Agilné postupy pri vývoji medicínskeho softvéru*

Irina Makarova
Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
xmakarovai@stuba.sk

11. decembra 2021

Abstrakt

V súčasnosti sa rýchlo vyvíja trh zdravotníckych prenositeľných zariadení. Vodopádový model, ktorý sa klasicky používal pri vývoji bezpečnostne kritického softvéru, už pre svoju zdĺhavosť a nepružnosť nepredstavuje optimálny postup. Do popredia sa dostávajú rôzne agilné a kombinované prístupy, čo prináša so sebou otázku, ako implementovať do procesu vývoja agilné metódy a pri tom dodržať požadované normy. Tento článok prináša stručný prehľad aktuálne platných noriem; vysvetľuje podstatu vodopádového a agilných modelov (Scrum, XP, DSDM), porovnáva ich a poukazuje na črty jednotlivých stratégií, ktoré sú prínosné pre túto oblasť. V článku sa uvádzajú niektoré existujúce riešenia, ako sa tieto modely dajú efektívne kombinovať pri vývoji medicínskeho softvéru.

1 Úvod

S postupným vývojom internetu vecí sa do domácností dostáva čoraz viac zariadení so vstávaným softvérom. V súčasnosti vzrastá dopyt po prenositeľných zdravotníckych pomôckach slúžiacich na monitorovanie alebo korekciu zdravotných problémov. Vývoj softvéru, ktorý je súčasťou týchto zariadení, má prebiehať v súlade s platnými normami. Požiadavky uvedené v normách a smerniciach sú všeobecné a nemožno ich jednoducho nasadiť do praxe. Klasický sekvenčný prístup zdanlivo najlepšie pokrýva všetky rozmanité požiadavky. Dodnes sa väčšina firiem v tejto doméne riadi V-modelom vývoja softvéru [5]. Tento postup má však jednu zásadnú nevýhodu – netoleruje zmeny. V rýchlo sa meniacom svete technológií na každom stupni vývoja môže nastať potreba niečo upraviť s ohľadom na nové požiadavky. Okrem toho sa produkt musí rýchlo dostať k užívateľom, aby mal šancu etablovať sa na trhu. Agile si osvojili softvérové spoločnosti po celom svete. 84% respondentov pracujúcich v oblasti softvérového inžinierstva uviedlo, že ich tím využíva nejakú formu agile. Hlavnými príčinami popularity týchto metodík je ich efektívnosť, flexibilnosť a urýchlené dodanie funkčného produktu [3]. Ich miera adopcie v doméne vývoja bezpečnostné kritických systémov, akým je aj medicínsky softvér, je však stále nízka [5]. Jedným z dôvodov je, že

*Semestrálny projekt v predmete Metódy inžinierskej práce, ak. rok 2021/22, vedenie: Ing. Vladimír Mlynarovič PhD.

agilné postupy sa javia byť v rozpore s regulačnými požiadavkami. Výrobcovia softvéru pre medicínske zariadenia sú pre schválenie produktu povinní predkladať rozsiahlu dokumentáciu, avšak jeden z agilných princípov znie "Funkčný softvér je viac ako vyčerpávajúca dokumentácia". Aj keď samostatne žiadna z agilných metodík naozaj nie vhodná na vývoj zdravotníckeho softvéru, zavedením istých agilných praktík sa proces vývoja dá zoptimalizovať a lepšie prispôsobiť aktuálnemu stavu prostredia. Hlavným cieľom tohto článku je poukázať na užitočnosť agilných prístupov vo vývoji zdravotníckeho softvéru a vysvetliť na príklade rámca MDevSPICE a agilného V-modelu ako sa klasické a iteratívne modely dajú výhodne kombinovať tak, aby sa vzájomne dopĺňali.

2 Normy pre medicínsky softvér

Podľa nariadenia o zdravotníckych pomôckach 2017/745 (MDR) sa za zdravotnícku pomôcku považuje akýkoľvek výrobok, ktorý je výrobcom určený na použitie za účelmi ako sú diagnostika, predikcia, sledovanie, prevencia, liečba alebo zmiernenie choroby, a ktorý nedosahuje svoj účinok pomocou farmakologických alebo imunologických prostriedkov. Pod túto definíciu spadá nielen softvér zabudovaný v špecializovaných prístrojoch ale aj softvér ako taký, tzv. SaMD (Software as a Medical Device). Zdravotnícky softvér sa zaraďuje medzi bezpečnostne kritický softvér. To znamená, že jeho nesprávna funkčnosť môže ohroziť zdravie, alebo dokonca život človeka. V závislosti od rizík spojených s jeho používaním sa softvér klasifikuje na tri triedy. Z tejto klasifikácie potom vyplývajú spôsoby, akými sa preukazuje jeho bezpečnosť, a požiadavky, ktoré má spĺňať. V závislosti od regiónu, v ktorom sa má zdravotnícka pomôcka uviesť na trh, sa musia dodržiavať rôzne normy alebo usmernenia. V USA je to Úrad pre kontrolu potravín a liečiv (FDA). Vo všeobecnosti platí, že pri posudzovaní a schvaľovaní zdravotníckych pomôcok v Európskej únii neexistuje žiadny centralizovaný regulačný orgán, ako napríklad Európska agentúra pre lieky (EMA), alebo americký úrad FDA. V EÚ výrobca musí preukázať, že produkt spĺňa platné štandardy. Kľúčovým štandardom pre vývojárov medicínskeho softvéru je ISO IEC 62304:2006 [2]. Podľa neho musí vývoj softvéru prebiehať podľa vopred stanoveného modelu životného cyklu.

3 Modely riadenia vývoja softvéru

Všeobecne sa proces vývoja softvéru dá organizovať dvoma spôsobmi – sekvenčne a iteratívne. Pri tradičnom sekvenčnom vývoji sa najprv zostaví dokument všetkých požiadaviek na softvérový produkt. Na základe neho sa potom navrhne a implementuje produkt. Môže sa však stať, že požiadavky, ktoré zadá zákazník budú nepresné a neúplné, alebo sa pri testovaní v neskorších fázach objaví chyba. Vtedy sa bude treba vracať k predchádzajúcim etapám vývoja softvéru a aplikovať nové požiadavky na návrh, testovanie alebo implementáciu. V dôsledku častých alebo výrazných zmien požiadaviek je potom nutné zvýšiť rozpočet alebo oddialiť termín dodania. Toto zase môže viesť až k nepoužiteľnosti vytvorených softvérových systémov kvôli ich neaktuálnosti. Snaha nájsť alternatívu tomuto postupu vyústila k vzniku množstva techník a metodík, ktoré sa všeobecne volajú agile. Základné princípy agile boli sformulované skupinou 17 vývojárov v

roku 2001 a sú zadokumentované v Manifeste pre agilný vývoj softvéru [1], ktorý postuluje nasledujúce:

- Ľudia a komunikácia sú viac ako len procesy a nástroje
- Funkčný softvér je viac ako vyčerpávajúca dokumentácia
- Spolupráca so zákazníkom je viac ako dojednávanie zmluvy
- Radšej reagovať na zmenu ako sa držať plánu

Dodržiavať tieto hodnoty v praxi umožňuje inkrementálny spôsob vývoja a tesná spolupráca so zákazníkom. Agilné metodiky sú iteratívne – tvorba projektu prebieha v cykloch. Každý cyklus, alebo iterácia, sa dá chápať ako menší projekt, ktorého výstupom je vždy funkčná otestovaná verzia produktu. Výsledkom toho je, že sa k zákazníkovi už po prvých iteráciách dostane prevádzková verzia softvéru, ktorú vie vyskúšať a poskytnúť spätnú väzbu. Prípadné pripomienky zákazníka sa potom stávajú súčasťou nasledujúcej iterácie. Táto prispôsobivosť zmenám a flexibilita požiadaviek má významný dopad na kvalitu produktu a jeho prijatie zákazníkom.

V-model je v oblasti vývoja zdravotníckeho softvéru najpoužívanejším modelom. Je rozšírením vodopádového modelu. Dáva do súvislosti v ňom špecifikované vývojové aktivity s testovacími aktivitami. Pozostáva z dvoch vetiev, ktoré znázorňujú časovú postupnosť jednotlivých etáp vývoja a testovania.

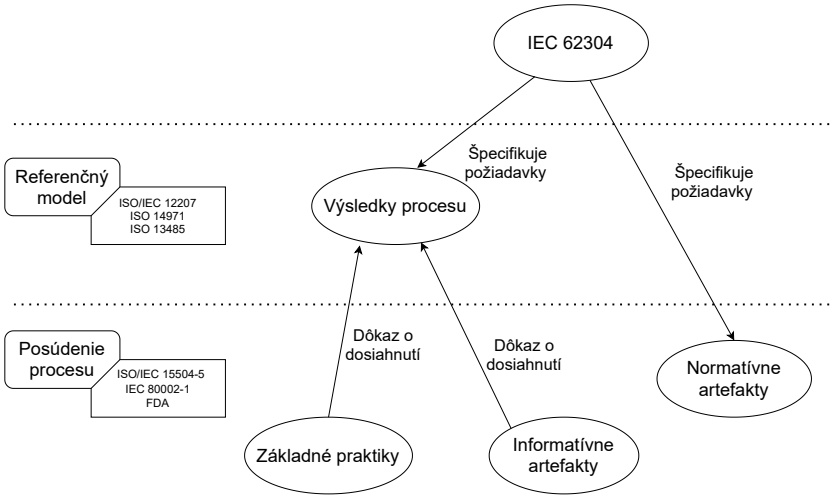
Scrum je predovšetkým spôsob manažmentu, menej sa zaoberá technickou stránkou softvérového inžinierstva. V súčasnosti je najrozšírenejšou agilnou metodikou [3]. Vývoj podľa scrum prebieha v rámci postupnosti pevne časovo ohraničených cyklov, tzv. sprintov. Po skončení každého z nich má tím byť schopný dodať potenciálne nasaditeľný produkt. Kontrola a prispôsobenie procesu vývoja sa vykonáva počas denných porád a taktiež na konci každého sprintu. Aktivita na projekte sa priebežne zaznamenáva v štyroch hlavných dokumentoch – produktový backlog (zoznam požiadaviek na produkt), sprint backlog (zoznam úloh pre daný sprint), release burndown (sledovanie ešte nerealizovaných položiek produktového backlogu), sprint burndown (sledovanie ešte nerealizovaných položiek v rámci prebiehajúceho sprintu). Tímy pracujúce podľa scrum sú samoorganizované a obsahujú tri roly - vlastník produktu, scrum master a vývojár. Prácu vývojárskeho tímu usmerňuje scrum master, ktorý slúži ako styčný bod medzi ním a zákazníkom [6].

Extrémne programovanie (XP) je v porovnaní s inými agilnými rámcami pomerne mladá metodika. Jej vznik sa datuje od roku 1999. XP zhrňuje praktiky softvérového inžinierstva tak, aby sa dalo vyvíjať čo najkvalitnejší produkt v udržateľnom čase. Najdôležitejšími z týchto praktík sú: jednoduchý návrh, refaktorizácia, priebežné testovanie funkčných blokov, programovanie v pároch, krátke iterácie, tesná prepojenosť so zákazníkom. Ideový základ XP tvorí päť hodnôt: *jednoduchost* – konať len na základe súčasných požiadaviek zákazníka, nesnažiť sa urobiť to, čo by zákazník mohol chcieť implementovať neskôr, pri návrhu sa riadiť otázkou 'ako vyzerá najjednoduchšie funkčné riešenie?'; *spätná väzba* – zákazník je aktívne zapojený do procesu vývoja, vývoj je riadený testami; *komunikácia a rešpekt* – metodika predpokladá priaznivé podmienky pre spoluprácu, členovia tímu by nemali pracovať samostatne podľa vlastných pravidiel, spoločné vlastnenie kódu, keďže sa na jeho písaní podieľa celý tím, párové programovanie; *odvaha*.

Metóda dynamického vývoja systémov presne definuje postupy a pravidlá vývoja softvéru. Je založená na deviatich princípoch, ktoré by mali byť dodržiavané počas jednotlivých fáz životného cyklu. Vývoj sa riadi tak, aby bol ukončený do stanoveného času a za stanovené prostriedky. DSDM sa menej zaoberá programovacimi prístumi, ale podporuje skôr riadenie projektov. Dá sa kombinovať s inými agilnými metodikami.

4 Kombinované prístupy

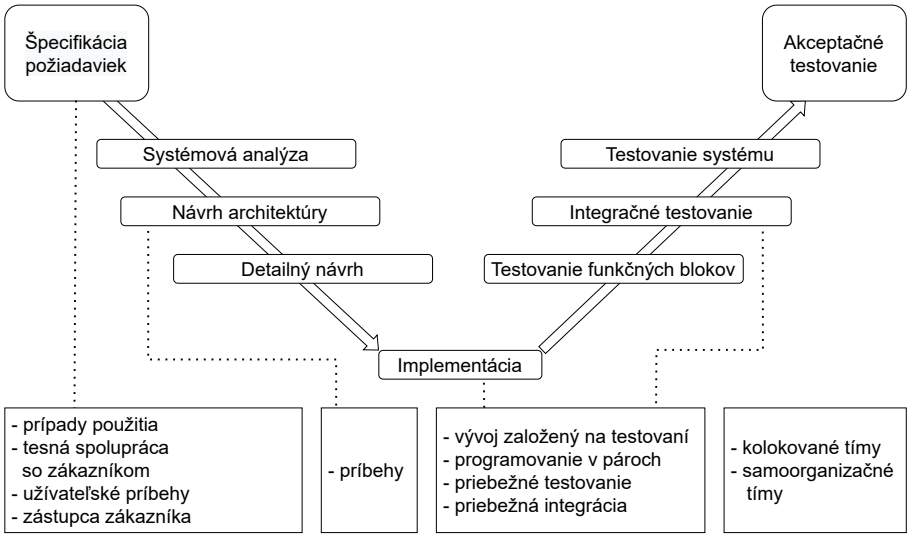
MDevSPICE je rámec postupov vývoja softvéru pre zdravotnícke pomôcky. Výzva, ktorej čelia spoločnosti vyvíjajúce zdravotnícky softvér, je dodržiavanie veľkého množstva regulačných požiadaviek uvedených v rôznych medzinárodných normách. Tento rámec bol vyvinutý s cieľom pomôcť firmám lepšie sa pripraviť na náročné a nákladné regulačné audity [7].



Obrázok 1. Základné prvky rámca MDevSPICE [4] - procesný referenčný model a model posudzovania efektívnosti.

Cieľom procesného referenčného modelu MDevSPICE® je poskytnúť komplexný model posúdenia softvéru a procesov vývoja systémov na základe všeobecne uznávaných predpisov o zdravotníckych pomôckach, noriem a smerníc, ktoré má firma zaoberajúca sa vývojom zdravotníckeho softvéru dodržiavať. Daný model, podobne ako ISO/IEC 15504-5 (SPICE), má dve časti - dimenziu procesu a dimenziu spôsobilosti.

Agilný V-model ako je vidieť z obrázku 2 rozširuje klasický V-model o agilné praktiky, je jeho nadstavbou. V-model bol autormi zvolený ako základ práve preto, že mnohé firmy ho už dobre poznajú a majú s ním dlhoročné skúsenosti. Na prechod do hybridného procesu vývoja je potrebná nielen znalosť agilných techník, ale aj istá miera odvahy. Tento model zmierňuje obidve prekážky. Jednak vymenúva konkrétne agilné praktiky, ktoré sú použiteľné a mapuje ich na jednotlivé fázy V-modelu. A na druhej strane využitie známeho V-modelu ako základu eliminuje strach z nového a neistotu tímov.



Obrázok 2. Agilný V-model podľa [5]. Plné čiary znázorňujú prechod medzi fázami vývoja. Prerušované čiary spájajú agilné metodiky s fázami, do ktorých môžu byť inkorporované. Dve praktiky nie sú spojené so žiadnou fázou, lebo sa týkajú organizácie tímov a nie procesu.

Tvorba tohto modelu prebiehala v spolupráci mnohými organizáciami. Daný model sa aj vyskúšal vo firme BBT, ktorá sa predtým riadila V-modelom a pri práci sa vo veľkej miere spoliehala na dopredné plánovanie. A preto zapracovanie zmien do vyvíjaného produktu bolo identifikované ako hlavný problém. V snahe zmieniť tento problém firma zaviedla inkubačnú dobu, ktorá predchádzala vývoju a dávala všetkým zainteresovaným stranám čas na odhalenie nepresností v požiadavkách. S ohľadom tieto skutočnosti bol autormi navrhnutý agilný V-model, ktorý prevzal niektoré praktiky z rámca Scrum a metodiky XP. Iteratívny prístup spočíval v tom, že sa výsledný softvérový produkt rozdelil na komponenty, a tie – na ešte menšie prvky, a jednotlivé prvky sa vytvárali v nezávislých cykloch. K integrovaniu komponentov dochádzalo v neskorších štádiách, pričom žiadna z požiadaviek sa nepovažovala za splnenú, kým každý komponent neprešiel fázu implementácie. Z rámca Scrum boli prebrané dokumenty umožňujúce sledovať stav projektu, ohraničenie času (time-boxing), technika odhadu Scrum poker, roly, denné porady, aktivity ohľadom plánovania šprintov. Z extrémneho programovania to boli refaktorizácia a vývoj riadený testovaním. Tento na mieru vytrovený AV-model sa následne vyskúšal v praxi na jednom projekte. AV-model zefektívnil proces vývoja a komunikácie so zákazníkom. Na záverečných pohovoroch členovia tímu urobili odhad, že projekt by sa predĺžil o 14% a rozpočet prekročil o 7%, keby postupovali podľa V-modelu.

5 Záver

Literatúra

[1] Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmit, Andrew Hunt, Ron

Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, and Dave Thomas. Manifesto for agile software development. `https://agilemanifesto.org/iso/sk/manifesto.html`. Accessed: 2021-11-05.

[2] Jan Benedikt Brönneke, Jennifer Müller, Konstantinos Mouratis, Julia Hagen, and Ariel Dora Stern. Regulatory, legal, and market aspects of smart wearables for cardiac monitoring. *Sensors*, 21(14):4937, 2021.

[3] Richard Knaster. 15th state of agile report. `https://digital.ai/resources/state-of-agile/`. Accessed: 2021-11-05.

[4] Marion Lepmets, Fergal McCaffery, and Paul Clarke. Development and benefits of mdevspice®, the medical device software process assessment framework, 2016.

[5] Martin McHugh, Oisin Cawley, Fergal McCaffcry, Ita Richardson, and Xiaofeng Wang. An agile v-model for medical device software development to overcome the challenges with plan-driven software development lifecycles. *2013 5th International Workshop on Software Engineering in Health Care (SEHC)*, 2013.

[6] Ken Schwaber and Jeff Sutherland. The scrum guide. `https://scrumguides.org/`. Accessed: 2021-11-02.

[7] Özcan Top, Özden, Mccaffery, and Fergal. To what extent the medical device software regulations can be achieved with agile software development methods? xp—dsdm—scrum. *The Journal of Supercomputing*, 75(8):5227–5260, 2019.