

Schuster György

SZOFTVERTERVEZÉS

DOI: 10.32560/rk.2019.1.2

A szoftver fejlesztés rendkívül összetett, bonyolult folyamat, továbbá köszönhetően a termékek rendkívüli bonyolultságának a tesztelés semmilyen szinten nem lehet teljes. A kívánt minőség a fejlesztés folyamatának szabályozásában és ennek szigorú követésében rejlik. Ennek a folyamatnak az egyik legfontosabb lépés a tervezés. A tervezés minden biztonságkritikus esetben egy szabvány és szabályrendszer alapján történik. Ez repülés szempontjából kritikus szoftver fejlesztésnél a DO-178C szabvány, amely a teljes szoftverfolyamatot felöleli. A cikk a legfontosabb tervekkel és azok jelentőségével foglalkozik mindvégig szem előtt tartva a biztonságkritikus fejlesztés szempontjait.

Kulcsszavak: szoftvertervezés, biztonságkritikus rendszerek

BEVEZETÉS

„Ha nem tervezel, akkor a kudarcot tervezed” [2] ez az idézet tökéletesen kifejezi a szoftvertervezés fontosságát. Maga a tervezés is egy folyamat, amely lépésről lépésre biztosítja, hogy a fejlesztők az adott eljárásoknak megfelelően állítsák elő a kérdéses terméket. A tervezés során nem csak a konkrét szoftver elemekre kell kitérni, hanem az életciklusnak és az adott fejlesztési modellnek megfelelően a kapcsolódó további szinteket is figyelembe kell venni.

Biztonságkritikus szoftverek tervezése az alkalmazott szabvány szerinti tervezési folyamatot, a feladatnak megfelelő és a kapcsolódó projekt terveknek a kidolgozását foglalja magában.

A szabványnak megfelelés és általában a fejlesztési minőségbiztosítás megköveteli, hogy a tervek folyamatosan és maradéktalanul dokumentálják. Ha a tervek az alkalmazott szabvány céljainak elérése szerint készítik, akkor ezen elvek követése biztosítja a célok elérést. A tanúsító szervezet a termék megfelelőségét alapvetően a tervek minőségére és teljességére alapozva adja ki, természetesen a tesztelési és használati mérési eredményeket is figyelembe véve.

ÁLTALÁNOS TERVEZÉSI SZEMPONTOK

A szoftver tervezésénél az alábbi szempontokat célszerű betartani [2]:

1. gondoskodni kell arról, hogy minden kitűzött cél és járulékos célkitűzés le legyen fedve. Ilyenkor célszerű a kiválasztott szabvány előírásai és a célok közötti megfeleltetés elkészítése;
2. a tervek oly módon kell elkészíteni, hogy a projektben résztvevő összes csapat (kód készítők, teszterek, minőség irányító stb.) számára érthető legyen;
3. ki, mikor és hogyan. Ez nem okvetlenül időterv, hanem egymásra épülés;
4. a tervezés lehetőleg előzze meg a megvalósítás lépéseit, mert ellenkező esetben nehezen módosíthatók az esetlegesen nem pontosan teljesített célkitűzések;

5. a tervek legyenek minden aspektusukban következetesek. A belső ellentmondásokat fel kell oldani a lehető legnagyobb mértékben;
6. meg kell határozni a tervek egymás közötti kapcsolódási és átmeneti kritériumait, amit a tervezés során be kell tartani;
7. ha több szoftvert is tervezünk segítséget nyújthat egy – akár vállalati szintű – egységes tervezési sablon;
8. a tanúsítót a tervezési fázis korai szakaszában célszerű bevonni (figyelembe véve a függetlenség elvét);
9. a menedzsmenet be kell vonni a projekt tervezési szakaszában. Egy támogató menedzsmenet elősegítheti a projekt sikerét [4];
10. a terveknek megfelelően részleteseknek kell lenniük. A terveknek azonosítaniuk és össze kell foglalniuk az eljárásokat, de az eljárásoknak nem kell teljes részletességben szerepelniük. Azonban szükséges az adott eljárások fontosságának kiemelése;
11. a tanúsító hatóságot informálni célszerű a terv benyújtása előtt a kérdéses projektről (de a pártatlanság biztosítása fontos);
12. a terv részeinek szétosztásánál a végrehajtó csapatoknak pontosan érteniük kell, az adott fázis tartalmát és helyét a teljes projektben;
13. a terv legyen kellően rugalmas, ha szükséges legyen változtatható, de ez ne jelentse azt, hogy a terv nem kellően részletes vagy mély;
14. A tanúsítás során előforduló változtatási igényekre is célszerű felkészíteni a terveket. A változtatásokat szintén tanúsítani kell.

A fenti szempontok mellett öt szoftverrel kapcsolatos terv elkészítése szükséges a projekt megvalósítása szempontjából [2].

A TERVEK

A szoftver tanúsítási terv

A szoftver tanúsítási terv egy olyan terv, amit a minősítő hatóság mindenképpen megvizsgál. Ezért – látszólag – legjobb megoldás, hogy ez a terv a lehető legkorábban elkészüljön azért, hogy a minősítést végző személyek időben megkaphassák [2].

Azonban azt tény is célszerű szem előtt tartani, hogy nagyon sok esetben a leg gondosabb tervezés esetén is előfordulhat, hogy lesznek olyan folyamatok, amelyek a korai szakaszban nem derülnek ki, vagy emberi hibából, vagy egyszerűen azért, mert nem voltak előre láthatók. Ilyenkor előfordulhat, hogy az egész anyagot át kell dolgozni, ami időbeli, anyagi problémákat jelent.

A tanúsítási terv egy magas szintű leírást ad a projektről és kellő részletességgel elmagyarázza, hogyan teljesülnek a célkitűzések és a magvalósítás, illetve a célkitűzés hogyan teljesíti a szabvány előírásait.

Ez a terv az a dokumentum, amelyet a minősítő hatóságnak biztosan be kell nyújtani és többi négy további terv összefoglalását is tartalmazza. Természetesen ez nem jelenti azt, hogy az alábbiakban ismertetett tervek minden részletét tartalmaznia kell, de ezekről pontos és következetes összefoglalót kell adnia [2][3].

Ideális esetben a tanúsítási terv elég részletesen taglalja a folyamatokat. Ezáltal a tanúsító hatóság megértheti a folyamatokat, és pontos értékelést tud adni a megfelelésről.

A tanúsítási tervet világosan, tömören és érthetően kell megírni. Minél világosabb és érthetőbb a dokumentum, annál könnyebb a projektet átlátni és annál valószínűbb a hatóság jóváhagyása.

Milyen elemei vannak a tanúsítási tervnek [3]:

- ➔ ütemterv: ez tartalmazza a szoftver fejlesztés és jóváhagyás ütemezését. Célja, hogy elősegítse mind a fejlesztés, mind a jóváhagyás erőforrásainak tervezését. Általában magába foglalja a főbb mérföldköveket, például a tervek elkészülési idejét, a tesztelés megkezdésének idejét,
- ➔ egyéb megfontolások: minden fontos információt közöljük a minősítő hatósággal. Kezéljük a meglepetéseket. Olyan dolgokat is ismertetni kell például, hogy egy külső szoftver beszállító milyen kódjait használjuk és nem használt részeket milyen módon deaktiváljuk.

Célszerű a célkitűzéseket és fontos információkat mellékletben közölni.

Nagy figyelmet kell arra is fordítani, hogy a minősítő hatóságban fel se merüljön a gyanú, hogy bármilyen információ elrejtésre került. A pontos és teljességre törekvő magatartás növeli a bizalmat.

Szoftverfejlesztési terv

Ez a terv összefoglalja az összes szoftverfejlesztési lépést, beleértve a követelményeket, a tervezési, kódolási és integrációs fázisokat, továbbá az ellenőrzési és tesztelési eljárásokat a tesztelés minden fázisában.

Ez a terv a fejlesztésben résztvevő szakemberek számára készül, akik végrehajtják a fenti tevékenységeket. A szoftverfejlesztési tervnek kellően részletesnek kell lennie, hogy jó irányt biztosítson a fejlesztési folyamatok végrehajtására, de ne legyen annyira részletes, hogy akadályozza a döntéshozatal és a végrehajtás lépéseit. Gyakorlati tapasztalat, hogy a megfelelő szintet eléggé nehéz „eltalálni”. A terv vagy nagyon felületes, vagy túlságosan részletes [2].

A szoftverfejlesztési tervnek három fő elem leírását kell tartalmaznia:

1. a fejlesztéshez használt szabványok (pl. ISO 61508, vagy DO-178C);
2. a szoftver életciklus modelljének leírását az átmeneti fázisok ismertetésével;
3. a fejlesztési környezet ismertetését (a követelmények kezelésének, a tervezés és a kódolás eszközeinek, valamint a tervezett fordító, linker, tömörítő programok kiválasztását és az alkalmazott hardver platform leírását).

A szabványok meghatározzák azokat a korlátozásokat, amelyek segítenek a fejlesztőknek abban, hogy elkerüljék a biztonsági vagy a szoftver funkcionalitását hátrányosan befolyásoló csapdákat.

A szabványokat és korlátozásokat alkalmazni kell az alkalmazott módszertanra vagy nyelvre (lásd MISRA C-re) [4].

A szoftverfejlesztési tervnek azonosítania kell a szoftver életciklus modelljét. Biztonság szoftverek fejlesztése esetén az életciklus modell nevének azonosítása mellett ajánlott a mo-

dell magyarázata is, mivel sajnos az életciklus modelleknek többféle értelmezése létezik. Az életciklus grafikája és az egyes fázisokra létrehozott adatok hasznosak.

Kerülendők azok az életciklus modellek, amelyek a gyors fejlesztést helyezik előtérbe a módszeresen betartott biztonsági eljárásokkal szemben. További tapasztalat, hogy néhány projekt egy életciklus-modellt azonosít a terveikben, de valójában valamilyen mást modellt követ.

A szoftverfejlesztési tervben dokumentálni kell a fázisok közötti átmenetek feltételeit. Ez részletezi az adott fázisok belépési és kilépési feltételeit. Ezeket táblázatos formában célszerű összefoglalni.

A fejlesztői környezet és a további alkalmazott szoftverek definiálásának kérdése is elengedhetetlen követelmény. Egy nagyobb projekten nem egy ember dolgozik, hanem sokszor több tíz – esetleg – több száz is. Ekkor az egységes eszközök és módszerek elengedhetetlenek. Lényeges az is, hogy milyenek a futtatható kód előállításának eszközei, ide értve a szerkesztő, fordító és linker programokat, de nem utolsósorban – ha az engedélyezett – az alkalmazott könyvtári elemek és idegen kódok minősége.

A környezet pontos definiálásának a további nagyon fontos következménye, hogy az eredményezett kódok reprodukálhatók.

Szoftver ellenőrzési terv

A szoftverfejlesztési terv összefoglalót nyújt a követelményekről, a tervezési, kódolási és integrációs előírásokról (például a szakértői értékelésről). A szoftver ellenőrzési terv rendszerint további részleteket ad a felülvizsgálatokról, beleértve a felülvizsgálati folyamat részleteit és fázisait, az ellenőrző listákat, a szükséges résztvevőket. A szoftver ellenőrzési terv adott fázisa a szoftver készültségi szintjétől függ [2].

A szoftver ellenőrzési terv meghatározza és indokolja az ellenőrzést végző csoport összetételét és – ami rendkívül fontos – a fejlesztőktől a függetlenség mértékét. A legtöbb projektnek önálló és független ellenőrző csoportja van. Ez nem okvetlenül szükséges, például a DO-178C szabvány nem követeli meg.

Tapasztalat azonban az, hogy a független ellenőrző csoport alkalmazása célszerű [1][4].

A szoftver és a szoftver projekt ellenőrzési dokumentum (kicsit több, mint a terv) tartalmazza a véleményeket, az elemzéseket és a tesztelési eredményeket.

A szoftver ellenőrzési terv magyarázza a felülvizsgálati folyamatot, beleértve a részletes felülvizsgálati eljárásokat, a felülvizsgálatok fázisainak átmeneti kritériumait, valamint az ellenőrző listákat és fontos sablonokat.

A szabványok gyakran tartalmazznak, vagy hivatkoznak ellenőrző listákat a követelmények, a tervezés és a kód felülvizsgálatához.

Ezek az ellenőrző listák jellemzően olyan elemeket tartalmaznak, amelyek biztosítják a szükséges célok értékelését, beleértve a nyomon követhetőséget, a pontosságot és a következetességet és azt, hogy szabványokban leírt követelmények teljesülnek.

Fontos, hogy a szoftver ellenőrzési tervnek azonosítani kell azokat az eljárásokat és módszereket, amelyek a szoftver megváltoztatása – például javítás - után az újbóli ellenőrzéshez szükségesek.

További fontos kérdés a régebben fejlesztett és minősített szoftver egységek beépítésével kapcsolatos ellenőrzési tevékenységek elírása (pl. megváltozott környezet kérdése és egymásra hatások).

Szoftver konfigurációs és kezelési terv

Ez a terv elmagyarázza, hogyan kezelendő az életciklus alatt képződő adatok konfigurációja a szoftverfejlesztés és az ellenőrzés során. A szoftverkonfiguráció kezelés a tervezési fázisban kezdődik, és az egész szoftver életciklusára kiterjed - egészen a szoftver telepítését, karbantartását és visszavonását követően [2].

A szoftver konfigurációs és kezelési terv elmagyarázza az konfiguráció kezelés eljárásait, eszközeit és módszereit, amelyek a fejlesztéshez és az üzemeltetéshez szükségesek. A egyes tevékenységek röviden a következők [3]:

1. konfiguráció azonosítás: az egyes konfigurációhoz tartozó elemek (beleértve az egyedi forráskódot és a tesztfájlokat) egyedileg azonosíthatóak legyenek. Az egyedi azonosítás jellemzően tartalmazza a dokumentum-, adat-, verzió és alverzió számokat;
2. alapvető jellemzők és nyomon követhetőség: a terv elmagyarázza az alapvető jellemzők létrehozásának és azonosításának módját. Ebbe beleértjük a hivatalosan előírt jellemzőket is a tanúsítás szempontjából;
3. probléma bejelentés: egy új verzió létrehozásához egyértelműen és részleteiben jelenteni kell azt a problémát, amely egy új verzióhoz vezet. Erre a projekt folyamán, vagy vállalati szinten célszerű egy egységes mechanizmust fent tartani;
4. a változások ellenőrzése: elmagyarázza, hogyan szabályozzák az életciklus-adatok változásait annak biztosítása érdekében, hogy a változó-illesztőprogramokat a kérdéses adatelem megváltoztatása előtt hozzák létre és hagyják jóvá. Ez szorosan kapcsolódik a problémabejelentési folyamathoz;
5. a változás felülvizsgálata: ennek a folyamatnak az a célja, hogy biztosítsa, hogy a változtatásokat megfelelően tervezzék, jóváhagyják, dokumentálják, megfelelően végrehajtsák. Ezt általában egy változás-ellenőrző testület felügyeli, amely jóváhagyja a változások végrehajtását, és biztosítja, hogy a módosítás lezárása előtt az ellenőrizhető legyen;
6. konfigurációs állapot dokumentálása: a projekt folyamán folyamatosan figyelni kell a projekt állapotát. Ezt a konfigurációs állapot vizsgálata és dokumentálása folyamatosan követi. Az állapot ismerete és a problémák kezelése elengedhetetlen a minősítési eljárásban;
7. archiválás, kiküldés és visszakövethetőség (konfiguráció kezelés): általában erre vállalati előírások vannak. A tervnek a következő alapvető szempontokat kell figyelembe venni:
 - archiválási kérdések: a terv megadja, hogy mikor, mit és hogyan kell archiválni,
 - az adatok és a kódok kiadási rendje: vannak nyilvános, korlátozott és belső adatok. A terv ezeknek a tárolási, hozzáférési és kiadási rendjét adja meg.
 - a visszakeresés módja: hogyan, ki és mikor milyen adatokat keressen vissza. A terv ezt részletesen ismerteti.
8. szoftver terhelési terv, a terv elmagyarázza a szoftver célrendszerbe töltésének módját;

9. életciklus és környezet ellenőrzése, a fejlesztő csapat tagjai csak az ellenőrzött és előírt életciklus szerint és környezetben dolgozzanak;
10. az életciklus adatainak kezelése.

A beszállítók esetén, beleértve az alvállalkozókat, vagy a felhasznált külső erőforrásokat, a terv kifejti a beszállító hasonló szoftver folyamatait is. A beszállítónak sokszor külön szoftver konfigurációs és kezelési terve van. Vagy a beszállító követheti az a megrendelő tervét. Ha több ilyen tervet alkalmaznak a külső partnerek, akkor azokat konzisztenciájuk és kompatibilitásuk érdekében részleteiben felül kell vizsgálni [2].

Szoftver minőségbiztosítási terv

A szoftver minőségbiztosítási terv leírja a szoftverminőség csapat tervét arra, hogy szoftver megfeleljen a jóváhagyott terveknek és szabványoknak és célkitűzéseknek. Az szoftver minőségbiztosítási terv magában foglalja a minőségbiztosítási csoport szervezetét az adott fejlesztő cégén belül, és kiemeli a függetlenségüket [2].

Az szoftver minőségbiztosítási terv ismerteti a szoftverminőségi mérnök szerepét is, amely jellemzően a következőket tartalmazza:

- ➔ a tervek és szabványok felülvizsgálatát;
- ➔ a szakértői értékelést, ennek folyamatában való részvételt annak biztosítására, hogy a szakértői értékelés megfelelő legyen;
- ➔ a tervekben meghatározott átmeneti kritériumok érvényesítését;
- ➔ a környezet ellenőrzését azért, hogy a fejlesztők és a hitelesítők a megfelelő módon használják a meghatározott eszközöket, beleértve a fordítót, a linkert, a teszteszközöket;
- ➔ a tervek betartásának felügyeletét;
- ➔ a szoftverek fejlesztést és tesztelést megfigyelésének módját;
- ➔ a legfontosabb dokumentumok jóváhagyását;
- ➔ részvételt a változáskezelő fórumon.

Az szoftver minőségbiztosítási tervnek meg kell adnia az minőségi-nyilvántartás tárolási módját, beleértve, hogy ki, hol, miképpen tárolja a kérdéses nyilvántartásokat és ezek hogyan érhetők el.

Fejlesztési szabványok

A fenti öt terv mellett figyelembe kell venni a vonatkozó szabványokat is. Bár a tervek ezekre hivatkoznak, de ennek ellenére célszerű ezeket külön kiemelni. A szabványok vonatkozó szabályokat és korlátozásokat tartalmaznak, amelyek segítik a fejlesztőket a munkájuk megfelelő elvégzésében, továbbá elősegítik azt, hogy elkerüljék azokat a „csapdákat”, amelyek negatív hatással bírnak a biztonságra. Ezek a szabványok a következők [2]:

- ➔ szoftverkövetelmények szabványai. Meghatározzák a fejlesztésre szolgáló módszereket, eszközöket, szabályokat és korlátozásokat. Ezek a szabványok inkább felsőbb szintű előírásokat tartalmaznak és jellemzően útmutatást nyújtanak a követelményeket összeállító csapat számára [3];

- szoftvertervezési szabványok. Ezek a szabványok szabályokat és korlátozásokat tartalmaznak, amelyek segítik a fejlesztőket a munkájuk megfelelő elvégzésében. Ez a tervezési és az „alacsonyabb szintű” tevékenység iránymutatása [3];
- szoftver kódolási szabványok. Ezek az előző két szabványhoz hasonlóan útmutatást adnak a kód íróinak. Megmagyarázzák, hogyan kell megfelelően használni az adott nyelvet, korlátozzák a nyelv bizonyos konstrukcióit, amelyeket nem ajánlatos a biztonsági szempontból kritikus tartományban használni, egyértelműsítik az elnevezési szabályokat, megmagyarázzák a globális adatok használatát, valamint segítenek olvasható és karbantartható kódot fejleszteni [3].

További tervek

- Eszköz minősítési terv. Ha egy fejlesztő eszközt minősíteni kell, akkor ez tervezést igényel. A tervezési információkat szoftverminősítési terv is tartalmazza. Más projektekben újrahasznosítható eszközök esetében azonban további minősítési tervekre lehet szükség [2];
- Projektmenedzsment terv. Megadja a csapatok és a csapattagok felelősségét, konkrét feladatait, a részletes ütemtervet, az aktuális állapotokat és a mérés eszközeit [2];
- Követelmény kezelési terv. Ezt a tervet a szoftver fejlesztési tervvel együtt fejlesztik ki azért, hogy a dokumentáció, a feladat elosztás és követés megfelelő legyen [2];
- Vizsgálati terv. Ez a terv ismerteti a teszt stratégia részleteit, beleértve a szükséges be rendezéseket, eljárásokat, eszközöket, teszteset elrendezést, a nyomkövetési stratégiát stb. Ez általában része a szoftver ellenőrzési eljárások dokumentumnak, de esetenként lehet különálló terv is. A vizsgálati terv gyakran tartalmazza a teszteléssel kapcsolatos ellenőrző listákat és kritériumokat, amelyekkel a tesztelő-ellenőrző csapat számára biztosítja, hogy képesek legyenek a hivatalos tesztek végrehajtására [2].

ÖSSZEFOGLALÁS

A szoftver kritikus sikertényező, használata elkerülhetetlen. Kimerítő tesztelése emberi időben lehetetlen, a statisztikai adatok, amelyek a klasszikus műszaki világban rendelkezésre állnak, csak igen korlátozottan alkalmazhatók. Így a klasszikus statisztikai elemzések nem, vagy csak hozzávetőlegesen hajthatók végre.

Biztonságkritikus esetben ezért a hangsúly a „gyártásról” áttevődött a tervezésre. A jelen publikáció azt taglalta, hogy milyen tervekre van szükség és azok mit tartalmaznak [1].

Ez a publikáció csak az alapvető ismeretekkel foglalkozik, konkrét esetekben még számos más terv és tervrészlet szükséges lehet.

FELHASZNÁLT IRODALOM

- [1] Fagan, M. E. (1976). "Design and code inspections to reduce errors in program development". IBM Systems Journal. 15 (3): 182–211 DOI: <https://doi.org/10.1147/sj.153.0182>
- [2] Leanna Rierson DEVELOPING SAFETY-CRITICAL SOFTWARE A Practical Guide for Aviation Software and DO-178C Compliance DOI: <https://doi.org/10.1201/9781315218168>
- [3] IEC 61508 Functional safety of electrical/electronic/programmable electronic safety-related systems
- [4] Schuster György, Ady László: Biztonságkritikus szoftver fejlesztés,
http://www.repulestudomany.hu/folyoirat/2018_1/2018-1-11-0453_Schuster_Gyorgy-Ady_Laszlo.pdf

SOFTWARE PLANNING

Software development is an extremely complex, and complicated process due to the extraordinary complexity of the products. Considering this testing cannot be complete fully at any level. The desired quality lies in the regulation of the development process and its strict monitoring. Planning is one of the most important steps in this process. Design is always based on a standard and a set of rules in all security-critical cases. It is the DO-178C standard for safety-critical software development that covers the entire software process. This paper deals with the key questions of software development and their main features.

Keywords: *software planning, safety critical*

Schuster György (PhD)
docens, főiskolai tanár
Óbudai Egyetem
Kandó Kálmán Villamosmérnöki Kar
Műszertechnikai és Automatizálási Intézet
schuster.gyorgy@kvk.uni-obuda.hu
orcid.org/0000-0002-8573-3670

György Schuster (PhD)
associate professor, college professor
Óbuda University
Kandó Kálmán Faculty of Electrical Engineering
Institute of Instrumentation and Automation
schuster.gyorgy@kvk.uni-obuda.hu
orcid.org/0000-0002-8573-3670



<http://journals.uni-nke.hu/index.php/reptudkoz/article/view/258/31>