

Softvér v automobilovom priemysle*

Peter Zimmermann

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
xzimmermann@stuba.sk

11. október 2021

Abstrakt

...

1 Úvod

Motivujte čitateľa a vysvetlite, o čom píšete. Úvod sa väčšinou nedelí na časti.

Uveďte explicitne štruktúru článku. Tu je nejaký príklad. Základný problém, ktorý bol naznačený v úvode, je podrobnejšie vysvetlený v časti ?? . Dôležité súvislosti sú uvedené v častiach ?? a ?? . Záverečné poznámky prináša časť 6.

2 Postupy vo vývoji automobilového softvéru

V tejto sekcii sa primárne zameriavam na tri hlavné postupy pri vývoji automobilového softvéru.

Vývoj automobilového softvéru a jeho samotné aplikovanie je v dnešnej dobe jeden z najhlavnejších procesov pri zhotovovaní automobilu. Je to proces, ktorý pozostáva z viacerých postupov:

1. Budovanie systému
2. Vývoj softvéru
3. Prevádzka

Prvý z postupov procesu je budovanie systému. V tomto postupe sa výrobcovia originálnych zariadení obvykle zaoberajú požiadavkami, ktoré sú primárne zamerané na funkčnosť zariadenia, ktoré bude najviac vyhovovať stanoveným požiadavkám (npr. riadiaca jednotka, hardvér palubného počítača). Títo výrobcovia sa vo väčšine prípadov riadia architektúrou automobilového open systému s názvom „AUTOSAR“. AUTOSAR je v dnešnej dobe celosvetové vývojové partnerstvo v oblasti softvéru a elektroniky v automobiloch.

Druhým z postupov je vývoj samotného softvéru, ktorého hlavným cieľom je vytvoriť viacero programovacích kódov, z ktorých sa vyberajú tie najvhodnejšie,

*Semestrálny projekt v predmete Metódy inžinierskej práce, ak. rok 2021/22, vedenie: Ing. Fedor Lehocký, Phd.

ktoré budú najlepšie pracovať s už zvolenými zariadeniami. Aby sa dokázali rozlíšiť rôzne úrovne zložitosti softvéru, možno použiť dva najbežnejšie modely a V-Model a Agile model. Posledným postupom je samotná prevádzka. Znamená to, že každý softvér je aj po vyrobení a uvedenia do predaju udržiavaný aktualizáciami, aby prevádzka a funkčnosť samotného softvéru bola čo najdlhšia. [1]

3 Modely pre vývoj automobilového softvéru

Táto sekcia slúži na priblíženie dvoch široko známych modelov vo vývoji softvérov pre automobily.

3.1 V-Model

V-Model je v dnešnej dobe najpoužívanější model vo automobilovom priemysle. Je rozdelený do troch častí. Prvou časťou je definovanie si krokov. Po tejto časti prichádza na rad samotné programovanie, po ktorom nasleduje overovanie predošlých krokov. Funguje to na tom princípe, že sa po fáze programovania otočí naspäť nahor na overovanie a výsledok každej fázy sa overí ešte predtým, ako sa prejde do nasledujúcej fázy V-Modelu.

3.2 Agile Model

Tento model z veľkej časti dáva do popredia evolučný rozvoj a rapídne zmeny. Vo viacerých projektoch sa softvéroví inžinieri rozhodnú pre tento spôsob z dôvodu, že vďaka nemu dokážu spracovať aj nepredpokladané zmeny zo strany zákazníckych požiadaviek. Medzi najväčšie výhody patrí to, že nemá pevné fázy, na rozdiel od V-Modelu. Z tohto dôvodu je Agile Model viac flexibilný, čo má za následok zvýšenie produktivity.

3.3 Porovnávanie V-Modelu a Agile Modelu

Obidva modely su známe najmä z toho dôvodu, že sú jednoduché. Agile Model je navrhnutý tak, aby bol čo najmenej zložitý vo vývojových procesoch. Ale z dôvodu, že softvér v automobiloch vyžaduje správnu spojitosť medzi softvérovou integritou, sa v praxi častejšie využíva V-Model. Okrem toho, Agile Model ma viacero nevýhod, ktoré sa vo vývoji softvéru pre automobily považujú za kľúčové, ako napríklad: slabá softvérová integrita, obmedzené znovu použitie kódu, nepriaznivé efekty na kvalitu práce a zložitá organizácia celej štruktúry.

Niekedy treba uviesť zoznam:

- jedna vec
- druhá vec
 - x
 - y

Ten istý zoznam, len číslovaný:

1. jedna vec

2. druhá vec

(a) x

(b) y

3.4 Ešte nejaké vysvetlenie

Veľmi dôležitá poznámka. Niekedy je potrebné nadpisom označiť odsek. Text pokračuje hneď za nadpisom.

4 Testovanie

4.1 Komplexné informácie k testovaniu softvéru

5 Porovnávanie dvoch popredných automobiliek z hľadiska softvéru

6 Záver

Literatúra

[1] James O. Coplien. *Multi-Paradigm Design for C++*. Addison-Wesley, 1999.