interjúk a felhasználókkal:
- **etnográfia**: felhasználók tevékenységének figyelése
- elméleti kutatás, analógia más rendszerekkel,
- "gyakorlat" az adott alkalmazási területen,
- Ishikawa (halszálka) diagram készítése
  Az egyes feladatok elvégzésekor adódó problémák feltárására, problémák és az ok-okozati összefüggések elemzésére szolgál. A diagramon fel kell tüntetni a tevékenységeket, és a tevékenységek végzésekor valószínűleg felbukkanó problémákat.

A követelményelemzés tevékenységei: rendszerelemzés

- helyzetfelmérés, piackutatás (milyen termékek vannak már, amelyek ilyen célt szolgálnak),
- szakterület megismerése (pl. ha áruházi rendszerre van szükség meg kell vizsgálni, hogyan működnek az áruházak), amelynek lépései:
  - megfigyelés, a működési folyamatok végigkövetése,
  - interjúk a folyamatban résztvevőkkel,

- a jelenlegi rendszer, továbbá a már használatban lévő rendszerek teljes megértése,
- a meglévő dokumentációk teljes áttekintése,
- tanulmányok készítése.

A követelményelemzés tevékenységei: megvalósíthatóság vizsgálata

A megvalósíthatósági tanulmányban meg kell becsülni, hogy a felhasználók kívánságai

kielégíthetőek-e az adott szoftver- és hardvertechnológia mellett. Mennyibe fog kerülni a rendszer?

Ennek az elemzésnek relatíve olcsónak (esetleg ingyenesnek) és gyorsnak kell lennie. A tanulmány eredménye, hogy érdemes-e folytatni a munkát.

Az eredményeket ajánlott egy jelentésben összefoglalni, ami egy rövid, tömör tanulmány, amely számos kérdést próbál megválaszolni:

- Támogatja-e a rendszer a vállalat általános célkitűzéseit?
- Megvalósítható-e a rendszer a jelenlegi technológiával adott költségen belül és adott ütemezés szerint?
- Integrálható-e a rendszer más, már használatban lévő rendszerekkel?

Információforrást jelenthetnek annak az osztálynak a vezetői, ahol a rendszert használni fogják, azok a rendszerfejlesztők, akik már ismerik az olyan típusú rendszereket, amelyeket itt is terveznek használni, a technológiai szakértők, a rendszer végfelhasználói, stb.

Követelmény specifikáció

A szoftverkövetelmény specifikáció a rendszerfejlesztőkkel szemben támasztott eljárások hivatalos leírása. Ajánlott tartalmaznia a felhasználói követelményeket és a rendszerkövetelményeket. Úgy kell felépíteni, hogy a rendszer vevői és a szoftverfejlesztők számára is egyaránt használható legyen.