Metodiky pro vývoj počítačových her

Semestrální práce k předmětu 4IT421 - Zlepšování procesů budování IS

Obsah

Jak se počítačové hry liší od jiných informatických projektů?

Počítačové hry nemají jednoho konkrétního zákazníka. I v případě, kdy na vývoj dohlíží mateřská firma, investor nebo jiný stakeholder, hra se nakonec bude stejně prodávat mase zákazníků, kteří nemohou formulovat svoje požadavky, dohlížet na vývoj a požadovat změny. Navíc je to vysoce kreativní proces, kdy má mnoho lidí svoje názory, jak by výsledný produkt měl vypadat a co by měl obsahovat a zároveň existují všeobecně uznávané poučky a vědecké poznatky, jak to dělat správně.

Životní cyklus

Před masovým rozšířením Internetu se životní cyklus hry do značné míry podobal životním cyklům jiných aplikací. Tento klasický model začal designem, vyvinula se hra, ta se pustila na trh a tím vývoj skončil. Bylo jen málo možností jak po vypuštění hru opravit nebo doplnit, a tak se musela hra vydat už v hotovém stavu. Zejména hry na herní konzole bylo až do poslední generace konzolí nemožné po vydání opravovat a musela být finálnímu testování věnována velká péče. Konzole mají navíc tu specialitu, že jejich majitelé očekávají, že bude hra fungovat tak, jak si ji koupili a nebudou se muset věnovat konfiguraci ani nebudou muset čekat na patche, a tak mnoho firem stále pokračuje v klasickém modelu vývoje her.

Dnes už ale je možné dodávat patche koncovým zákazníkům a mnoho firem přišlo na to, že hru nemusí vydávat v hotovém stavu a začaly toho využívat. Jsou firmy, které jsou zvyklé vydávat nehotové hry, např. aby stihly atraktivní datum vydání (např. před Vánoci) a postupně vydávají patche nebo čekají, že hru opraví komunita fanoušků. Na této strategii není samo o sobě nic špatného, ačkoliv poněkud penalizuje zákazníky, kteří si hru koupili krátce po vypuštění na trh. Firma tak může získat mnohem víc údajů o chybách, než z klasického interního testování a může do hry přidávat obsah na základě požadavků zákazníků. Pokud ale firma opravy neprovádí, může nenávratně poškodit své dobré jméno. Navíc mnoho distributorů přikládá největší váhu prodejům za poměrně krátké období po vypuštění (např. dva týdny až měsíc). Pokud firma hru vydá s chybami, kvůli kterým zákazníci odloží koupi až na dobu, kdy budou opraveny, může distributor hru ohodnotit jako neúspěšnou i přes to, že po opravení chyb se o ní prudce zvedne zájem.

Příklad: Bethesda Softworks

Bethesda Softworks je firma známá především díky sérii her The Elder Scrolls a nedávného oživení série Fallout. Hry ze série The Elder Scrolls jsou zasazeny ve fantasy světě plném magie, bájných zvířat a hrdinů, zatímco hry ze série Fallout se odehrávají v postapokalyptickém světě, kde je moderní civilizace zničená jadernou válkou. V podstatě se ale jedná o stejné hry – velký otevřený svět plný lokací a úkolů, kde se hráč může soustředit na hlavní příběhovou linii, nebo plnit vedlejší mise a aktivity. Od roku 2002 vydala Bethesda 5 v podstatě stejných her (pokud nepočítáme různé přídavky), které v době vydání obsahovaly mnoho závažných chyb, ale po vydání následovalo vždy několik patchů, které většinu z nich opravily. Navíc k těmto hrám vydali nástroje, díky kterým mohla komunita fanoušků vytvářet nejrůznější přídavky a modifikace, kterými si každý hráč může přizpůsobit hru tak, jak se mu zlíbí. Ke hře *The Elder Scrolls III: Morrowind*, vydané v roce 2002, vycházejí ještě dnes významné modifikace a vylepšení vytvořené fanoušky, které stále mohou být důvodem hru si koupit.

Také jsou firmy, které tuto strategii berou ještě dál a otevřeně přiznávají, že hra, kterou vydávají, není hotová. Opravují a dokončují ji v době, kdy už je na trhu a zákazníky, kteří si ji koupili v nehotovém stavu, často nějak odměňují. Takto fungují například „sociální hry“ na síti Facebook nebo mnoho nezávislých her distribuovaných pomocí Internetu. Hra se vydá v okamžiku, kdy obsahuje základní funkcionalitu a už v té chvíli začne generovat příjmy. Postupně se opravují chyby, přidává další obsah a mění stávající vlastnosti tak, aby byli zákazníci spokojení. Zde ale hrozí riziko, že si takové hry všimne někdo s více prostředky, vybere z ní nápady a vydá vlastní hru, v horším případě dokonce kopii, která je lepší než původní hra, které tím sebere zákazníky.

Příklad: Minecraft

Minecraft je hra švédského vývojáře Markuse Perssona. Hráč se objeví ve světě náhodně generovaném z kostek 1m x 1m x 1m, které může dle libosti přesouvat a stavět z nich. Hra donedávna neměla žádný daný cíl, je na hráčích, jestli budou stavět, prozkoumávat, hromadit materiál. Markus Persson začal prodávat alfa verzi, která se od té současné dost lišila. Zájem o hru byl ale dost velký na to, aby se mohl začít věnovat vývoji na plný úvazek. Kolem hry se také utvořila komunita fanoušků, kteří hlásili bugy a vymýšleli různé modifikace, z nichž pak některé Markus Persson implementoval do jádra hry. Hra Minecraft prodala 4 milióny kopií ještě před tím, než byla oficiálně vydána plná verze. (Downin, 2011)

Zodpovědnost

Co se týče business aplikací, často jsou pro firmu, která si je objednává, kritické a může na nich záviset osud celé firmy. To u her samozřejmě neplatí – ačkoliv jsou organizace, které závisí na nějaké konkrétní hře (např. ligy elektronických sportů, tvůrci machinim[1]), většinou si vybírají už existující hru,

[1] Machinima – video vytvořené v enginu hry, např série Red vs. Blue

ke které jsou dostupné všechny prostředky nutné pro jejich business a s vývojářem spolupracují jen vzácně. Velcí vývojáři a distributoři navíc často práva zákazníků významným způsobem omezují, ale zákazníci se proti tomu neumí, nemohou nebo nechtějí bránit. Proto jediné, co je v sázce při vývoji her, jsou peníze vložené do vývoje a dobré jméno vývojáře, popř. distributora.

Příklad: Hacknutí a únik dat z PlayStation Network

PlayStation Network je internetová služba, která majitelům konzole PS3 umožňuje hrát hry online, získávat nejrůznější obsah od her a doplňků k nim přes filmy až k hudbě. V dubnu 2011 došlo k velkému výpadku PSN a následnému úniku nezašifrovaných dat, mj. údajů o kreditních kartách zákazníků PSN. Tito zákazníci se spojili v několika hromadných žalobách na Sony. Reakce Sony? Změnit podmínky používání PSN tak, že se uživatelé automaticky vzdávají práva na hromadnou žalobu. (Vilches, 2011)

Jak sbírat požadavky?

Vývoj her je kreativní proces, kdy má každý, kdo je do vývoje zapojený (a spousta těch, kteří do vývoje zapojeni nejsou), spoustu nápadů, které chce prosadit. I kdyby byly všechny skvělé, není praktické snažit se je do hry dostat všechny. Proto by Product Owner měl být člověk, který má jasnou vizi, jak má výsledný produkt vypadat. Zároveň by však měl být schopný poznat, jakou má která vlastnost prioritu. Pokud má vyvíjená hra nějakého investora, měl by být dost silný na to, aby mu dokázal rozmluvit vlastnost, kterou chce investor do hry za každou cenu dostat. Ano, vím, že jsem právě popsal dokonalého Product Ownera.

Podívejme se blíže, kde by se tedy požadavky měly brát. V ideálním případě je Product Owner člověk, který má skvělý nápad na hru a má znalosti a prostředky, jak ho uskutečnit. Od něj by měla pocházet většina vlastností kritických pro funkčnost hry. Měl by také přijímat nápady od všech ostatních členů týmu a zvažovat jejich přidávání. Product Owner ale nemůže izolovat Product Backlog od všech lidí nepodílejících se přímo na vývoji.

Pokud má hra nějakého investora, ten se bude téměř vždy snažit mluvit do vývoje – bude chtít přidat vlastnosti z jiné hry, která se díky nim stala nečekaným hitem, vydat hru na jinou platformu, než vývojář zamýšlí anebo změnit tón hry tak, aby se vešla do mírnější přístupnostní kategorie a mohla se tak prodávat i mladším zákazníkům. Ačkoliv se to může z pohledu investora zdát jako logický krok, pokud se tak stane v průběhu vývoje, na výsledku to bývá vidět a málokdy to přispěje k jeho kvalitě.

> Příklad: Haze
>
> Haze je střílečka z pohledu první osoby vydaná v roce 2008 na PlayStation 3 firmou Ubisoft. Hra měla původně vyjít dříve, i na ostatní platformy, měla mít podstatně jinou zápletku (kterou nechal Ubisoft několikrát celkově přepsat) a Ubisoft nakonec vydal zjevně nedokončenou verzi, aby na trhu nekolidovala s jinou, podobně zaměřenou hrou, kterou plánoval Ubisoft vydat o něco později. Haze si na trhu vedla mizerně a vyústila v bankrot vývojářů. (Gaultier, 2007)

Samozřejmě byste také měli zjistit, co přesně chce váš cílový zákazník. Pokud zvolíte vývojový model, kdy vypustíte hru na veřejnost a zároveň pokračujete ve vývoji, máte skvělou možnost. Rozhodně byste měli vytvořit oficiální fórum, kde se hráči můžou o hře bavit a pravidelně ho sledovat. Také můžete do hry zabudovat nástroje, které vám budou dávat informace o tom, jak hráči hrají, kterými činnostmi tráví nejvíce času, v jaký moment hru vypínají apod. Tuto aktivitu je ale potřeba ošetřit tak, aby se nejednalo o narušování soukromí dle platných zákonů.

Pokud veřejnost během vývoje nemá vaši hru k dispozici, měli byste navštěvovat fóra jiných, podobných her a tam zjišťovat, co si hráči přejí, jaké modifikace vyrábějí a nejčastěji stahují atd. Pokud vyrábíte hru, která používá nějakou existující značku (např. remake, pokračování, hru podle filmu atd.), měli byste zjistit, jaké jsou hlavní prvky té značky a také se rozhodnout, zda se zaměříte spíše na existující fanoušky oné značky nebo na nové zákazníky. Dle mého názoru nemůžete uspokojit obě skupiny najednou, protože pokud se zaměříte na stávající fanoušky, noví zákazníci budou zmateni, v opačném případě bude obsah hry připadat starým fanouškům triviální a pokud se pokusíte

najít zlatý střed, výsledkem bude nemastný-neslaný mix, se kterým nebude spokojena ani jedna skupina. Samozřejmě je zde otázka, zda má cenu využívat (a být omezen) stávající značkou, pokud se nechcete zaměřit na její fanoušky.

Příklad: Team Fortress 2

Team Fortress původně začal jako modifikace hry střílečky Quake. Hra spočívá v souboji dvou týmů, kde si hráči mohou vybrat, za jakého vojáka (např. špión, plamenometčík, inženýr, …) budou hrát. Tato modifikace se snažila vypadat realisticky a byla natolik úspěšná, že se ji firma Valve rozhodla vydat její pokračování jako samostatnou hru. Toto pokračování se ale v mnoha ohledech lišilo – z hratelnostního hlediska byly přepracovány třídy vojáků, zmizely granáty atd., umělecký styl byl pak změněn k nepoznání. Omaskáčované vojáky nahradily charakterizované postavy ve víceméně civilním oblečení (např. špión je štíhlý Francouz v obleku a kukle, odstřelovač je stereotypní Australan, demoliční expert je jednooký černý Skot atd.) a celkový styl se ze zobrazení realistického boje změnil spíše na kreslenou karikaturu ve stylu Normana Rockwella.

Ačkoliv skalní fanoušci původních modifikací TF2 odsoudili, hra měla úspěch u kritiků i u hráčů a je jednou z nejhranějších online stříleček i čtyři roky po svém vydání.

Velká část požadavků by měla pocházet od testerů, mělo by se však jednat spíše o požadavky na změny stávajících vlastností, než na úplně nové vlastnosti. Jsou to právě oni, kdo mohou kromě klasických bugů odchytit i velkou část chyb v designu hry. Při vývoji her je potřeba věnovat jim zvláštní pozornost. O tomto problému napíšu více v jedné z dalších kapitol.

V ideálním případě by Product Owner dokázal sepsat všechny požadavky sám. V tom skutečném by měl být schopen posoudit, jak která vlastnost ovlivní výslednou hru a být schopen zdůvodnit její přijetí nebo odmítnutí čistě na základě jejího přínosu k hratelnosti.

Příklad: Magicka

Magicka je hra švédských vývojářů Arrowhead Game Studios z roku 2011. V jejím jádře stojí unikátní systém kouzlení pomocí kombinací mystických prvků. V raném stádiu vývoje ale při testování přišli na to, že tento systém byl ve své tehdejší podobě pomalý, složitý a neobratný. Ačkoliv tak byl původně zamýšlený, přišli na to, že hra tak není zábavná. Současný systém je rychlejší a jednodušší a významně se podílí na úspěchu hry.

Při vývoji jim také někteří zavedení vývojáři dávali rady, které používala většina současných her a které se obecně považují za „best practice“ – nedávat hráči všechny mystické prvky hned na začátku ale postupně až v průběhu hry, při kooperativní hře více hráčů jim znemožnit navzájem se zraňovat nebo mít klávesové zkratky pro oblíbená kouzla. Tyto rady šly naprosto proti duchu hry, kterou se Arrowhead snažili vytvořit a proto je neposlechli.

Modifikace metodik

Z výše zmíněných důvodů se pro vývoj většiny her hodí metodika Scrum. Ta je dobře popsaná v mnoha publikacích a nebudu se jí tedy zde příliš zabývat. Spíše se zaměřím na kroky, které tato metodika nepopisuje, ale kterých by se měl každý vývojář her držet. Nejdříve ale napíšu, pro které situace a projekty není dle mého názoru Scrum vhodný.

- Velké, vysokorozpočtové hry. Jsou hry, které měly při vývoji až 100 programátorů. To už je číslo, které Scrumem nejde uřídit. Je možné, že by šla použít nějaká metodika pro kooperaci více Scrumů, u takových projektů se ale spíše využívají rigidní metodiky. Ani ty ale kvůli výše zmíněným zvláštnostem vývoje her nejsou příliš vhodné, nastává tak otázka, zda vůbec existuje nějaká metodika vhodná pro takovéto hry.
- Hry na zakázku. Vývojářské týmy často vyvíjejí hry pro klienty, kteří hru nehodlají prodávat koncovým zákazníkům. Jedná se například o hry, které mají propagovat nějakou existující značku na stránkách klienta. Tito klienti mají často velmi přísné požadavky, jak má taková hra vypadat a kreativitě nedávají mnoho prostoru.
- Týmy, kde není vhodná osobnost pro roli Project Ownera nebo Scrum Mastera. To se stává zejména u malých týmů, které fungují na demokratické bázi nebo u týmů, které nemají zkušenosti s metodikou Scrum. Takové týmy často tuto metodiku implementují špatně, a proto jim nepřinese očekávané výsledky.

Na co by se tedy vývojáři měli zaměřit?

Pro koho vyvíjíte?

Ačkoliv agilní metodiky nekladou velkou váhu na úvodní plánování projektů, určitá míra plánování je nutná. Co nejdříve ve vývoji se musíte rozhodnout, pro jakou platformu budete vyvíjet, jaký monetizační model zvolíte, kdo jsou vaši cíloví zákazníci atd. Samozřejmě se v průběhu vývoje můžete rozhodnout vydat se jiným směrem, ale čím později takové rozhodnutí uděláte, tím to bude dražší, při převodu vznikne více bugů, přijdete o účinek dosavadní propagace apod.

Hratelná verze co nejdříve

Podle metodiky Scrum má na konci každého Sprintu vzniknout funkční produkt. To ale neznamená funkční hra – mohou to být různé samostatné části jako například návrh UI nebo logika konkrétního herního mechanismu. Vytvořit první funkční verzi hry pravděpodobně potrvá déle, ale je potřeba se na to zaměřit. To totiž umožní lepší testování, bude lépe vidět rozsah projektu a v neposlední řadě umožní týmu vydat hru před dokončením, kdyby k tomu byl donucen. Také není potřeba u každé vlastnosti dbát na co nejkvalitnější naprogramování a dokumentaci – alespoň hned napoprvé. Pokud totiž testování ukáže, že o tuto vlastnost nemají hráči zájem, zbytečně byste ztráceli čas.

Kvalitní nástroje

To neznamená jen IDE, grafické programy a další nástroje třetích stran. Během vývoje hry budete potřebovat mnoho nástrojů, které budou specifické pro vývoj konkrétní hry a které si budete muset sami vytvořit. Cílem těchto nástrojů je zjednodušit a zrychlit vývoj hry a jejího obsahu, obzvlášť v pozdních stádiích vývoje hry.

Jsou to např. následující nástroje:

- Build pipeline. Toto jsou nástroje, které celkově usnadňují vývoj. Patří sem různé exportéry a importéry, nástroje na propojení různých druhů obsahu (např. aktérů se zvuky) nebo nástroj na výrobu lokálního buildu, který umožní zapínat a vypínat různé vlastnosti hry.
- Content creation. Tyto nástroje by měly umožnit i lidem, kteří neznají specifický kód vaší hry nebo programování obecně, vytvářet pro vaši hru obsah jako jsou různé úrovně, aktéři, zvuky, scénář, herní mechaniky atd. Měly by být dostatečně uživatelsky příjemné a obsahovat validaci vytvářeného obsahu.
- Debugovací nástroje. Tyto by měly usnadnit testování a debugování hry. Tyto nástroje by měly umožnit například v reálném čase zobrazovat data ze hry, rychle upravovat různé proměnné nebo simulovat velké množství různých událostí. Také se hodí nástroj, který umožní rychle a pohodlně měnit různá drobná nastavení, aby se při testování mohly rychle zkoušet různé varianty.
- Lokalizační nástroje. Pokud se rozhodnete vydat hru ve více jazycích, nestačí pouze přeložit prostý text v datových souborech. Jazyky mezi sebou mají velké rozdíly, např. průměrnou délku slova, takže pouhý překlad může rozhodit UI, vypsat část textu mimo obraz a způsobit další problémy.

U všech těchto nástrojů je také důležité, aby měly kvalitní dokumentaci a jednotné UI. Pokud budete potřebovat vytvořit nástroj vyžadující znalosti z oblastí, které nikdo z vašeho týmu nemá, je lepší zaplatit si specialistu, než se snažit učit za pochodu. Je potřeba si uvědomit, že s těmito nástroji budou často pracovat lidé, kteří nejsou zběhlí v programování a pro ty je často špatný nástroj horší než žádný nástroj.

Teambuilding

Při vývoji her potřebujete mít lidi s celou řadou schopností – 2D a 3D grafiky a animátory, zvukaře, hudebníky, dabéry, scénáristy atd. Je potřeba, aby všichni tito lidé nevytvářeli izolované týmy, ale aby spolupracovali a navzájem si předávali znalosti. Skvělý scénář, který kvůli omezení enginu hry nejde vytvořit, je k ničemu.

Získání faktů

Ačkoliv se design her podobá spíše umění než exaktní vědě, vztahují se na něj poznatky z jiných oborů a exaktních věd. Například současné 3D hry se potýkají s konceptem „tajemného valu“, kdy lidem připadá odpuzující téměř úplně, avšak ne zcela lidská postava. Tvůrci takzvaných „sociálních her“ zase využívají znalosti neuropsychologie tak, aby dosáhli co nejvyššího zájmu a konverze hráčů a ve výsledku učinili své hráče závislými (Luscombe, 2009).

Pečlivé plánování releasů

Během vývoje budete pravděpodobně potřebovat vydat několik různých verzí pro různé účely – prototyp hry pro potenciální investory, předváděcí verzi pro různé výstavy a jiné veřejné události, demoverzi pro potenciální zákazníky apod. Tyto releasy je potřeba plánovat s dostatečným předstihem nebo mít možnost kdykoliv takový release sestavit z již existujících vlastností a obsahu.

Jde o to, abyste na výrobě těchto verzí nestrávili příliš času a zároveň mohli to, co vytvoříte speciálně pro ně, použít v dalším vývoji.

Také je potřeba vybrat datum vydání – například zda vydat hru před vánoci, kdy prodeje značně rostou, ale také je mnohem vyšší konkurence, nebo v létě, kdy tolik her nevychází. To je spíše otázka pro marketing, ale je potřeba, aby měl tým dost času na vývoj a aby se toto datum příliš neposouvalo. Co se týče her, které jsou ve vývoji a zároveň k dispozici pro veřejnost (např. hry na Facebooku), je dobrý nápad nevydávat nové verze v pátek. Pokud se totiž po vydání objeví chyba, ztratíte celý víkend, protože na ni nemůžete dostatečně rychle reagovat.

Vydání v každém případě

Může to znít samozřejmě, ale hru je potřeba vydat. Je ale mnoho projektů, které vyšuměly do ztracena proto, že se tým zasekl na nepodstatných detailech a ztratil zájem, došly mu peníze a podobně. Jen vydáním hry se můžete poučit, co jste udělali dobře a co ne, na čem záleží a na čem ne atd. Hra je hotová v okamžiku, kdy ji dokáže dohrát osoba, která ji vidí poprvé.

Testování

Hry, stejně jako jakýkoliv jiný software je potřeba testovat, aby byla zajištěna jejich kvalita. To se týká nejen kvality kódu, ale i jednotlivých vlastností a obsahu. Takové testování může odhalit nejen chyby, ale i nové vlastnosti, které mohou výrazně přispět k výsledné kvalitě hry nebo dokonce navždy změnit celý herní žánr.

> Příklad: Komba v bojových hrách
>
> V bojových hrách je kombo technika, kdy můžete protivníka správnou sekvencí útoků několikrát po sobě zasáhnout, aniž by měl možnost reagovat. Dnes tato vlastnost nesmí v žádné hře tohoto žánru chybět, poprvé se ale objevila jako bug ve hře Street Fighter 2. Testeři ho objevili a nahlásili, nebyl ale shledán natolik závažným, aby byl vývojáři opraven a tak ho ve hře nechali. Hráči ho ale objevili, zdokumentovali a začali masově využívat, takže se v dalších hrách objevil už jako regulérní vlastnost, kterou hra sleduje a uděluje za správné provedení odměny.

Testování her zahrnuje jak klasické jednotkové testování při vývoji, tak playtesting, testování použitelnosti a také kontrolu výstupů členů týmu, kteří nejsou programátory (scénáristé, animátoři atd.). O každém z těchto způsobů testování je napsáno nespočet publikací, proto se zde budu věnovat hlavně playtestingu a u ostatních jen zmíním základy.

U klasických modelů vývoje her, kdy je první funkční verze hotová až téměř ke konci vývoje, slouží testování téměř výhradně k nalezení a odstranění chyb, protože už je příliš pozdě na to dělat významnější změny. Často probíhá tak, že se najme velké množství testerů, krátce se vyškolí a pustí se do hledání chyb. Tato praktika má několik zásadních vad.

- Potřebujete najmout velkou pracovní sílu na krátkou dobu. I kdybyste totiž ihned po vydání začali pracovat na další hře, testovatelnou verzi budete mít zase až ke konci projektu.
- V pozdní části projektu se dá očekávat, že projekt má už nějaké zpoždění a tak mnohá studia zavádějí praktiku zvanou „crunch-time“, kdy po testerech i po vývojářích vyžadují velké množství neplacených přesčasů.
- Často odhalíte závažné chyby až v době, kdy s nimi už nemůžete kvůli nedostatku času nic dělat. Stejně tak nemůžete použít poznatky z testování k vytvoření nových vlastností nebo změně stávajících.

Pokud používáte agilní metodiky, měli byste mít první hratelnou verzi hry již v rané části vývoje (ideálně hned po prvním sprintu) a přesně od té chvíle byste ji také měli začít testovat. Tím se vyhnete předchozím nešvárům a ve výsledku budete mít lepší produkt.

Testovat hru by měli všichni členové vývojového týmu, samozřejmě byste také měli mít testery a pak také hráče, kteří vaši hru uvidí poprvé. Profesionálové, kteří vaši hru hrají osm hodin denně, často ztratí perspektivu, nedokážou ohodnotit obtížnost a často minou bugy, které prvohráč objeví. Bohužel nemůžete čekat, že vám napíší kvalitní report a tak je musíte bedlivě pozorovat. U takovýchto testů by měli být přítomni vývojáři, také by se sezení mělo pořádně dokumentovat – hra by měla ukládat log činností, vstupů a výstupů, optimálně byste ještě takového prvohráče měli nahrávat na video – jaká tlačítka mačká, kam se na obrazovce dívá atd. Rozhodně byste jim neměli jakkoliv pomáhat nebo radit – pokud se někde zaseknou, ukazuje to na chybu designu. Měli byste je přimět, aby říkali, co si myslí, a to nejen při rozhovoru po hraní, ale i při hraní samotném. Měli by vám říkat, kdy se baví, kdy se nudí, kde se zasekli, ale, jak jsem zmiňoval dříve, nenapovídejte jim. Po hraní byste si s nimi měli promluvit, probrat, co si o hře myslí a sepsat zprávu. Podle té pak vložíte požadavky do Product Backlogu.

Pokud hru vydáváte pro veřejnost ještě za vývoje, máte možnost získávat informace i od vašich zákazníků. Stačí, když do hry zabudujete nástroje, které budou odesílat informace o hraní. Ačkoliv tak nezískáte tolik detailních informací, jako při monitorovaném hraní, budete mít velký vzorek, pomocí kterého lépe určíte „průměrného hráče“ vaší hry.

Citovaná literatura

Downin, J. (7. 11 2011). *Minecraft beta cracks 4 Million.* Získáno 22. 12 2011, z GameSpot: http://www.gamespot.com/news/minecraft-beta-cracks-4-million-6344310

Gaultier, P. (2. 11 2007). *Spinning The Moral Compass.* Získáno 22. 12 2011, z Gamasutra: http://www.gamasutra.com/view/feature/2003/spinning_the_moral_compass_.php

Luscombe, B. (30. 11 2009). *The Troubling Rise of Facebook's Top Game Company.* Získáno 11. 12 2011, z Time Magazine US: http://www.time.com/time/magazine/article/0,9171,1940668,00.html

Vilches, J. (16. 9 2011). *Sony changes PSN terms to block class action lawsuits.* Získáno 22. 12 2011, z TechSpot News: http://www.techspot.com/news/45505-sony-changes-psn-terms-to-block-class-action-lawsuits-.html