FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání diplomové práce

Název:	Rogue planet: 2D strategická hra
Student:	Bc. Ladislav Strnad
Vedoucí:	Ing. Radek Richtr, Ph.D.
Studijní program:	Informatika
Obor / specializace:	Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	do konce letního semestru 2023/2024

Pokyny pro vypracování

Cílem práce je finální realizace a vydání hry Rogue planet. Tato práce navazuje na bakalářskou práci, které se věnoval stejný autor.

1) Proveďte důkladnou analýzu současného stavu herního prototypu realizovaném v bakalářské práci.
2) Analyzujte možné postupy a cesty, jak dopracovat a oficiálně vydat danou hru pro budoucí hráče.
3) Na základě analýz navrhněte vhodné kroky včetně SI postupů a řešení k dosažení finálního vydání použitelné a plně hratelné hry.
4) Zaměřte se na výsledný softwarový produkt a jeho použitelnost.
5) Implementujte finální herní produkt.
6) Proveďte důkladné testování. Zvolte vhodné sady testů.
7) Pokuste se hru vydat pomocí alespoň jednoho vámi zvoleného game store.

Elektronicky schválil/a Ing. Michal Valenta, Ph.D. dne 29. prosince 2022 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Rogue planet: 2D strategická hra

Bc. Ladislav Strnad

Katedra softwarového inženýrství
Vedoucí práce: Ing. Radek Richtr, Ph.D.

4. května 2023

Poděkování

Děkuji Ing. Radku Richtrovi, Ph.D. za vedení této práce a věcné připomínky. Dále děkuji Bc. Davidu Primusovi za pomoc s testováním práce v rané fázi vývoje a Tomáši Krejčíkovi za důkladné otestování hry v závěru vývoje. Děkuji též Ing. Lence Strnadové za pomoc s korekturou práce a Bc. Kristýně Víškové za morální i pravopisnou podporu. Na závěr pak děkuji také všem hráčům, kteří se zúčastnili uživatelského testování a dali si práci s poskytnutím zpětné vazby.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené.

V Praze dne 4. května 2023 .....

Odkaz na tuto práci

Strnad, Ladislav. *Rogue planet: 2D strategická hra.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2023.

Abstrakt

Tato magisterská práce se zabývá finální realizací a vydáním 2D strategické hry Rogue Planet vyvíjené v Unity, jejíž základy byly položeny v bakalářské práci stejného autora. S použitím metod a nástrojů softwarového inženýrství představených v rešerši je provedena analýza stavu již existujícího prototypu hry, jsou identifikovány a zhodnoceny možné cesty dalšího vývoje hry a následně je vytvořen softwarový a herní návrh její nové podoby. Dle vytvořeného návrhu je implementována nová verze hry, která je následně otestována kvantitativním uživatelským testováním. Hra je na základě analýzy výsledků testování vhodně upravena a následně vydána skrze zvolenou digitální distribuční platformu.

Klíčová slova strategická hra, 2D hra, Unity, uživatelské rozhraní, pixel art

Abstract

This master's thesis deals with the final realization and release of the 2D strategy game Rogue Planet developed in Unity, the foundations of which were laid in the bachelor's thesis of the same author. Using the methods and tools of software engineering presented in the research, an analysis of the state of the already existing prototype of the game is performed, possible ways of further development of the game are identified and evaluated, and then a software and game design of its new form is created. According to the created design, a new version of the game is implemented, which is subsequently tested by quantitative user testing. Based on the analysis of the test results, the game is appropriately adjusted and subsequently released through the chosen digital distribution platform.

Keywords strategy game, 2D game, Unity, user interface, pixel art

Obsah

Úvod **1**

1 Cíl práce **3**

2 Rešerše **5**
2.1 Metody návrhu uživatelského rozhraní 5
2.1.1 Persony 5
2.1.2 Případy užití 6
2.2 Metody testování uživatelského rozhraní 7
2.2.1 Heuristická evaluace 7
2.2.2 Kognitivní průchod 9
2.2.3 Testování použitelnosti 9
2.3 Principy softwarového inženýrství 10
2.3.1 Principy SOLID 10
2.3.2 DRY – Don't repeat yourself 11
2.3.3 KISS – Keep it simple stupid 11
2.3.4 SoC – Separation of concerns 11
2.4 Refaktorování 11
2.5 Návrhové vzory 12
2.5.1 Návrhové vzory implementované v Unity 13
2.5.2 Abstract Factory 14
2.5.3 Object Pool 14
2.5.4 Singleton 15
2.5.5 Service Locator 15
2.5.6 Command 15
2.5.7 State 16
2.5.8 Observer 16
2.5.9 Model View Controller a Model View Presenter 16
2.5.10 Data Locality 17

2.5.11 Entity Component System 17

3 Analýza **19**
3.1 Současný stav Rogue Planet 19
3.1.1 Funkcionality a mechaniky 19
3.1.2 Uživatelské rozhraní a grafické zpracování 20
3.1.3 Provedené testování použitelnosti 21
3.2 Možnosti dalšího rozvoje 21
3.3 Herní návrhový dokument 23
3.3.1 Koncept 23
3.3.2 Příběh a postavy 24
3.3.3 Úrovně a postup hrou 24
3.3.4 Suroviny 25
3.3.5 Budovy 26
3.3.6 Schopnosti 27
3.3.7 Nepřátelé 27
3.3.8 Hudba a zvuky 28
3.4 Persony 28
3.4.1 Pavel 28
3.4.2 Luboš 29
3.4.3 Markéta 30
3.5 Případy užití 31
3.6 Heuristická analýza současného UI 38
3.6.1 Hlavní menu 38
3.6.2 Uživatelské rozhraní v průběhu hry 39
3.6.3 Obrazovka s vylepšeními 40
3.6.4 Shrnutí 41

4 Návrh **43**
4.1 Vyhodnocení možností dalšího rozvoje 43
4.2 Uživatelské rozhraní 46
4.2.1 Hlavní menu 46
4.2.2 Rozhraní v průběhu úrovně 47
4.2.3 Rozhraní mezi úrovněmi 50
4.3 Budovy 52
4.4 Správa surovin 55
4.5 Vylepšení a persistence dat 56
4.6 Nepřátelé 58

5 Realizace **59**
5.1 Příběh a jeho prezentace 59
5.2 Tutoriál 60
5.3 Uživatelské rozhraní 62
5.3.1 Rozhraní v průběhu úrovně 63

5.3.2 Rozhraní mezi úrovněmi 64
5.3.3 Nastavení 65
5.3.4 Ovládání 67
5.4 Úrovně 67
5.5 Vizuální efekty 68
5.6 Zvuky 69
5.6.1 Zvukové efekty 70
5.6.2 Hudba 70
5.6.3 Řeč 70

6 Testování **71**
6.1 Dotazník 71
6.2 Verze a změny hry v průběhu testování 73
6.3 Výsledky testování 74
6.3.1 Úvodní otázky 74
6.3.2 Srozumitelnost tutoriálu 74
6.3.3 Obtížnost jednotlivých úrovní 76
6.3.4 Hodnocení různorodých aspektů hry 78
6.3.5 Návrhy na další obsah 80
6.3.6 Informace o respondentech 80
6.4 Změny provedené po testování 81

Závěr **83**

Bibliografie **85**

A Seznam použitých zkratek **89**

B Obrázky návrhu UI **91**

C Obsah přiloženého média **101**

Seznam obrázků

2.1 Schéma vzoru Abstraktní továrna 14
2.2 Diagram MVC 17
2.3 Diagram MVP 17

3.1 Ilustrace umělé inteligence 24
3.2 Ilustrace jednotlivých úrovní 25
3.3 Ikony budov a schopností 26
3.4 Persona Pavel 28
3.5 Persona Luboš 29
3.6 Persona Markéta 30
3.7 Diagram případů užití 32

4.1 Návrh obrazovky s nastavením 47
4.2 Návrh rozhraní s informacemi o budově 48
4.3 Návrh rozhraní pro interakci s budovou 48
4.4 Návrh obrazovky s informacemi o vylepšení 51
4.5 Návrh obrazovky pro výběr úrovně 51
4.6 Diagram tříd pro rafinerii v prototypu hry 53
4.7 Diagram tříd pro rafinerii v nové podobě 54
4.8 Diagram tříd pro správu surovin v nové podobě 56
4.9 Diagram tříd pro vylepšení v nové podobě 57

5.1 Ukázka příběhové scény 60
5.2 Panel jednoho z úkolů v tutoriálu 61
5.3 Ukázka rozhraní úrovně 63
5.4 Náhled kulometu při jeho stavění 64
5.5 Obrazovka s vylepšeními 65
5.6 Panel pro přistání na mapě s trhlinou 66
5.7 Nastavení obrazu 66
5.8 Nové statické textury ve hře 67
5.9 Změny textury nepřátel v závislosti na zdraví 69

6.1 Čas strávený hraním Rogue Planet 75
6.2 Verze hry, které hráči testovali 75
6.3 Volnost tutoriálu 75
6.4 Hodnocení obtížnosti úrovně Lava Flats 77
6.5 Hodnocení obtížnosti úrovně Giant Ravine 77
6.6 Hodnocení obtížnosti úrovně Impact Crater 78
6.7 Počet podobných her, které testeři hráli 80

Seznam tabulek

6.1 Jasnost konceptů v tutoriálu 76
6.2 Hodnocení různorodých aspektů hry 79

Úvod

Rogue Planet je 2D strategická hra v grafickém stylu pixel art, jejíž základy byly položeny v bakalářské práci *Rogue planet: využití multiagentních systémů ve 2D hře* téhož autora obhájené v roce 2021. Ta se zabývala vícero modely chování nepřátelských agentů ve hře založených na multiagentních systémech a jejím hlavním výstupem je hratelný prototyp. Pozornost byla též věnována podobě uživatelského rozhraní, mechanikám hry a grafickému zpracování. V této diplomové práci bude navázáno na výsledky výše zmíněné bakalářské práce s cílem strategickou hru Rogue Planet dopracovat a vydat pro budoucí hráče.

Strategické hry jsou herním žánrem, ve kterém hraje hlavní roli hráčova schopnost plánování a taktického uvažování pro dosažení vítězství. Často v nich hráči musí hospodařit s omezenými zdroji a pečlivě zvažovat své kroky, což platí i v případě Rogue Planet. Jedná se o klasický videoherní žánr s mnoha podžánry, který je stále důležitou součástí videoherního průmyslu.

V současné době videoherní průmysl svojí výdělečností překonává hudební i filmové odvětví dohromady, a je tak bezkonkurenčně nejvýdělečnějším odvětvím zábavního průmyslu. Ačkoli jsou nejprodávanější videohry doménou především velkých vývojářských studií, díky dnešním možnostem digitální distribuce her a existenci stále pokročilejších volně dostupných kreativních nástrojů významně roste i trh s nezávislými hrami tvořenými jedinci či malými týmy vývojářů.

Aby se Rogue Planet dala považovat za plnohodnotnou strategickou hru a mohla stát po boku ostatních nezávislých herních titulů, bude jí třeba vylepšit v mnoha různých ohledech. Tato práce proto bude zahrnovat jak klasické úkony softwarového inženýrství, jako je zvýšení kvality kódu a jeho další rozšiřování nebo analýza a návrh uživatelského rozhraní, tak i úkony spadající do kategorie herního designu, jako je návrh herních mechanik, tvorba a prezentace příběhu hry, level design, zvukový design či vyvažování herní obtížnosti.

Kapitola 1

Cíl práce

Cílem této práce je navrhnout a realizovat finální podobu hry Rogue Planet a vydat ji pro budoucí hráče. Pro tento účel budou nejprve představeny metody a principy softwarového inženýrství přínosné pro tvorbu počítačových her a návrhové vzory, které se při vývoji her používají. Ty pak budou využity při analýze současného stavu prototypu Rogue Planet, návrhu finální podoby hry a následné implementaci.

Bude provedena analýza současného stavu již existujícího prototypu hry Rogue Planet a možností, jak v jejím vývoji pokračovat a hru úspěšně dopracovat do finální podoby. V rámci krátkého herního návrhového dokumentu bude rovněž stručně popsána cílová podoba hry z pohledu herního návrhu. Jedním z důležitých cílů nového herního návrhu bude začlenění již existujících modelů chování nepřátelských agentů do hlavního herního módu. Pro účely vytvoření kvalitní a použitelné nové podoby uživatelského rozhraní hry budou kromě podrobného rozboru stavu uživatelského rozhraní prototypu též důsledně analyzovány případy užití nového rozhraní, které se budou odvíjet mimo jiné i od nových mechanik popsaných v návrhovém dokumentu. Za stejným účelem pak budou rovněž vytvořeny persony cílových uživatelů.

Na základě výsledků analýzy pak budou zvoleny vhodné možnosti dalšího vývoje hry a bude vypracován návrh její nové podoby se zaměřením na její kompletnost a použitelnost. Bude dbáno na dodržování představených principů softwarového inženýrství a na vhodných místech bude využito zmíněných návrhových vzorů.

Dle vypracovaného návrhu bude implementován finální herní produkt. Ten pak bude důsledně otestován s důrazem na funkčnost a hratelnost. Výsledky testování budou pečlivě zanalyzovány a budou provedeny úpravy potřebné pro úspěšné vydání hry. Výsledná hra pak bude publikována skrze vhodnou platformou pro distribuci her.

Kapitola 2

Rešerše

V této kapitole jsou představeny metody a principy softwarového inženýrství užitečné pro tvorbu počítačových her. Budou představeny metody návrhu a testování uživatelského rozhraní, které mohou pomoci s tvorbou nové verze rozhraní hry, jež bude pro hráče snadno použitelná a bude též zahrnovat funkcionality a mechaniky nově přidané do hry. Dále pak bude věnována pozornost principům softwarového inženýrství, které je třeba dodržovat při přetváření prototypu hry na plnohodnotný softwarový produkt. Na závěr pak budou popsány návrhové vzory využívané při vývoji počítačových her, které budou v tomto přetváření ve vhodných případech využity.

2.1 Metody návrhu uživatelského rozhraní

Tato sekce se věnuje metodám a technikám, které se běžně užívají pro návrh uživatelského rozhraní softwarových produktů. Využití těchto metod pomáhá vývojářům lépe navrhovat uživatelské rozhraní softwaru, které bude pro jeho uživatele přívětivé a snadno použitelné.

2.1.1 Persony

Persony jsou nástrojem interakčního designu, který navrhl softwarový návrhář a programátor Alan Cooper, který jej též svojí knihou *The Inmates Are Running The Asylum* [1] z roku 2004 zpopularizoval.

Cooper zdůrazňuje, že je lepší vyvíjet software pro jednu osobu, než se snažit vytvářet software pro vícero osob s různými požadavky. Funkcionalita přidaná pro pohodlí některých lidí může být obtíží pro ostatní, a tak těžko budou uspokojeni všichni, spíše naopak. Při zúžení návrhu na jednu konkrétní osobu naopak nic nestojí mezi danou osobou a kompletní spokojeností. Než takového cílového zákazníka hledat a ptát se ho na otázky ohledně návrhu, na které stejně nejspíše nebude znát odpovědi, je dle Coopera vhodné

si zákazníka reprezentovat právě personou. Persony jsou tedy hypotetickými archetypy cílových uživatelů a jsou úzce spjaty s cíli uživatelů.

Při tvorbě persony je dle Coopera třeba dbát na konkrétnost. Konkrétní formulace a místy až zdánlivě přehnaná důslednost mají vést k tomu, že se z neurčité persony stává skutečný člověk, do jehož role se návrháři i programátoři dokáží lépe vžít a jsou ochotnější s ním pracovat. Důležitá je též uvěřitelnost, ať už se jedná o jméno, fotku či popis dané osoby. Z tohoto důvodu má při tvorbě person obvyklost a reprezentativnost přednost před diverzitou a výstředností.

Person je dle Coopera vytvářeno vícero v závislosti na obsáhlosti projektu, a to na základě kvalitativních dat, přičemž alespoň jednu personu je třeba identifikovat jako primární, tedy jako hlavní personu, pro kterou je uživatelské rozhraní vytvářeno. Pokud je v projektu identifikováno více hlavních person, je třeba vytvořit též více uživatelských rozhraní. Je možné také identifikovat takzvanou negativní personu, která pomůže specifikovat, pro koho uživatelské rozhraní naopak vytvářeno není.

Kromě Cooperova přístupu k personám založeného na uživatelských cílech a konkrétním popisu lze dle Lene Nielsenové [2] rozlišit ještě tři hlavní částečně odlišné přístupy, které vznikly později.

Přístup orientovaný na role se kromě cílů zaobírá i chováním uživatelů. Popisy person zde musí být podpořeny množstvím kvalitativních i kvantitativních dat. Soustředí se na vztah mezi získanými daty, personou a její rolí v organizaci a výstupem je poměrně velké množství person.

Skrze příběhy usiluje o hlubší porozumění personám takzvaný poutavý přístup s důrazem na to, aby vývojář personu skutečně vnímal jako osobu ve snaze předejít rutinním řešením ze strany programátorů. Vyžaduje rozsáhlá kvantitativní data o uživatelích.

Posledním z přístupů je tvorba person založených na fikci, někdy též nazývaných *proto persony*. Persony jsou tvořeny na základě intuice, zkušeností a představ vývojářů o jejich zákaznících, nebo dle již existujících dat bez provádění dalších průzkumů. Tento přístup je vhodný při nedostatku dat či času pro jejich sběr.

2.1.2 Případy užití

Případy užití (angl. use cases) jsou způsobem zachycení požadavků z pohledu použitelnosti užívaným při softwarovém návrhu. Prvně byly představeny Ivarem Jacobsonem [3] na konference OOPSLA 87, avšak běžného užívání v praxi se dočkaly až po vydání jeho knihy *Object-Oriented Software Engineering: A Use Case Driven Approach* v roce 1992.

Myšlenku případů užití dále rozvinul Alistair Cockburn ve své knize *Writing Effective Use Cases*. [4] Cockburn zde případy užití představuje jako popis možné posloupnosti interakcí mezi systémem a aktéry ve vztahu k určitému cíli. Mohou být použity k popisu pracovních procesů v podniku, jako základ

diskuse o systémových požadavcích softwaru, jako samotné funkční požadavky, či jako forma dokumentace. Kvůli jejich velké rozmanitosti Cockburn dělí případy užití dle následujících os rozdílnosti:

- casual / dressed
- business / system
- corporate / computer system / system innards
- white-box / black-box
- strategic / user-goal / subfunction

Každý případ užití je obvykle možné zařadit dle kombinace hodnot těchto os. V pozdější části své knihy Cockburn definici případu užití zužuje na vyjádření cíle primárního aktéra vůči systému obsahující též kolekci scénářů. Scénář je pak posloupnost akcí vedoucí ke splnění daného cíle, či selhání. Při tvorbě případů užití lze poměrně volně nastavit jejich strukturu a podrobnost dle potřeb projektu. Je však vždy třeba dbát na jejich správnost.

2.2 Metody testování uživatelského rozhraní

Již byly představeny metody pro návrh kvalitního uživatelského rozhraní. Použití těchto metod ale samo o sobě kvalitní výsledek nemůže zajistit, a proto je třeba tyto metody podpořit testováním vytvářeného rozhraní. Metody testování uživatelského rozhraní představené v této sekci umožňují nalézat chyby v návrhu rozhraní a usměrnit jeho vývoj k uspokojivému výsledku.

2.2.1 Heuristická evaluace

Heuristická evaluace představená Jakobem Nielsenem [5] je metodou systematického průchodu uživatelského rozhraní pro zvýšení použitelnosti, která je vhodná i pro použití při iterativním vývoji. Tato metoda probíhá bez účasti uživatelů. Je prováděna experty, kteří posuzují, do jaké míry je uživatelské rozhraní v souladu se zavedenými heuristikami.

Jeden expert je schopen najít jen poměrně malý podíl chyb v rozhraní, a proto je vhodné, aby průchod provádělo více expertů. Množství nedostatků je však omezené a při vyšším počtu expertů již přidávání dalších přinese mizivé množství nových nálezů. Ideální počet expertů je tedy dle Nielsena obvykle tři až pět. Aby byla metoda efektivní, musí experti průchod provádět samostatně, aby jejich nálezy byly nezávislé a nebyly ovlivněné ostatními experty. Ke sdílení výsledků dojde až po dokončení průchodů. Jakob Nielsen [6] pro heuristickou evaluaci popisuje následujících deset heuristik:

1. **Viditelnost stavu systému**
   Uživatel by měl být vždy v rozumné době informován o tom, co se právě děje, a to skrze vhodnou zpětnou vazbu.

2. **Shoda mezi systémem a realitou**
   Systém by měl používat slova, kterým uživatelé rozumí a používají je. Pořadí prezentovaných informací by mělo být přirozené a odpovídat logice reálného světa.
3. **Uživatelská kontrola a svoboda**
   Jelikož se uživatelé často dopouštějí chyb, měli by mít možnost snadno ukončit neúmyslně započaté akce.
4. **Konzistence a standardy**
   Používané výrazy a symboly by měly být konzistentně užívány v rámci systému i v souladu s dalšími systémy, na které je uživatel zvyklý.
5. **Prevence chyb**
   Je třeba identifikovat situace s vysokou pravděpodobností na pochybení ze strany uživatele. Těmto situacím je vhodné buď předejít, nebo po uživateli žádat potvrzení dané akce. Přednost mají možné chyby s velkým dopadem.
6. **Rozpoznání spíše než vzpomínání**
   Je vhodné snížit náročnost na uživatelovu paměť. Neměl by být nucen pamatovat si informace mezi jednotlivými části rozhraní. Potřebné informace musí být uživateli dostupné k rozpoznání, než aby si na ně musel pracně vzpomínat.
7. **Flexibilita a efektivita používání**
   Je vhodné pravidelným uživatelům poskytnout metody pro usnadnění a zefektivnění jejich práce, které pro nové uživatele nemusí být viditelné. Typickým příkladem jsou klávesové zkratky nebo možnosti individuálního nastavení.
8. **Estetika a minimalistický design**
   Nadbytečné a zřídka používané informace pouze snižují přehlednost rozhraní, a je tyto informace tedy vhodné skrýt. Přednost mají elementy podporující uživatelovy hlavní cíle.
9. **Pomozte uživatelům rozpoznat, porozumět a zotavit se z chyb**
   Chybové hlášky musí jednoduše a jasně popsat problém a navrhnout vhodné řešení. Měly by být výrazné a jasně viditelné.
10. **Nápověda a dokumentace**
    Uživatelům by měla být dostupná nápověda a dokumentace, kterou lze snadno prohledávat.

Tyto heuristiky je možné pro evaluaci doplnit o další principy specifické pro konkrétní systém či jeho část. Mezi výhody heuristické evaluace dle Nielsena patří též fakt, že pro její provedení není nutné mít použitelný systém, ale lze ji provádět i nad neinteraktivním prototypem, pokud je dostatečně podrobný.

2.2.2 Kognitivní průchod

Kognitivní průchod je dle Whartonové, Riemana, Lewise a Polsona [7] metoda kontroly použitelnosti zaměřená na to, jak snadné je pro uživatele naučit se pracovat se systémem, a to především prozkoumáváním. Experti v této metodě posuzují použitelnost v sezeních na základě předpokládaných informací o uživatelích, scénářů, úloh a stanoveného kontextu použití.

Během průchodu se postupně posuzují jednotlivé úlohy, přičemž se porovnává předpokládané chování uživatele v dané situaci s možnostmi, které mu navržené uživatelské rozhraní nabízí. Rozhraní může být pro kognitivní průchod definováno mockupem či prototypem.

Metoda může být prováděna skupinově i individuálně. Při skupinové evaluaci představuje tvůrce rozhraní svoji vizi kolegům a na základě jimi poskytnuté zpětné vazby upraví rozhraní při tvorbě další verze tak, aby nalezené nedostatky byly eliminovány. Při samostatném provedení kognitivního průchodu vyhodnocuje tvůrce svůj vlastní návrh, což mu může pomoci internalizovat si principy užívané při kognitivním průchodu pro zlepšení budoucího návrhu.

Před začátkem průchodu je třeba definovat, jakým způsobem je popsáno rozhraní systému, kdo jsou uživatelé, jaké činnosti budou analyzovány a jaký je správný postup řešení daných činností. Při samotném průchodu jsou pak hledány odpovědi na následující otázky pro jednotlivé postupy řešení:

- Pokusí se uživatel dosáhnout správného efektu?
- Všimne si uživatel, že je správná akce dostupná?
- Spojí si uživatel správnou akci s účinkem, kterého se snaží dosáhnout?
- Pozná uživatel po zvolení správné akce, že došlo k pokroku pro naplnění cíle?

Řešení nalezeného problému je pak dle Whartonové et al. závislé na tom, která z otázek měla negativní odpověď. Obecně bývá lepší akce odstraňovat či reorganizovat dle uživatelova vnímání úlohy, než se soustředit na lokální vylepšení.

2.2.3 Testování použitelnosti

Uživatelské testování použitelnosti je dle Nielsena [5] základní metodou, která poskytuje vývojářům informace o tom, jak s vyvíjeným rozhraním interagují skuteční uživatelé a na jaké problémy při interakci narážejí.

Testování použitelnosti se dělí dle účelu na formativní a sumativní. Formativní testování je určeno k nacházení slabých a silných stránek rozhraní pro další zlepšení v rámci vývoje. Sumativní testování se užívá k celkovému ohodnocení použitelnosti rozhraní, například pro porovnání dvou různých návrhů.

Testování se dle Nielsena skládá ze čtyř částí. Nejprve je třeba připravit místo a materiály pro provedení testu. Poté následuje představovací fáze, kdy

jsou testujícím uživatelům vysvětleny účely a průběh testu. Samotný test pak probíhá ideálně bez zásahu vedoucího testu. Uživatel zde plní předem připravené úkoly, nejlépe s přemýšlením nahlas. Nakonec proběhne závěrečný pohovor, při kterém je uživateli dána možnost vyjádřit míru spokojenosti se systémem a případná doporučení. Taktéž je vhodné probrat problémy či nejasnosti, na které se narazilo během testu.

Aby byly výsledky co nejvíce přesné a validní, je třeba dbát na to, aby bylo prostředí testu co nejpodobnější podmínkám, za kterých bude systém skutečně nasazen. Totéž platí i pro prováděné úkony. Ještě důležitější jsou pak samotní testeři, kteří musí co nejvíce odpovídat předpokládaným uživatelům.

Způsob získávání dat se liší dle účelu prováděného testu. Problematické části uživatelského rozhraní, které je třeba upravit, lze snadno identifikovat podle průběhu testu a závěrečného rozhovoru jako takového. V případě potřeby kvantitativních dat je vhodné testování nahrávat.

2.3 Principy softwarového inženýrství

V softwarovém inženýrství existuje poměrně vysoké množství různých principů pro tvorbu srozumitelného, udržovatelného a snadno rozšiřitelného kódu. Touto podkapitolou budou představeny nejdůležitější z nich.

2.3.1 Principy SOLID

SOLID je akronym pro pět principů objektově orientovaného návrhu, které říkají, jak funkcionality a data programu uspořádat do tříd a tyto třídy propojit. Softwarový inženýr Robert Martin [8], jenž tyto principy sestavil, je pak popisuje následovně:

Single responsibility principle
Každý modul by měl mít právě jeden důvod se měnit. Měl by být odpovědný právě jednomu aktérovi.

Open-closed principle
Softwarový systém by měl být otevřený rozšířením, ale uzavřený změnám. Měl by tedy být navržen tak, aby nové funkcionality byly přidávány dodáním nového kódu, nikoli úpravou stávajícího.

Liskov substitution principle
Aby bylo možné stavět systém ze zaměnitelných částí, musejí se tyto části podřizovat kontraktu, který dovolí jejich záměnu.

Interface segregation principle
Je třeba se vyhýbat vzniku závislostí na funkcionalitách, které nepoužíváme.

Zbytečné závislosti přinášejí zbytečné potíže, a tak je vhodné se jim vyhnout rozdělováním funkcionalit do menších rozhraní.

Dependency inversion principle
Kód implementující funkcionality vysoké úrovně by neměl záviset na kódu implementujícím nízkoúrovňové detaily. Ve flexibilním systému se tedy ve zdrojovém kódu nacházejí reference pouze na abstrakce a nikoli na konkrétnosti.

2.3.2 DRY – Don't repeat yourself

Vzhledem k častým změnám informací v průběhu života systému přinesli Hunt a Thomas [9] pro snazší údržbu, přehlednost a spolehlivost princip DRY s následujícím zněním: „Veškeré znalosti musí mít uvnitř systému právě jednu jednoznačnou a autoritativní reprezentaci.“ [9] (překlad autora) V případě vícero reprezentací je nutné při změně jedné myslet i na úpravy ostatních, jinak dojde k rozporům uvnitř systému.

2.3.3 KISS – Keep it simple stupid

Tento princip praví, že je třeba usilovat o největší možnou jednoduchost. Uplatnění dle Sheikhové [10] nalezne při tvorbě uživatelských rozhraní, kde uživatelům jednoduchost může usnadnit používání, v softwarovém inženýrství pro srozumitelnější návrh, i v programování pro tvorbu čitelného kódu.

2.3.4 SoC – Separation of concerns

Princip oddělení zodpovědností se zakládá na myšlence rozdělení programu do více modulů s minimálním překryvem funkcionalit. Makabee [11] uvádí, že míru dodržení tohoto principu lze sledovat s využitím metrik zvaných provázanost (angl. coupling) a soudržnost (angl. cohesion). Provázanost určuje míru závislosti mezi jednotlivými moduly. Soudržnost pak určuje míru spojitosti funkcí jednoho modulu. Dobré oddělení zodpovědností se projevuje nízkou provázaností a vysokou soudržností.

2.4 Refaktorování

Refaktorování (angl. refactoring) je dle Fowlera [12] proces, při kterém je vylepšována vnitřní struktura systému, aniž by se měnilo jeho pozorovatelné chování. Cílem je již existující kód udělat lépe srozumitelným a rozšiřitelným, zlepšit design softwaru a snížit technický dluh.

Martin Fowler identifikoval ve své knize *Refactoring: Improving the Design of Existing Code* [12] celkem 22 příznaků, kdy je vhodné přemýšlet o refaktorování kódu, jím označovaných jako „pachy v kódu“. Tyto pachy pak lze dle stránky *Refactoring Guru* [13] logicky rozdělit do pěti kategorií:

- **Bobtnající kód**
  Příliš dlouhé třídy a metody jsou těžko udržovatelné a práce s nimi je složitá. Nabobtnalé třídy i metody mají být děleny na menší metody a třídy, pokud je to možné. Logické pojmenování takto vzniklých částí kódu pak může výrazně zlepšit srozumitelnost a čitelnost kódu. Součástí této kategorie jsou též zbytečně dlouhé seznamy parametrů metod či lpění na používání primitivních datových typů v případech, kdy by lépe posloužil složený datový typ.
- **Špatné použití objektově orientovaného programování**
  Objektově orientované programování (OOP) může při nevhodné aplikaci přinášet více potíží než užitku. Příkladem je dědění v případě, kdy je potřeba jen část funkcionality rodiče. Nedostatečné využití OOP se naopak může projevit častým výskytem operátoru `switch` v místech, kde by ho bylo vhodné nahradit polymorfismem.
- **Vady zabraňující změnám**
  Tato kategorie značí případy, kdy změna funkcionality vyžaduje provedení mnoha malých změn kódu, nebo vyžaduje provedení jedné změny kódu na mnoha různých místech.
- **Zbytečnosti**
  V průběhu vývoje může dojít k tomu, že nějaká část kódu přestane být užitečná. Důvodem může být změna požadavků na systém během vývoje, nebo nadbytečná snaha o vývoj funkcionalit, které v budoucnu možná budou užitečné. Problémem jsou i duplicity v kódu či nadměrné množství komentářů, jež bývá často příznakem, že kód vyžaduje složité vysvětlení, jelikož je nepřehledný.
- **Nadbytečná provázanost**
  Poslední kategorií jsou případy, kdy jedna třída nadužívá data či interní metody jiné třídy. V takových případech je vhodné zvážit přesun funkcionality do používané třídy. Problematické mohou být i dlouhé řetězce volání metod či zbytečné použití prostředníka.

2.5 Návrhové vzory

Dalším užitečným nástrojem softwarového inženýrství jsou takzvané návrhové vzory, které nabízejí praxí ověřená řešení často se opakujících problémů, na které je při návrhu systému možné narazit. Nejedná se vyloženě o připravená řešení pro začlenění do softwarového projektu, ale spíše o abstraktnější šablony, podle kterých je možné potřebné řešení zkonstruovat.

Tato kapitola se bude zabývat především návrhovými vzory využívanými při tvorbě počítačových her. Čerpá hojně ze tří zdrojů. Prvním z nich je kniha *Design Patterns* [14] shrnující velké množství návrhových vzorů, jejímiž autory jsou Gamma, Helm, Johnson a Vlissides, známí též jako „Gang of Four“.

Druhým je kniha *Game Programming Patterns* [15] od Roberta Nystroma zaměřující se na návrhové vzory pro vývoj počítačových her. Třetím je pak kniha *Level up your code with game programming patterns* [16] od Unity Technologies, která se zabývá využitím návrhových vzorů pro vývoj her v Unity. Vzhledem k častému střídání těchto zdrojů v průběhu textu je pro přehlednost za každou částí textu této části rešerše uvedeno, ze kterého z těchto zdrojů parafrázuje bez dalšího opakování jmen těchto knih a jejich autorů.

Nejprve bude provedeno krátké shrnutí návrhových vzorů, které jsou implementovány přímo uvnitř Unity. Poté budou představeny další vzory, a to především ty, jež se užívají v designu počítačových her v Unity dle *Level up your code with game programming patterns*. [16]

2.5.1 Návrhové vzory implementované v Unity

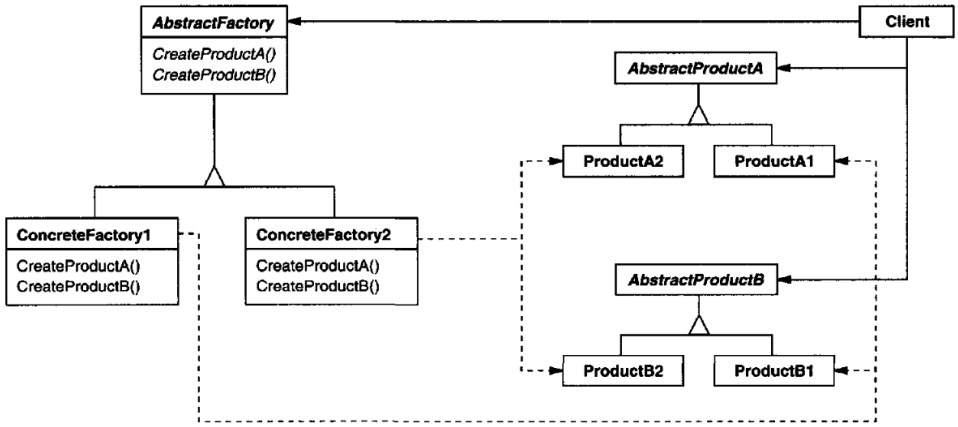
Představme si nejprve čtyři návrhové vzory, které herní engine Unity v základu sám přímo implementuje [16]:

Prototype
Tento vzor umožňuje vytváření kopií objektů bez vzniku závislosti na jejich třídě či nutnosti pro každou třídu vytvořit dedikovanou podtřídu pro její kopírování. [14] *Prefab* systém v Unity umožňuje vytvářet vzory herních objektů, libovolně je duplikovat a vytvářet varianty s mírnými odlišnostmi. [16]

Component
Díky tomuto vzoru je možné vytvářet menší třídy s jednou zodpovědností a složitější objekty pak sestavovat z nich. Zároveň lze vytvořené komponenty využívat v objektech, aniž by tím mezi nimi vznikla závislost. [15] Unity poskytuje širokou paletu komponent pro fyziku, animace, zvuky, grafiku atd., které jde skládat a utvářet tak herní objekty. Vlastní kód pak do hry dodáváme komponentou *Script*. [16]

Game loop
Videoherní návrhový vzor Game loop se stará o nezávislost běhu času ve hře na výkonnosti počítače a uživatelském vstupu. Unity obstarává jak smyčku s pevnou délkou kroku skrze metodu `FixedUpdate`, tak i smyčku s variabilním časem `Update` s možností získat čas, který uplynul mezi současným a předchozím během smyčky. [16]

Update
Aktualizovat stav všech objektů při každém při každém snímku umožňuje návrhový vzor Update. Jedná se o poměrně jednoduchý vzor, kde každý objekt implementuje aktualizační metodu, kterou pak při každém snímku voláme pro každý objekt právě jednou. [15] Unity tuto funkcionalitu zprostředkovává



Obrázek 2.1: Schéma vzoru Abstraktní továrna [14]

automaticky metodami `Update`, `LateUpdate` a `FixedUpdate` pro všechny třídy, které dědí od třídy `MonoBehaviour`. [16]

2.5.2 Abstract Factory

Návrhový vzor Abstract Factory (česky Abstraktní továrna) umožňuje tvořit rodiny příbuzných objektů bez závislosti na jejich konktétních třídách. Pro použití továrny musí její produkty (tj. generované objekty) implementovat společné rozhraní. [14]

Při implementaci tohoto rozhraní je možné pro každý objekt definovat jiné chování provedené při jeho vzniku. Pro tvorbu více rodin objektů pak lze použít abstrakci nad továrnou, která umožní definovat rozličné vlastnosti rodin v rámci každé konkrétní továrny. [16] Schéma vzoru Abstract Factory můžete vidět na obrázku 2.1.

2.5.3 Object Pool

Object Pool je optimalizační návrhový vzor, při kterém je snižováno využití výkonu a paměti tím, že místo opakovaného vytváření a mazání používáme opakovaně objekty z jedné kolekce. [15] Používané objekty si lze připravit ve chvíli, kdy si uživatel sníženého výkonu nevšimne (například při načítání úrovně) a v případě potřeby je již jen aktivujeme.

Vhodným případem užití tohoto návrhového vzoru ve hře jsou například střely vystřelené ze zbraně. Hráč jich může během minut vystřelit stovky či dokonce tisíce, avšak aktivních střel prolétajících vzduchem bude vždy v jeden okamžik relativně málo. Ve verzi 2021 bylo do Unity přidáno *UnityEngine.Pool API*, které využívání tohoto vzoru výrazně usnadňuje. [16]

2.5.4 Singleton

„Singleton zajišťuje, že má třída právě jednu instanci a poskytuje k ní globální přístup.“ [14] (překlad autora) Kromě zmíněného udržování právě jedné instance a zajištění globálního přístupu k ní umožňuje tento vzor vytvořit instanci až v moment, kdy je skutečně potřeba a tím šetřit paměť. U her však tato funkcionalita může být kontraproduktivní, jelikož načítání instance v čase potřeby může způsobit krátký zásek, který by hráčovi zhoršil zážitek více než načítání instance na začátku hry či během načítání úrovně. [15]

Titul *Game Programming Patterns* na rozdíl od ostatních návrhových vzorů v případě vzoru Singleton věnuje poměrně velkou část příslušné kapitoly vysvětlování, proč by tento vzor neměl být používán. Singleton může mít velmi negativní vliv na provázanost aplikace, ztížit testování a přinášet skryté závislosti, proto ho je třeba používat s mírou. Jelikož se ale jedná o jednoduché řešení mnoha problémů, bývá často nadužíván. Navíc je často zbytečně používán v případech, kdy je vyžadována garance pouze jedné ze dvou vlastností Singletonu. [15] *Level up your code with game programming patterns* tyto nevýhody též zmiňuje, přesto použití Singletonu především pro manažerské třídy s potřebou globálního přístupu na menších her schvaluje. Je však vždy třeba dbát na to, aby tento vzor nebyl používán příliš. [16]

2.5.5 Service Locator

Jednou z alternativ k používání vzoru Singleton je návrhový vzor Service Locator. Účelem tohoto vzoru je poskytnutí globálního přístupu ke službě bez vzniku závislosti na konkrétní implementační třídě, čímž je snížena míra provázanosti aplikace. Skládá se z jedné či více abstraktních služeb, jejich implementací a jednoho lokátoru, který služby lokalizuje, udržuje, a poskytuje k nim přístup zbytku aplikace. Ačkoli se dá použití vzoru Service Locator považovat za zlepšení oproti přímé vazbě na Singleton, stále tento vzor trápí některé z neduhů Singletonu a je ho tedy rovněž vhodné používat střídmě. [15]

2.5.6 Command

Tento vzor zapouzdřuje požadavek jako objekt. Tím je umožněno požadavky zaznamenávat, vkládat do fronty, vracet zpět, či provádět opožděně. Command může sloužit jako objektově orientovaná alternativa pro callback. Jeho základ tvoří třída definující rozhraní příkazu, které vždy obsahuje metodu `Execute` k provedení příkazu a případně i metodu `undo` pro jeho zrušení. Toto rozhraní je pak implementováno libovolným počtem konkrétních příkazových tříd. [14]

2.5.7 State

Návrhový vzor State (česky Stav) umožňuje měnit chování objektu na základě jeho vnitřního stavu. Skládá se z třídy `Context`, jejíž vnitřní stav má být reprezentován, třídy `State`, která definuje společné rozhraní pro stavy a z tříd implementujících jednotlivé stavy. [14]

Prakticky se pak jedná o implementaci konečného automatu. Konečný automat lze reprezentovat i naivně například výčtovým typem v kombinaci s výrazem `switch`, což ale vede k růstu velikosti a nepřehlednosti třídy, která všechny stavy reprezentuje. Vzor Stav naproti tomu stavové chování implementuje ve shodě s principy SOLID, kdy lze nové stavy snadno přidávat nezávisle na již existujících stavech. [16] Zároveň potenciálně nákladné vyhodnocování podmínky nahrazuje voláním virtuální funkce. [14]

2.5.8 Observer

Vzor Observer, známý též jako Publish-Subscribe, umožňuje definovat vazbu 1:N mezi objekty tak, že vícero pozorovatelů může sledovat změny stavu jednoho subjektu, aniž by subjekt na těchto pozorovatelích musel záviset, či znát jejich počet. [14]

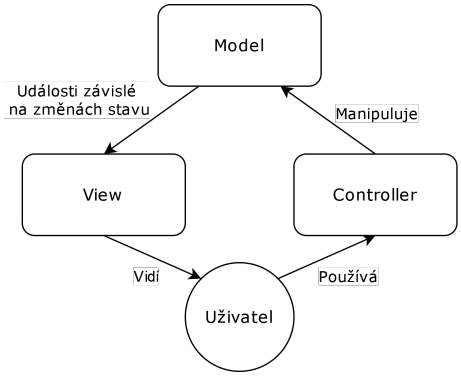
Tento vzor je díky jeho užitečnosti a častému používání implementován přímo v jazyce C# skrze události (angl. events). Subjekty definují události a pozorovatelé se k nim mohou zaregistrovat funkcí odpovídající delegátovi dané události. Když pak subjekt událost vyvolá, registrované funkce pozorovatelů na ni automaticky reagují.

Kromě C# událostí, které lze v Unity použít, nabízí Unity i vlastní implementaci událostí zvanou *UnityEvents*. Implementace událostí od Unity nabízí možnost spravovat události v editoru, avšak oproti systémovým událostem z C# bývají pomalejší. Jejich použití je tedy vhodné především v případě, kdy je třeba události zpřístupnit kolegům, kteří nepracují s kódem, či dalších situacích, kdy vývojářům přístup z editoru usnadní práci a případné zpomalení nevadí. [16]

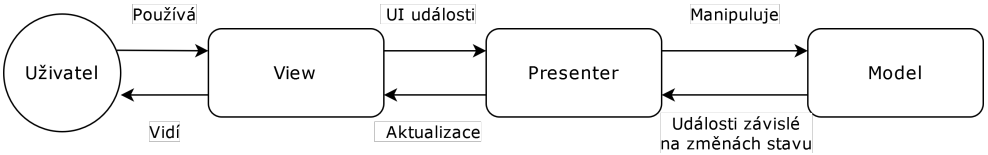
2.5.9 Model View Controller a Model View Presenter

Model View Controller (MVC) je návrhový vzor pro rozdělení aplikace na tři logické vrstvy: *Model* pro správu dat, *View* pro zobrazení informací na obrazovce a *Controller* pro zpracování herní logiky a uživatelských vstupů. MVC tak pomáhá dodržovat princip jedné zodpovědnosti (první princip SOLID). Schéma vztahů vrstev MVC můžete vidět na obrázku 2.2.

Model View Presenter (MVP) je vzor odvozený od MVP, který rovněž dělí aplikaci na tři vrstvy, avšak zodpovědnosti vrstev se oproti MVP mírně liší. *Model* stále zodpovídá za správu dat, *View* však kromě zobrazování informací slouží i ke zpracování vstupů. *Presenter* je pak prostředníkem mezi



Obrázek 2.2: Diagram MVC (vytvořeno autorem dle diagramu v [16])



Obrázek 2.3: Diagram MVP (vytvořeno autorem dle diagramu v [16])

těmito vrstvami, jenž jedním směrem formátuje a předává data pro zobrazení, a druhým směrem přijímá informace o vstupech a manipuluje dle nich s daty. [16] Schéma vzoru MVP můžete vidět na obrázku 2.3.

MVC a MVP jsou oproti dříve zmíněným vzorům poměrně velké architektonické vzory a jejich využití vyžaduje větší přípravy a není je v *Unity* možné uplatnit ve všech případech. Jejich využití vede k snadnějšímu využití unit testů, a u větších vývojářských týmů umožňují snazší dělbu práce díky možnosti pracovat na jednotlivých vrstvách odděleně. [16]

2.5.10 Data Locality

Poměrně obecný optimalizační vzor *Data Locality* praví, že je možné výrazně urychlit program tím, že data v paměti uspořádáme v pořadí, ve kterém jsou zpracovávána pro co nejlepší využití mezipaměti procesoru.

Jeho využití je vhodné především v případě problémů s výkonem kvůli výpadkům mezipaměti (angl. cache miss). Pro dobré uspořádání dat v paměti je obvykle nutné vyhnout se skákání mezi částmi paměti skrze ukazatele a reference. Implementace *Data Locality* tím pádem vede k nutnost omezit v kódu abstrakce jako jsou rozhraní či dědičnost. [15]

2.5.11 Entity Component System

Entity Component System (ECS) je dle Johna Terry [17] návrhový vzor, který je zdánlivě podobný *Entity-Component* (EC) frameworkům. EC umožňuje tvorbu komponent a skládání těchto komponent do entit. ECS oproti tomu

dělí systém do tří částí, kterými jsou entity, komponenty a systémy. Dokumentace [18] balíčku *Entities* implementujícího ECS v Unity tvrdí, že entity reprezentují jsoucna, která jsou součástí aplikace. Samy o sobě neobsahují žádná data ani chování, pouze uvádějí, z jakých komponent se skládají. Komponenty nesou data potřebná pro různá chování entit. Nejsou v paměti uspořádány dle příslušnosti k entitám, ale dle vlastní logiky, která je uzpůsobena vzoru *Data Locality* a dělá z ECS datově orientovaný architektonický vzor. Systémy pak obsahují logiku prováděnou nad komponentami.

Terra [17] uvádí, že ECS lze do určité míry považovat za alternativu ke klasickému objektově orientovanému programování. ECS oproti OOP dává přednost skládání před dědičností. Data jsou zde oddělena od chování a místo zapouzdření dat je preferováno použití jednodušších datových struktur, které v kombinaci s vhodným uspořádáním lépe využívají mezipaměť procesoru.

Jak již bylo dříve zmíněno v podsekci *Návrhové vzory implementované v Unity* 2.5.1, Unity poskytuje EC framework pro tvorbu a skládání komponent. V rámci *Data Oriented Technology Stack* (DOTS) pak Unity poskytuje balíček *Entities* [18], který je implementací vzoru ECS.

Kapitola 3

Analýza

V této kapitole naleznete popis současného stavu prototypu hry a analýzu možností dalšího vývoje k dosažení plně funkční hry, kterou bude možné vydat. Následovat bude stručný popis plánovaného finálního stavu hry z pohledu herního návrhu v podobě jednoduchého herního návrhového dokumentu. Rovněž bude provedena analýza person a případů užití potřebných pro nový návrh uživatelského rozhraní. V poslední části kapitoly pak bude provedena heuristická analýza současného stavu uživatelského rozhraní s cílem identifikovat nedostatky, které je v nové verzi třeba eliminovat.

3.1 Současný stav Rogue Planet

Rogue Planet je 2D strategická hra, jejíž základy především v podobě analýzy podobných titulů, návrhu základů hry, hratelného prototypu implementovaného v herním enginu *Unity* a provedených testů použitelnosti rozhraní byly položeny v bakalářské práci [19] autora této magisterské práce v roce 2021. Cílem této diplomové práce je na její výstupy navázat a Rogue Planet dopracovat do podoby vhodné ke zveřejnění. V této podkapitole budou stručně shrnuty aspekty hry, které již byly navrženy a implementovány. Podrobnější popis může být nalezen přímo ve zmíněné bakalářské práci. Některé implementační detaily prototypu budou dále probrány společně s případným návrhem optimálnějšího řešení v kapitole 4.

3.1.1 Funkcionality a mechaniky

Ve hře se hráč ujímá role kapitána vesmírné lodi s cílem vytěžit vzácné suroviny na nově objevené toulavé planetě[1]. Ve výstavbě těžební kolonie a prozkoumávání povrchu hráčovi kromě věčné tmy brání především místní mimozemský život.

[1] Angl. *rogue planet*, jedná se o vesmírné těleso velikosti planety, které neobíhá kolem žádné hvězdy [20].

Budovy, suroviny a prvky mapy
Hra obsahuje osm druhů budov, které hráč může stavět na herní mapě, postupně ji tak osvětlovat, nacházet suroviny k těžbě a bránit se před útoky nepřátel. Těžené suroviny v podobě barevných rud může hráč spotřebovávat k výrobě energie, kterou všechny budovy potřebují k funkci, nebo rafinovat na krystaly. Bílá ruda pak slouží k stavbě budov.

Mapa obsahuje kromě surovin též další prvky v podobě flóry, lávy či prastarých budov, které lze se správnými vylepšeními dále využívat k osvětlování mapy. Jejich podoba v prototypu hry je nicméně poměrně strohá a jedná se spíše o doplněk pro větší rozmanitost mapy, než o plnohodnotné mechaniky.

Přistávání a vylepšování
Své přistání na planetě může hráč kdykoliv ukončit a vrátit se s přistávacím modulem zpět na loď na orbitě. To přináší do hry rogue-lite element v podobě krystalů získaných rafinací z natěžených rud, které lze na lodi použít k pořízení různých vylepšení hráčových budov pro další přistání. Hráči tak zůstávají pouze krystaly, zakoupená vylepšení a případné znalosti o rozložení mapy. Vše ostatní se při odletu z planety resetuje. Po zakoupení vylepšení může hráč opět přistát na povrchu planety a pokusit se dostat dále, než při předchozím přistání, nebo alespoň získat krystaly na vylepšení pro další pokusy.

Nepřátelští agenti
Součástí prototypu hry jsou tři různé modely chování nepřátelských agentů. První z nich odpovídá autorově představě o vhodném chování agentů z pohledu hratelnosti, další dva jsou pak inspirovány chováním včel a mravenců při hledání potravy.

Pouze první model chování se nachází přímo v hlavním herním módu. Všechny tři si pak s různými nastaveními může hráč prohlédnout ve speciálním předváděcím herním módu. Agenti se na mapě pravidelně rodí na předem určených místech zvýrazněných dírami v zemi.

Persistence dat
Krystaly a zakoupená vylepšení budov nepřetrvávají pouze mezi jednotlivými přistáními, ale přetrvají i po vypnutí a zapnutí hry. Persistence těchto údajů je zajištěna skrze speciální třídu Unity `PlayerPrefs`.

3.1.2 Uživatelské rozhraní a grafické zpracování

Hra je stylizována do grafického stylu pixel art. Uživatelské rozhraní také dodržuje tento styl, přičemž tématicky usiluje o retrofuturisticky působící kombinaci vzhledu počítačového terminálu ve sci-fi zasazení. Inspirací pro vzhled UI byly především hry *Duskers* [21] a *Alien: Isolation* [22], které tuto kombinaci též využívají.

Uživatelské rozhraní ve hře je možné rozdělit na tři distinktivní části, a to hlavní menu, UI při samotném hraní (informace o budovách a surovinách, výběr budovy pro postavení, menu při pozastavení hry atd.) a obrazovku pro kupování trvalých vylepšení.

3.1.3 Provedené testování použitelnosti

Rozhraní prošlo uživatelským testováním a mnoho odhalených nedostatků bylo již opraveno. Největším neopraveným problémem z hlediska použitelnosti se dle výsledků zdá být absence tutoriálu, který by hráče seznámil s herními mechanikami. Za největší slabinu provedeného testování považuji výběr testerů, který kvůli časové tísni a probíhající pandemii Covid-19 neodpovídal dostatečně cílové skupině uživatelů.

3.2 Možnosti dalšího rozvoje

V této sekci budou představeny možné oblasti dalšího rozvoje hry Rogue Planet oproti současnému stavu po dokončení bakalářské práce, ve které byl vytvořen její prototyp.

- **Přehlednost a rozšířitelnost kódu**
  Ačkoli byly při tvorbě prototypu na kvalitu kódu brány ohledy, jedná se stále o prototyp, a prioritou tak bylo především rychlé dodání funkční verze. Bylo by tak vhodné zdrojové kódy upravit, aby odpovídaly již zmíněným principům softwarového inženýrství, k čemuž mimo jiné může posloužit i vhodné využití návrhových vzorů.
- **Uživatelské rozhraní**
  Ačkoli se současné uživatelské rozhraní osvědčilo jako uspokojivé, obsahuje oproti původnímu návrhu různorodé dodatky, jejichž potřeba se projevila až během realizace a testování. Bylo by tak vhodné navrhnout novou verzi rozhraní, která tato rozšíření lépe začlení a bude zároveň obsahovat i případné další funkcionality navržené v rámci této práce.
- **Využití modelů chování agentů**
  Současná implementace hry Rogue Planet obsahuje tři modely chování agentů. V hlavním herním módu se však hráči setkají pouze s jedním z nich a ostatní si lze prohlédnout pouze ve speciálním předváděcím režimu. Pro plnohodnotnou podobu hry by bylo vhodné nalézt způsob, jak v hlavním herním režimu využít všechny tři modely. Tento nedostatek je poznamenán i v posudku oponenta bakalářské práce [23], ve které byl vývoj Rogue planet započat.

- **Tutoriál**
  V provedeném testování použitelnosti se negativně projevila absence tutoriálu, který by hráče postupně seznámil s dostupnými mechanikami. Je tak třeba navrhnout vhodnou podobu tutoriálu a tu implementovat.

- **Grafické provedení**
  Ačkoli je současný vzhled hry ucházející a odpovídá zvolenému stylu, je zde jednoznačně prostor pro vylepšení. Toho lze dosáhnout například přidáním částicových a dalších vizuálních efektů, postprocessingu nebo více animací.

- **Hudba a zvuky**
  Současný prototyp hry neobsahuje žádnou hudbu ani zvukové efekty. Ty přitom mohou výrazně přispět k navození správné atmosféry či pomoci hráčovi porozumět, co se ve hře právě děje.

- **Persistence dat**
  V současnosti je ukládáno pouze množství hráčem ukořistěných krystalů a zakoupená vylepšení. Bylo by možné ukládání rozšířit na další součásti hry. Pro takovouto změnu by však bylo nutné přejít od jednoduššího způsobu ukládání přes třídu `PlayerPrefs` k pokročilejší technice, kterou je například serializace dat do souboru ve formátu JSON.

- **Množství mechanik a obsahu**
  Hru lze rozšířit o další mechaniky pro odhalování mapy, obranu základny, získávání energie či těžbu a zpracování surovin. Je možné též přidat zcela nové mechaniky či prvky mapy. Současné prvky mapy jsou navíc nyní spíše zpestřením než plnohodnotnou mechanikou. Lze je tak více rozvinout po funkční stránce i z pohledu zasazení do příběhu hry.

- **Příběh**
  Nyní je příběh hry pouze pomůckou pro stylizaci a navození atmosféry při návrhu prvků hry, avšak sám o sobě se ve hře nijak nevyskytuje. Je možné příběh dále rozpracovat a integrovat do hry.

- **Ovladatelnost a klávesové zkratky**
  Hra poskytuje pouze jednoduché ovládání pro pohyb po mapě a interakci s budovami. Ovládání lze rozšířit o další způsoby navigace a interakce, na které jsou hráči zvyklí z obdobných her. Pokročilejším hráčům je pak vhodné usnadnit práci přidáním klávesových zkratek v souladu s Nielsenovou šestou heuristikou (viz 2.2.1).

- **Možnosti nastavení**
  Hra v současnosti neobsahuje prakticky žádné možnosti nastavení. Je jí tedy vhodné rozšířit o nastavení hlasitosti zvuků, rozlišení hry, množství a kvality efektů atd. Taktéž lze hráčům poskytnout možnost změny přiřazení kláves.

- **Verze Unity**
  Prototyp Rogue Planet byl vytvořen ve verzi Unity 2019.4.13f1 LTS, která byla v době jeho vytváření nejnovější verzí Unity s dlouhodobou podporou. V současnosti je nejnovější verzí Unity s dlouhodobou podporou verze 2021.3.19f1 LTS. Aby bylo možné využívat nejnovějších funkcí Unity, je třeba hru převést na novější verzi.
- **Optimalizace**
  V případě výkonnostních nedostatků je možné hru více optimalizovat například kompletním přepsáním kódu agentů či dalších entit pro použití frameworku ECS.
- **Lokalizace**
  V současnosti je celá hra v anglickém jazyce. Může být rozšířena, aby podporovala i další lokalizace (například českou).
- **Herní obtížnost a vyváženost**
  Je vhodné nalézt optimální nastavení příjmu surovin, cen budov, vylepšení a dalších aspektů hry tak, aby nedocházelo k přílišné jednoduchosti či obtížnosti v různých částech hry. Špatná vyváženost může vést k nadužívání některých prvků hry, nebo naopak k úplné ignoraci prvků, které pro hráče budou nevýhodné.
- **Testování**
  Je třeba navrhnout rozšířené uživatelské testování, které bude pokrývat nové funkcionality a ideálně využívat testery blízké cílové skupině. Testování je vhodné rozšířit nejen na použitelnost uživatelského rozhraní, ale též hratelnost, vyváženost a funkčnost na různých systémech.

3.3 Herní návrhový dokument

Tato sekce je věnována stručnému popisu plánovaného finálního stavu hry Rogue Planet z pohledu herního návrhu, pro což byla zvolena forma krátkého herního návrhového dokumentu [24] (angl. game design document). Návrhový dokument vznikl souběžně s ostatními částmi kapitol 3 a 4. Doplněn je pak rovněž texturami a ilustracemi vytvořenými až v kapitole 5. Oproti obvyklé struktuře herních návrhových dokumentů zde nebude obsažena část věnovaná cílovému publiku, jelikož budou cíloví uživatelé popsáni v rámci analýzy person v sekci 3.4. Ze stejného důvodu se zde nenachází ani popis uživatelského rozhraní, jehož návrhu a realizaci jsou věnovány sekce 4.2, respektive 5.3.

3.3.1 Koncept

Rogue Planet je sci-fi 2D strategická hra odehrávající se převážně na povrchu toulavé planety. Hráč v roli kapitána korporátní těžařské lodi vysílá na povrch přistávací modul, kolem kterého pak staví další budovy, objevuje temný povrch planety a těží nalezené suroviny. Natěžené suroviny může hráč přeměnit

Obrázek 3.1: Ilustrace umělé inteligence, která kráče provází hrou

na herní měnu, kterou může použít po návratu na loď k pořízení trvalých vylepšení, která mu umožní přistávat na dalších částech planety a usnadní mu postup při dalších přistáních zlepšením vlastností jeho budov.

3.3.2 Příběh a postavy

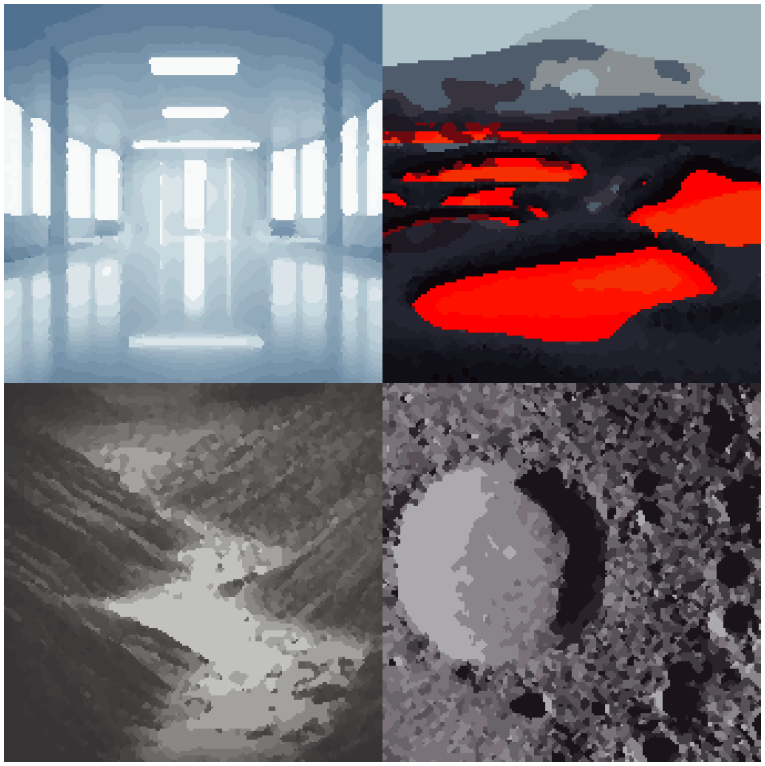
Příběh začíná probuzením hráče z kryospánku před příletem hráčovy lodě na oběžnou dráhu planety, kde se hra bude primárně odehrávat. Do děje je hráč uveden lodní umělou inteligencí, která hráče provází i po zbytek hry a je kromě hráče jedinou postavou ve hře. Portrét reprezentující lodní umělou inteligenci ve hře můžete vidět na obrázku 3.1. Hráč je obeznámen s faktem, že se na planetě nachází artefakt prastaré mimozemské civilizace, který musí získat. Lodní vybavení bohužel neumožňuje přistát přímo u lokace artefaktu, tudíž hráč musí nejprve získat na lépe přístupných místech suroviny pro vylepšení přistávacího modulu, a až poté může artefakt vyzvednout.

Hlavní komplikací jsou pro hráče uměle vytvořené formy života určené k obraně zmíněného artefaktu, které hráči ztěžují postup. Hra končí tím, že se hráči podaří dosáhnout lokace artefaktu a bránit se dostatečně dlouho, aby artefakt mohl být vyzvednut.

3.3.3 Úrovně a postup hrou

Hra se skládá celkem ze čtyř úrovní, které hráč postupně odemyká. První úroveň je pouze simulací na palubě lodi, která má hráče seznámit s mechanikami hry a hráč si z ní neodnáší žádné suroviny.

Druhá úroveň je odemčena dokončením první, obsahuje první dvě ze tří barevných rud a nabízí klasickou hratelnost s prozkoumáváním mapy. Tato úroveň je svým vzhledem stylizována jako pobřeží lávové řeky. Láva hráči poslouží jako přírodní zdroj světla a energie, čímž mu usnadní jeho první kroky ve hře.



Obrázek 3.2: Ilustrace jednotlivých úrovní

Třetí a čtvrtá úroveň jsou odemčeny pořízením vylepšení za získané suroviny. Třetí úroveň je hratelností podobná s druhou, avšak má větší rozlohu a nabízí i třetí z barevných rud. Evokovat by měla prostředí dna hluboké trhliny a bude obsahovat fluorescentní houby, které hráči po připojení do sítě mohou posloužit jako přírodní zdroj světla.

Ve čtvrté úrovni má hráč za úkol přežít dostatečně dlouho, aby mohl vyzvednout artefakt. Aby hráči nestačilo bránit pouze jedno místo, což by bylo velmi jednoduché, bude muset pro vyzvednutí artefaktu napájet čtyři pilíře rozmístěné po mapě. Tím přetíží ochranné pole artefaktu a ten může následně vyzvednout. Jeho odnesením zpátky na palubu lodi pak dokončí hru. Vizuálním stylem by úroveň měla připomínat obrovský kráter, v jehož středu se mimozemský artefakt nachází. Jednotlivé úrovně může hráč po odemčení libovolně navštěvovat a opakovat. Ilustrace všech úrovní můžete vidět na obrázku 3.2.

3.3.4 Suroviny

Hra obsahuje čtyři druhy rud, které hráč může těžit. Bílá ruda slouží jako stavební materiál. Tři barevné rudy (červenou, zelenou a modrou) s vzestupnou vzácností lze spotřebovávat pro výrobu energie, nebo rafinovat na barevné

Obrázek 3.3: Ikony budov a schopností

krystaly, které si hráč z mapy při odletu odnáší pro zakoupení vylepšení budov a schopností. Všechny budovy spotřebovávají energii, tudíž hráč musí balancovat mezi tvorbou krystalů pro vylepšení a výrobou energie pro údržbu a rozšiřování základny. Správa surovin a energie je důležitou součástí hry, která nutí hráče ke strategickému uvažování.

3.3.5 Budovy

Ikony všech budov a schopností můžete vidět na obrázku 3.3. Hráčovi je k dispozici osm typů budov, přičemž stavět může sedm z nich. Každý druh budovy má vlastní vylepšení a účel. Dostupné druhy budov jsou následující:

- **Přistávací modul**
  Jediná budova, kterou hráč nemůže sám stavět. Na začátku úrovně má hráč postavenou pouze tuto budovu a další začíná stavět kolem ní. Poskytuje základní příjem energie a bílé rudy pro stavbu budov.
- **Rozvodna energie**
  Tato budova poskytuje energii okolním políčkům. Budovy lze stavět pouze na políčka s energií. Pokud nějaká budova přijde o zdroj energie, automaticky se deaktivuje a nejde aktivovat, dokud není dodávka proudu obnovena.
- **Bodové světlo**
  Osvětluje celé své okolí. Světlo hráč potřebuje k hledání surovin a obranu před nepřáteli.
- **Těžební vrták**
  Po umístění na rudu začne tuto rudu těžit. Lze též umístit na pole s lávou pro zisk energie.

- **Reflektor**
  Má větší dosah než bodové světlo, avšak místo celého okolí osvětluje pouze úzkou kruhovou výseč. Otáčí se kolem své osy a při nalezení nepřítele ho začne sledovat, dokud se nevzdálí z dosahu či není zastřelen.
- **Kulomet**
  Automaticky střílí po nepřátelích v okolí. Může zaměřit pouze nepřátele, kteří jsou osvětlení a nacházejí se v dostřelu kulometu.
- **Generátor**
  Spotřebovává barevnou rudu pro tvorbu energie. Na umístění generátoru nezáleží. Vzácnější rudy jsou v produkci energie efektivnější. Typ rudy, který je spotřebováván, může hráč volně měnit.
- **Rafinérie**
  Spotřebovává velké množství energie a vybrané barevné rudy pro výrobu příslušných barevných krystalů. Krystaly si hráč po dokončení úrovně odnáší zpět na loď pro nákup vylepšení.

3.3.6 Schopnosti

Hráčovy možnosti průzkumu a obrany dále doplňují schopnosti sesílané z lodi, které mu jsou k dispozici zdarma a může je použít kdekoli na mapě. Každá schopnost je po použití určitou dobu nedostupná, než jí je možno použít znovu. Dostupné jsou následující schopnosti:

- **Světlice**
  Na omezenou dobu osvětlí zvolenou oblast. Umožňuje prozkoumat oblast bez nutnosti rozšiřování infrastruktury. Poslouží též jako dočasná náhrada osvětlovacích budov k osvětlení nepřátel pro kulomety.
- **Bombardování**
  Poměrně jednoduchá schopnost, která způsobí vysoké poškození všem nepřátelům v oblasti. Lze využít jako záchrana při nedostatcích v obraně, nebo pro vyčištění oblasti od nepřátel před jejím zastavěním budovami.
- **Radar**
  Zobrazí hráčovi lokaci všech nepřátel na mapě. Tato schopnost nepřátele nezviditelňuje kulometům, avšak je důležitým zdrojem informací pro hráče. Umožňuje snáze pozorovat chování nepřátel a dle něj přizpůsobit obranu základny či nalézt ložiska surovin, která nepřátelé navštěvují.

3.3.7 Nepřátelé

Nepřátelé se budou v úrovních pravidelně rodit na předem stanovených pozicích v závislosti na tvaru herní mapy. Jejich síla by měla růst v závislosti na době uplynulé od začátku úrovně, aby byl hráč nucen úroveň po delší době opustit, a nedržel pouze jednu výhodnou pozici. Agresivita a síla nepřátel

by též měla mít vzestupnou tendenci v závislosti na spuštěné úrovni, aby nepřátelé dokázali držet krok s hráčovými vylepšeními, a i pozdější úrovně pro něj představovaly výzvu.

3.3.8 Hudba a zvuky

Hudba ve hře by měla doplňovat ponurou futuristickou atmosféru hry. Skladby použité v hlavním menu a závěru hry pak budou výraznější. Zvuky by měly doplňovat všechny akce ve hře a působit realisticky.

3.4 Persony

Tato sekce je věnována popisu uživatelských person pro Rogue Planet. Persony by měly pomoci s upřesněním očekávání uživatelů vzhledem k vytvářené hře při definování případů užití. Nápomocné mohou být též při určování priority možných vylepšení popsaných v sekci 3.2.

Persony jsou tvořeny dle autorových zkušeností a představ o cílových uživatelích, zvyklostí z obdobných her a poznatků z předchozí bakalářské práce. Využito bylo též hardwarového a softwarového průzkumu zákazníků [25] od Valve Corporation. Z pohledu rozdělení zmíněného v sekci 2.1.1 se zde popisované persony řadí mezi takzvané *proto persony*. Obrázky person byly vygenerovány s použitím stránky *This Person Does Not Exist*[2] pro generaci portrétů neexistujících osob s pomocí umělé inteligence.

3.4.1 Pavel

Obrázek 3.4: Persona Pavel

Pavel je dvaadvacetiletý student informatiky se zaměřením na teoretickou informatiku. Jeho podobu můžete vidět na obrázku 3.4. Od mládí je studijně nadaný a své středoškolské vzdělání prodělal na gymnáziu. Matematika mu vždy šla a počítačové hry, které hrál už od mládí, v něm vzbudily zvědavost ohledně fungování počítačů a počítačových programů. S počítači

[2] `https://this-person-does-not-exist.com/cs`

strávil většinu svého náctiletého života a po maturitě se tak rozhodl pro studium informatiky na vysoké škole.

Hry hraje Pavel především doma na svém postarším počítači střední třídy, který mu rodiče koupili na střední škole, menší hry rád vyzkouší i na svém notebooku, který s sebou vozí do školy. Je velkým fanouškem her od menších studií, ale sleduje i dění kole her s velkým rozpočtem, které ale sám příliš nehraje. Skromnějšího zpracování mu nevadí, pokud je hra něčím originální či zajímavá, ať už se jedná o neobvyklou kombinaci mechanik, nebo strhující příběh. Nebojí se výzvy a rád řeší zajímavé logické i strategické hádanky, vadí mu ale, když mu nedává hra čas si vše v klidu promyslet. Jelikož hraje mnoho her od nezávislých studií z ciziny, které obvykle neposkytují českou lokalizaci, nedělá mu angličtina problémy. Je fanouškem série *Creeper World*. Budovatelské strategické hry ho velmi baví, tudíž si zahrál i *Prison Architect*, *They Are Billions či Mindustry*. Má rád i jednoduché taktické hry se silnou stylizací, jako je *Hacknet* či *Duskers*.

Z pohledu této práce je Pavel hlavní personou, pro kterou je hra vytvářena. Je tedy potřeba dbát na to, aby mu navrhované funkcionality vyhovovaly a věnovat pozornost tomu, zda je Pavel využije.

3.4.2 Luboš

Obrázek 3.5: Persona Luboš

Luboš je šestatřicetiletý kameraman a střihač na volné noze. Přivydělává si též streamováním her, avšak zatím je to pro něj spíše koníčkem. Jeho podobu můžete vidět na obrázku 3.5. Škola ho nikdy nebavila a studijní výsledky měl velmi různorodé. Již během studia na střední průmyslové škole si začal přivydělávat jako střihač a přiučil se i práci s kamerou. Školu nakonec úspěšně dokončil, ale rozhodl se místo strojírenství následovat svůj zájem o audiovizuální techniku. Několik let pracoval jako kameraman a střihač pro televizní pořad, ale nakonec monotónnost této práce nevydržel a začal se věnovat jednorázovým zakázkám od různým firem jako živnostník. Má rád filmy a hry, přičemž se rozhodl své hraní streamovat pro kamarády. Jeho streamy v průběhu let nabraly větší množství diváků a nyní jsou pro něj mimo zábavu i vedlejším zdrojem příjmů.

Kvůli práci si Luboš pořídil velice drahý a výkonný počítač, na kterém si rád vychutnává hry v maximálních detailech. Hraní her věnuje poměrně velké množství času a stíhá si tak vyzkoušet všechny velké i malé tituly, které právě vycházejí a osloví ho, ačkoli více pozornosti věnuje těm větším. Po spuštění hry si Luboš rád nejprve vše pečlivě nastaví, než se pustí do akce. Angličtina mu problém nedělá, avšak překládání dlouhých textů je pro něj namáhavé a rád se mu vyhne. Tutoriály často přeskakuje s tím, že nejsou potřeba a později si pak ve hře neví rady. Má rád, když si ve hře může připadat jako velký stratég, aniž by musel vymýšlet strategie, které by byly příliš komplexní.

Luboš pro Rogue Planet není hlavní personou. Přesto budou jeho zájmy brány v potaz, pokud nebudou v konfliktu s cíli a zájmy hlavní persony.

3.4.3 Markéta

Obrázek 3.6: Persona Markéta

Markéta je čtyřiadvacetiletá studentka pedagogiky se zaměřením na český jazyk a základy společenských věd. Její podobu můžete vidět na obrázku 3.6. Její skvělí učitelé ji na gymnáziu inspirovali, aby se též vydala na učitelskou dráhu. Vždy ji bavila literatura, a tak se rozhodla pro obor český jazyk, nakonec ji ale více baví výuka sociálních věd. V současnosti studuje magisterské studium a při tom již pracuje na částečný úvazek jako učitelka.

Sama nemá mnoho zkušeností s videohrami, ráda však sleduje videa jejích oblíbených internetových tvůrců, jak různé hry hrají oni. Občas ji nějaká z her z videí zaujme a ráda by si ji též vyzkoušela. Její angličtina je slabší a pokud hra neobsahuje českou lokalizaci, je to pro ni problém. Kvůli menším zkušenostem jí obvykle déle trvá pochopit herní ovládání a mechaniky a proto potřebuje, aby ji tutoriál hry vedl za ruku.

Markétu lze považovat z hlediska vyvíjené hry za negativní personu. Není cílem návrh hry směřovat tak aby byl v rozporu s Markétinými zájmy. Pokud však dojde ke zjištění, že je nějaká funkcionalita přidávána pouze pro Markétu, zřejmě taková funkcionalita nemá pro vyvíjenou hru reálný přínos a hrozí, že bude spíše na obtíž ostatním uživatelům, nebo mohl čas věnovaný jejímu vývoji být věnován přínosnějším funkcionalitám.

3.5 Případy užití

Hra obsahuje z pohledu případů užití pouze jednoho aktéra, kterým je samotný hráč. Na základě současných i plánovaných mechanik byly identifikovány následující uživatelské cíle:

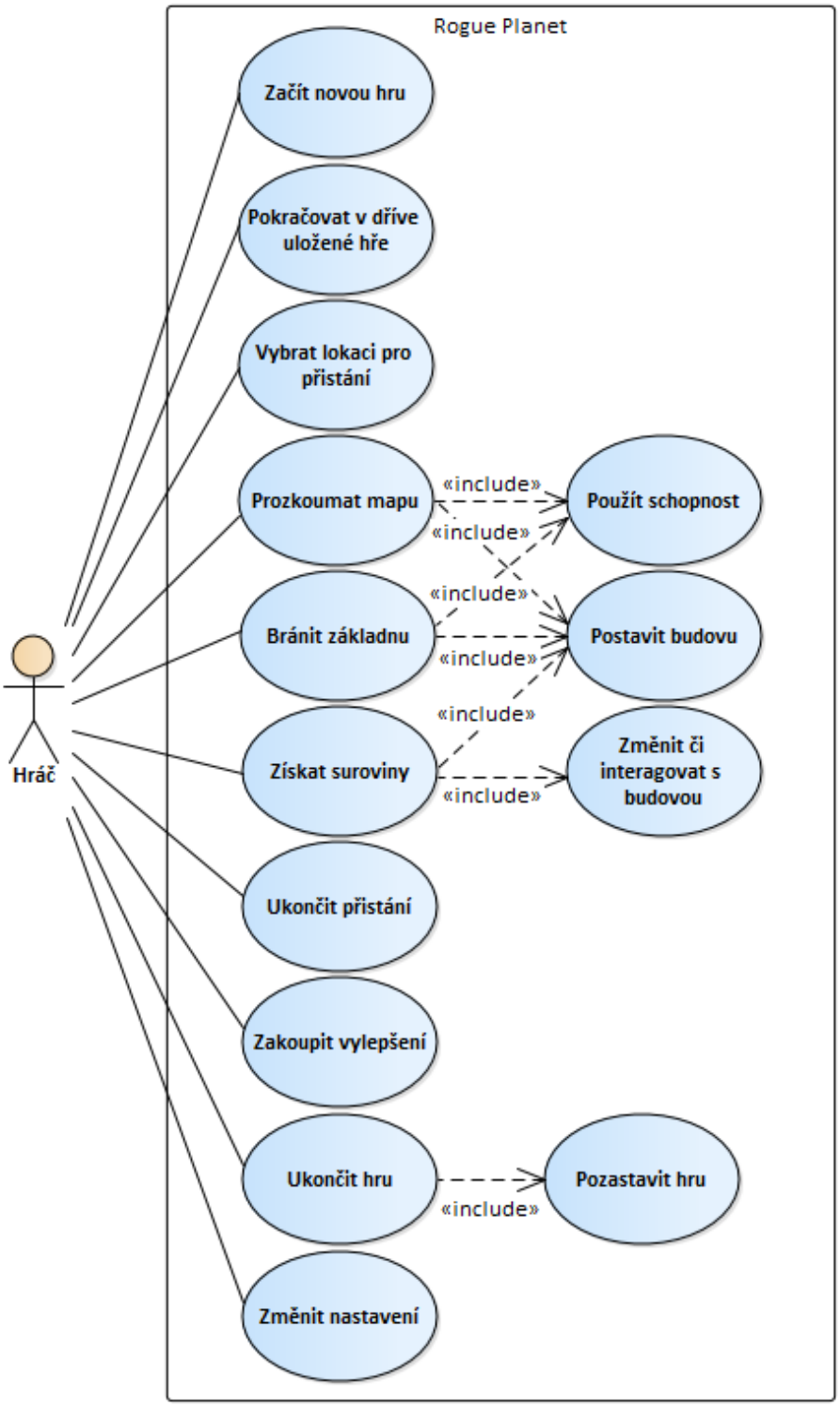
1. vybrat lokaci pro přistání
2. prozkoumat mapu
3. těžit rudu
4. získat energii
5. rafinovat krystaly
6. bránit základnu
7. ukončit přistání
8. zakoupit vylepšení

Z těchto cílů byly odvozeny případy užití zmíněné dále v této sekci. Některé cíle byly sloučeny do jednoho případu užití. Například *získat energii* a *rafinovat krystaly* jsou z pohledu interakce prakticky shodné činnosti, přičemž se liší pouze použitím jiné budovy. Činnosti opakující se při plnění různých cílů, jako *postavit budovu* a *použít schopnost*, pak byly pro přehlednost vyčleněny do samostatných případů užití.

Mimo případů užití odvozených z mechanik hry byly též identifikovány běžné případy interakce s UI nezávislé na mechanikách, jako je *pozastavení hry* či *změna nastavení*. Diagram všech popsaných případů užití můžete vidět na obrázku 3.7.

UC1 Začít novou hru

- **Stručný kontext a cíl:** Hráč chce začít novou hru. Obvykle po prvním spuštění, případně pokud si přeje začít znovu.
- **Podmínky:** Hráč se musí nacházet v hlavním menu hry.
- **Hlavní scénář:**
  1. Hráč zvolí možnost nové hry.
  2. Hra hráč upozorní, že spuštěním nové hry bude přepsán jeho dříve uložený postup.
  3. Hráč potvrdí svoji volbu.
  4. Hra zajistí spuštění nové hry.
- **Rozšíření:**
  - 2a. Pokud hra neobsahuje žádný dříve uložený postup, není upozornění potřeba.
  - 3a. Hráč si zde svoji volbu může rozmyslet a vrátit se do hlavní nabídky.



Obrázek 3.7: Diagram případů užití

UC2 Pokračovat v dříve uložené hře

- **Stručný kontext a cíl:** Hráč chce pokračovat v dříve uložené hře.
- **Podmínky:** Hráč se musí nacházet v hlavním menu hry.
- **Hlavní scénář:**
  1. Hráč zvolí možnost pokračovat ve hře.
  2. Hra zajistí načtení dříve uložené hry a hráčovi zobrazí nabídku míst k přistání.

UC3 Vybrat lokaci pro přistání

- **Stručný kontext a cíl:** Hráč chce zvolit lokaci pro přistání a započít tak fázi hry odehrávající se na herní mapě.
- **Podmínky:** Lokace pro přistání jde volit pouze mezi přistáními. Tedy pouze pokud neprobíhá fáze hry na herní mapě.
- **Hlavní scénář:**
  1. Hráč si nechá od hry zobrazit seznam lokací pro přistání.
  2. Hra zobrazí seznam lokací.
  3. Hráč si zvolí mapu, na které chce přistát.
  4. Hra zobrazí, podrobnější informace o surovinách mapy, případně i text o chování nepřátel či text doplňující příběh. Některé informace mohou být uzamčeny a poskytovány až po odemčení hrou.
  5. Hráč potvrdí přistání na zvolené mapě.
  6. Hra přejde do fáze na herní mapě dle hráčova výběru.
- **Rozšíření:**

  5a. Pokud hráč nemá lokaci odemčenou, tj. nemá vylepšení potřebné pro přistání na tomto místě, hra mu přistání na této mapě neumožní.

UC4 Prozkoumat mapu

- **Stručný kontext a cíl:** Hráč chce prozkoumat určitou část herní mapy, aby zjistil, zda se zde nacházejí suroviny, nepřátelé či další prvky.
- **Podmínky:** Mapu lze prozkoumávat pouze při fázi hry odehrávající se na herní mapě.
- **Hlavní scénář:**
  1. Hráč nalezne oblast, kterou chce prozkoumat.
  2. Hráč oblast osvětlí postavením osvětlovací budovy (*UC9 Postavit budovu*) či schopností (*UC10 Použít schopnost*) na místě, které chce prozkoumat.
  3. Po osvětlení hráč může rozpoznat jednotlivé prvky mapy.

UC5 Bránit základnu

- **Stručný kontext a cíl:** Hráč chce bránit svoji základnu před útoky nepřátel.
- **Podmínky:** Bránit základnu je potřebné a možné pouze při fázi hry odehrávající se na herní mapě.
- **Hlavní scénář:**
  1. Hráč postaví obrannou budovu na místě, které chce bránit. (*UC9 Postavit budovu*)
  2. Hra zajistí, že obranná budova bude automaticky střílet po blížících se nepřátelích.
- **Rozšíření:**
  1a. Alternativně může hráč dočasně bránit základnu použitím vhodné schopnosti (*UC10 Použít schopnost*).
  2a. Obranná budova nemůže vidět neosvětlené nepřátele, a tedy po nich nebude střílet.

UC6 Získat suroviny

- **Stručný kontext a cíl:** Hráč chce získat rudu, energii či krystaly.
- **Podmínky:** Suroviny lze získávat pouze při fázi hry odehrávající se na herní mapě.
- **Hlavní scénář:**
  1. Hráč postaví budovu pro tvorbu surovin, které potřebuje. (*UC9 Postavit budovu*)
  2. Hra zajistí změnu spotřeby a výroby surovin dle postavené budovy.
- **Rozšíření:**
  2a. Při nedostatku surovin pro spotřebu budovou hra budovu deaktivuje a příslušně upraví příjem surovin.
  2b. V případě budovy *těžební vrták* je příjem surovin závislý na tom, zda je umístěna na zdroji surovin.
- **Poznámka:** Alternativně může hráč suroviny získat snížením jejich spotřeby tím, že deaktivuje, zboural nebo přenastaví budovu, která je spotřebovává. (*UC11 Interagovat s budovou*)

UC7 Ukončit přistání

- **Stručný kontext a cíl:** Hráč chce ukončit fázi hry odehrávající se na herní mapě a tím přenést krystaly do fáze vylepšování.
- **Podmínky:** Ukončit přistání lze pouze při probíhajícím přistání, tedy při fázi hry na herní mapě.

- **Hlavní scénář:**
  1. Hráč se rozhodne ukončit přistání a navrátit se přistávacím modulem zpět na loď.
  2. Hráč zvolí přistávací modul.
  3. Hra zobrazí informace modulu včetně možností interakce.
  4. Hráč zvolí možnost návratu na loď.
  5. Hra hráčovi zobrazí informace o proběhlém přistání, které bylo právě dokončeno.
  6. Po potvrzení hra hráče přenese do sekce pro nákup vylepšení.
- **Rozšíření:**

  4a. Hráč si zde může odlet rozmyslet a pokračovat ve hře.

UC8 Zakoupit vylepšení

- **Stručný kontext a cíl:** Hráč si chce zakoupit trvalá vylepšení pro usnadnění hry při dalších přistáních.
- **Podmínky:** Vylepšení jde pořizovat pouze mezi přistáními.
- **Hlavní scénář:**
  1. Hráč naviguje na sekci s vylepšeními.
  2. Hra zobrazí přehled dostupných vylepšení. Je třeba indikovat, která již byla zakoupena a která vylepšení jsou dostupná.
  3. Hráč vybere vylepšení, které chce zakoupit.
  4. Hra indikuje, že vylepšení bylo zakoupeno.
- **Rozšíření:**

  4a. Pokud hráč nemá dostatek krystalů pro zakoupení, je naopak indikováno, že k zakoupení nedošlo.

UC9 Postavit budovu

- **Stručný kontext a cíl:** Hráč chce postavit budovu.
- **Podmínky:** Stavět budovy lze jen při fázi hry odehrávající se na herní mapě.
- **Hlavní scénář:**
  1. Hráč zvolí z nabídky druh budovy, který chce postavit.
  2. Hra zobrazí informace o zvoleném druhu budovy. Taktéž indikuje, kde lze budovu stavět. Stavět je možné pouze na místech pokrytých energetickou sítí.
  3. Hráč zvolí lokaci, kde chce budovu postavit.
  4. Hra zajistí vytvoření budovy na zvoleném místě.

- **Rozšíření:**
  - 1a. Volbu budovy může hráč před postavením změnit nebo zrušit.
  - 4a. Alternativně hra signalizuje, že budovu nelze postavit, pokud hráč nemá suroviny potřebné pro stavbu, nebo nezvolil validní lokaci.

UC10 Použít schopnost

- **Stručný kontext a cíl:** Hráč chce použít jednu z dostupných schopností.
- **Podmínky:** Používat schopnosti lze jen při fázi hry odehrávající se na herní mapě.
- **Hlavní scénář:**
  1. Hráč zvolí z nabídky schopnost, kterou chce použít.
  2. Hra zobrazí informace o dané schopnosti a její rozsah.
  3. Hráč zvolí lokaci, kde chce schopnost použít.
  4. Hra zajistí použití schopnosti na zvoleném místě.
- **Rozšíření:**
  - 1a. V případě schopností s globálním efektem je výběr lokace použití přeskočen.
  - 1b. Alternativně hra signalizuje, že schopnost nejde použít, pokud byla již nedávno použita.
  - 3a. Volbu schopnosti může hráč během výběru lokace změnit nebo zrušit.

UC11 Interagovat s budovou

- **Stručný kontext a cíl:** Hráč chce deaktivovat budovu či změnit její nastavení.
- **Podmínky:** Stavět budovy lze jen při fázi hry na herní mapě.
- **Hlavní scénář:**
  1. Hráč zvolí budovu, se kterou si přeje interagovat.
  2. Hra hráčovi poskytne možnosti interakce závislé na druhu budovy a aktuální informace o zvolené budově.
  3. Hráč zvolí z nabídky akci, kterou chce provést. Např. změnit druh surovin, deaktivovat budovu či zbourat budovu.
  4. Hra zajistí provedení zvolené akce.
- **Rozšíření:**
  - 1a. Hráč může volbu zrušit nebo zvolit jinou budovu.

UC12 Pozastavit hru

- **Stručný kontext a cíl:** Hráč chce pozastavit hru.
- **Podmínky:** Hru lze pozastavit jen při fázi hry na herní mapě.
- **Hlavní scénář:**
  1. Hráč zmáčkne klávesu pro pozastavení hry
  2. Hra zobrazí menu pozastavené hry s možnostmi, jako je návrat do hry či přesun do hlavního menu.

UC13 Ukončit hru

- **Stručný kontext a cíl:** Hráč chce ukončit hru.
- **Podmínky:** Bez podmínek, avšak v různých fázích hry probíhá odlišně.
- **Hlavní scénář:**
  1. Pokud je hráč ve fázi přistání, nejprve pozastaví hru. (*UC12 Pozastavit hru*)
  2. V nabídce akcí hráč zvolí ukončení hry.
- **Alternativní scénář:**
  1. Ve fázi mezi přistáními zvolí hráč možnost návratu do hlavního menu.
  2. Hra hráčovi poskytne hlavní menu s nabídkou akcí, z nichž jedna bude ukončením hry.
  3. Hráč tuto možnost zvolí a hra je tím ukončena.

UC14 Změnit nastavení

- **Stručný kontext a cíl:** Hráč chce změnit nastavení hry.
- **Podmínky:** Hráč se nachází v hlavním menu, nebo pozastavil hru.
- **Hlavní scénář:**
  1. Hráč zvolí nabídku nastavení.
  2. Hra zobrazí dostupná nastavení hry, která hráč může měnit.
  3. Hráč zadá nové hodnoty nastavení.
  4. Hra zajistí, že nové nastavení bude aplikováno.

3.6 Heuristická analýza současného UI

Pro rozšíření poznatků o použitelnosti uživatelského rozhraní prototypu Rogue Planet získaných uživatelským testováním provedeným v závěru bakalářské práce [19] byla provedena heuristické evaluace tohoto rozhraní. Rozhraní je hodnoceno dle Nielsenových heuristik popsaných v části 2.2.1. Obecně je dle Nielsena [5] vhodné, aby evaluace byla prováděna vícero hodnotiteli. Vzhledem k nutnosti samostatného vypracování diplomové práce je však jediným hodnotitelem evaluace autor práce. Nalezené problémy byly kategorizovány dle následující stupnice závažnosti:

- Kosmetický – Možnost zlepšení s malým dopadem na použitelnost.
- Mírně závažný – Mírně znesnadňuje hraní.
- Středně závažný – Znesnadňuje hraní hry, nebo může způsobit drobné chyby.
- Velmi závažný – Velice znesnadňuje hraní hry, nebo může způsobit vážné chyby.

3.6.1 Hlavní menu

1. Viditelnost stavu systému
   - V hlavním menu nemá hráč přehled o tom, jaký je stav jeho uložené hry, respektive zda nějaká vůbec existuje. (Mírně závažný)
   - V sekci s manuálem by mohlo být lépe zvýrazněno, na které stránce se hráč nachází. (Kosmetický)
   - Obdobně by i v sekci s volbou ukázky chování agentů mohla být aktivní stránka lépe zvýrazněna. (Kosmetický)
2. Shoda mezi systémem a realitou
   - Sekce s volbou ukázek chování agentů obsahuje popis nastavení parametrů chování, kterým však hráč bez přečtení příslušné bakalářské práce nemá šanci porozumět. (Mírně závažný)
3. Uživatelská kontrola a svoboda
   - Nenalezeny problémy. Veškeré akce v hlavním menu může hráč snadno vrátit zpět. Nevratná je pouze akce pro smazání uložené hry, která je ze své podstaty nevratná, na což je hráč upozorněn a také je vyžadováno potvrzení.
4. Konzistence a standardy
   - Ačkoli jsou významy tlačítek *hrát* a *obnovit postup* srozumitelné, obvyklejší bývá kombinace tlačítek pro započetí nové hry a pokračování v uložené hře. (Kosmetický)

5. Prevence chyb
   - Nenalezeny problémy. Jediná nevratná akce je ošetřena.
6. Rozpoznání spíše než vzpomínání
   - Nenalezeny problémy.
7. Flexibilita a efektivita používání
   - Chybí možnost vrátit se z jednotlivých částí zpět na hlavní obrazovku klávesou *Escape*. (Mírně závažný)
8. Estetika a minimalistický design
   - Sekce s volbou ukázek chování agentů obsahuje příliš textu najednou. (Kosmetický)
9. Pomozte uživatelům rozpoznat, porozumět a zotavit se z chyb
   - Nenalezeny problémy.
10. Nápověda a dokumentace
    - Nenalezeny problémy.

3.6.2 Uživatelské rozhraní v průběhu hry

1. Viditelnost stavu systému
   - Hráč je na zničení jeho budov či nedostatku surovin pro produkci upozorněn, avšak je obtížné poznat, kde k této události došlo. (Mírně závažný)
   - Při stavbě obranných věží ani po ní hráč nevidí jejich dostřel. (Středně závažný)
   - Dosah rozvodny energie je zobrazen pouze po postavení rozvodny. (Mírně závažný)
2. Shoda mezi systémem a realitou
   - Ačkoli je uspořádání surovin v informační tabulce logické, není na první pohled příliš přehledné. (Kosmetický)
3. Uživatelská kontrola a svoboda
   - Nenalezeny problémy.
4. Konzistence a standardy
   - Nenalezeny problémy.
5. Prevence chyb
   - Hráč je na nedostatek surovin pro stavbu budovy upozorněn pouze poté, co se budovu pokusí postavit. (Kosmetický)

6. Rozpoznání spíše než vzpomínání
    - Chybí signalizace, jaký druh rudy budovy právě spotřebovávají. Hráč tuto informaci získá až zvolením konkrétní budovy. (Mírně závažný)
    - Obdobně hráč nepozná, že je budova poškozena, pokud ji ručně neoznačí. (Mírně závažný)
    - Hráč si nemůže snadno zobrazit informace o prvcích mapy, jako jsou houby, rudy, láva atd. (Kosmetický)
7. Flexibilita a efektivita používání
    - Chybí klávesové zkratky pro volbu budov ke stavbě, ovládání rychlosti času a interakci s označenou budovou. (Mírně závažný)
8. Estetika a minimalistický design
    - Tabulka surovin zbytečně obsahuje v základu suroviny, ke kterým hráč získá přístup až později v průběhu hry. (Kosmetický)
9. Pomozte uživatelům rozpoznat, porozumět a zotavit se z chyb
    - Nenalezeny problémy.
10. Nápověda a dokumentace
    - Dodatečné informace o prvcích hry jsou seskupeny v jedné společné nápovědě a nejsou pohodlně dostupné v situaci, kdy hráč potřebuje informace o konkrétním prvku. (Mírně závažný)

3.6.3 Obrazovka s vylepšeními

1. Viditelnost stavu systému
    - Nenalezeny problémy.
2. Shoda mezi systémem a realitou
    - Návaznost jednotlivých vylepšení by mohla být lépe zvýrazněna. (Kosmetický)
3. Uživatelská kontrola a svoboda
    - Chybí možnost vzít nákup vylepšení zpět. (Středně závažný)
4. Konzistence a standardy
    - Nenalezeny problémy.
5. Prevence chyb
    - Chybí signalizace, zda si hráč dostupné vylepšení může dovolit. Hráč je na nedostatek surovin upozorněn až po neúspěšném pokusu o zakoupení. (Mírně závažný)

6. Rozpoznání spíše než vzpomínání
    - Kvůli rozdělení vylepšení do více záložek je pro hráče pracnější určit, kterou kombinaci vylepšení si může za své suroviny dovolit. (Mírně závažný)
7. Flexibilita a efektivita používání
    - Chybí možnost přesouvat se mezi záložkami klávesami. (Kosmetický)
8. Estetika a minimalistický design
    - Nenalezeny problémy.
9. Pomozte uživatelům rozpoznat, porozumět a zotavit se z chyb
    - Nenalezeny problémy.
10. Nápověda a dokumentace
    - Chybí možnost zobrazit si více informací o složitějších vylepšeních. (Kosmetický)

3.6.4 Shrnutí

Během heuristické analýzy nebyly nalezeny žádné velmi závažné problémy v oblasti použitelnosti, zřejmě díky tomu, že se je již podařilo vyladit během vývoje prototypu hry.

Hlavní menu hry trápí spíše drobné nedostatky, jako je chybějící indikace stavu uložené hry a nižší přehlednost podstránky s výběrem ukázek chování nepřátelských agentů. Jako největší problém rozhraní v průběhu hry byla identifikována absence signalizace dostřelu kulometů během jejich stavění i po něm. Nedostatek informací během stavění trápí i budovu pro rozvod energie. Problémem je též fakt, že pro získání mnohých informací hráč musí nejprve zvolit konkrétní budovu a je tedy pracné mít přehled o stavu vícero budov. Největším problémem obrazovky s vylepšeními je pak nemožnost brát zpět svá rozhodnutí. Ostatní problémy této části rozhraní jsou spíše kosmetického charakteru.

Kapitola 4

Návrh

Prototyp hry byl vytvořen s důrazem na rychlé získání funkční a hratelné verze hry. Při rychlém vývoji vzniklo mnoho technického dluhu a nedokonalostí. To vzhledem k účelu prototypu není problém, avšak pro plnohodnotnou verzi hry je vhodné kód upravit tak, aby lépe odpovídal principům SI a byl lépe srozumitelný a rozšířitelný.

Tato kapitola se zabývá návrhem úprav, které bude pro tyto účely třeba provést. Nejprve bude vyhodnoceno, které z možností dalšího vývoje identifikovaných v analýze je vhodné zvolit. Následovat bude kompletní návrh nové verze uživatelského rozhraní a poté bude věnována pozornost novému softwarovému návrhu stěžejních částí hry.

4.1 Vyhodnocení možností dalšího rozvoje

Pro začátek návrhu a vývoje je nejprve nutné posoudit, které z možných cest dalšího vývoje budou pro vytvářenou hru přínosné a které nikoliv. Tato sekce se proto vrací k možnostem rozvoje popsaných v sekci 3.2, přičemž zde bude na základě přínosu pro použitelnost a hratelnost rozhodnuto, které z dostupných cest dopracování hry budou zvoleny a jakou budou mít formu.

- **Přehlednost a rozšířitelnost kódu**
  Navzdory tomu, že hráč změny v kvalitě kódu a softwarové struktuře přímo nepozná, budou mít pozitivní vliv na složitost dalšího rozšiřování hry a prevenci chyb. Z tohoto důvodu bude značná část stávajícího kódu přepracována tak, aby lépe odpovídal zmiňovaným principům softwarového inženýrství a ve vhodných případech využíval návrhové vzory.
- **Uživatelské rozhraní**
  Jelikož má uživatelské rozhraní velký dopad na použitelnost a hratelnost hry, bude navržena a realizována nová podoba uživatelského rozhraní, která bude reagovat na nedostatky nalezené při testování předchozí verze UI, a zároveň bude podporovat nově navržené prvky a funkce hry.

- **Využití modelů chování agentů**
  Vytvořené modely chování nepřátelských agentů budou začleněny do hry jejím rozdělením na vícero úrovní, přičemž každá úroveň bude obsahovat jiný z dostupných modelů. Rozdělení do úrovní zároveň umožní více diverzifikovat vzhled prostředí ve hře a jejich plnění poskytne hráči satisfakci z dosaženého postupu.
- **Tutoriál**
  Jak již bylo zmíněno, tutoriál je velmi podstatnou částí, která hře dosud chyběla. Do hry proto bude přidán jakožto první úroveň, kterou hráč navštíví. V tutoriálu bude hráč řadou krátkých úkolů seznámen s ovládáním a mechanikami hry postupně od jednodušších po složitější.
- **Grafické provedení**
  Vzhled hry bude vylepšen přidáním nových vizuálních a částicových efektů. Budou též přidány nové textury a ilustrace jednotlivých úrovní.
- **Hudba a zvuky**
  Současná absence hudby a zvuků v prototypu hry je pro plnohodnotné vydání nepřijatelná. Do hry proto budou přidány zvuky a hudba, jež posílí její atmosféru a umocní hráčův zážitek. Kromě zvuků UI a prvků hry bude též posíleno vyprávění příběhu přidáním hlasu lodní umělé inteligence, která s hráčem komunikuje na jeho lodi před začátkem každé nové úrovně.
- **Persistence dat**
  Třída `PlayerPrefs`, která je v současnosti využívána pro zajištění persistence dat, umožňuje ukládat pouze jednoduché páry klíčů a hodnot. Tím znesnadňuje přidávání dalších vylepšení a má pouze omezené možnosti. Bylo proto rozhodnuto pro přechod na složitější, avšak výrazně flexibilnější metodu ukládání dat skrze jejich serializaci a uložení do souboru ve formátu JSON. Mimo vylepšení a krystalů budou nově ukládány i informace o hráčově postupu hrou.
- **Množství mechanik a obsahu**
  Hra bude rozšířena o mechaniku schopností a jejich příslušných vylepšení. Schopnosti pomohou hráči snadněji prozkoumávat herní mapy a lépe pozorovat chování nepřátelských agentů. Dále bude hra rozšířena již zmiňovaným rozdělením na úrovně. Každá úroveň pak bude obsahovat prvky specifické pouze pro ni.
- **Příběh**
  Do hry bude začleněn příběh v podobě krátkých scének před spuštěním každé úrovně. V nich bude hráči nastíněno, co je v dané úrovni jeho cílem a proč její navštívení potřebné pro dokončení hry. Zároveň bude hráči skrze příběh prezentován finální cíl, kterého má dosáhnout.

- **Ovladatelnost a klávesové zkratky**
  Hra bude rozšířena o klávesové zkratky pro výběr budov a schopností a také pro ovládání rychlosti herního času. Rovněž bude přidána možnost pohybovat kamerou držením prostředního tlačítka myši v kombinaci s pohybem myší.
- **Možnosti nastavení**
  Do hry bude přidána obrazovka pro změnu nastavení hlasitosti různých druhů zvuků a rovněž pro základní grafická nastavení, jako je rozlišení či přepnutí režimu celé obrazovky. Dá se předpokládat, že tato nastavení využije spíše menší podíl hráčů, avšak absence těchto nastavení v případě jejich potřeby by měla výrazně negativní vliv na zážitek hráčů, kteří je potřebují, a proto je přítomnost těchto možností nastavení velmi důležitá. Možnost přenastavit klávesové zkratky bylo rozhodnuto nepřidávat, jelikož je ovládání hry stále poměrně jednoduché.
- **Verze Unity**
  Prototyp hry byl v rámci bakalářské práce vyvíjen ve verzi Unity, které vypršela podpora v průběhu loňského roku. Dojde proto k přechodu na nejnovější dostupnou verzi Unity, kterou je v době psaní tohoto textu verze 2022.2.15f1.
- **Optimalizace**
  Pro účely optimalizace výkonu bylo zváženo využití Unity frameworku ECS, nakonec však bylo rozhodnuto tuto možnost nevyužít. Důvodem je především fakt, že hra se v současnosti nepotýká s výkonnostními problémy. Rozsáhlé a časově náročné změny potřebné pro využití tohoto frameworku by tak nebyly opodstatněné a jejich přínos diskutabilní. Náročnější výpočetní operace jsou v případě chování agentů navíc prováděny pouze jednou za čas, a jejich dopad na výkon je tak redukován rovnoměrným rozložením v čase. Využití ECS by naopak vyžadovalo provádění těchto operací ve stejný okamžik, aby bylo přínosné.

  Další zváženou možností optimalizace bylo využití návrhového vzoru *Object Pool*. Tato možnost rovněž v současnosti nebude využita vzhledem k absenci problémů s výkonem. Její využití je však považováno za vhodné v případě, že by v pozdější části vývoje či testování problémy s výkonem vyvstaly.
- **Lokalizace**
  Z analýzy person vyplynulo, že absence české lokalizace nehraje pro cílové uživatele významnou roli. Z tohoto důvodu nebude podpora lokalizace v současnosti implementována.

- **Herní obtížnost a vyváženost**
  Vyváženosti hry bude při vývoji v porovnáním s prototypem věnována zvýšená pozornost a stane se jedním z předmětů závěrečného testování.
- **Testování**
  Bude uspořádáno kvantitativní uživatelské testování, při kterém bude osloveno větší množství potenciálních hráčů s možností zahrát si vyvinutou hru a poskytnout zpětnou vazbu formou dotazníku. Testována bude obtížnost jednotlivých úrovní, srozumitelnost vysvětlení konceptů v tutoriálu a spokojenost hráčů s různými aspekty hry. Hráčům bude též poskytnut prostor pro nahlášení jakýchkoliv nalezených chyb či návrhů na další vylepšení hry. Rovněž budou sbírány informace, které pomohou určit, zda jsou respondenti součástí cílové skupiny či nikoliv.

4.2 Uživatelské rozhraní

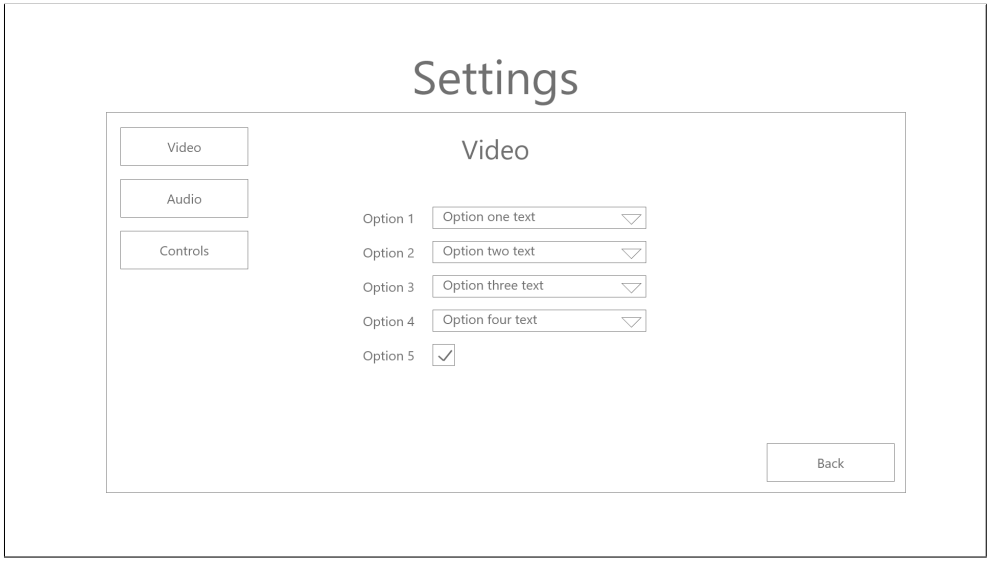
Tato sekce se zabývá návrhem nové verze uživatelského rozhraní pro Rogue Planet. Je třeba, aby návrh rozhraní podporoval případy užití identifikované v části 3.5. Taktéž by měl reagovat na problémy nalezené heuristickou analýzou popsané v části 3.6. Dodržovat bude již zavedený grafický styl popsaný v části 3.1.2.

Součástí této kapitoly je poměrně velké množství wireframů znázorňujících nově navrhovanou podobu rozhraní a obrázků vzhledu rozhraní v původním prototypu z bakalářské práce [19] . Tyto obrázky byly kvůli jejich počtu a velikosti pro přehlednost z velké části přesunuty do samostatné přílohy B.

4.2.1 Hlavní menu

První obrazovkou, kterou hráč po spuštění hry vidí, je hlavní menu. Podobu hlavního menu v prototypu hry si můžete prohlédnout na obrázku B.1. Menu je poměrně jednoduché a hlavní změny spočívají v jeho dalším zjednodušení a v reorganizaci jeho podobrazovek a jim příslušných tlačítek. Byla odstraněna obrazovka *How to play*, jejíž funkce bude nahrazena uživatelsky přívětivějším tutoriálem přímo ve hře. Rovněž byla odstraněna obrazovka s výběrem ukázek chování agentů, která je po začlenění vícero chování do hlavního herního módu nadbytečná. Totéž platí i pro obrazovku s titulky, která byla rovněž začleněna do hlavního herního módu.

Pro lepší podporu případů užití *UC1 Začít novou hru* a *UC2 Pokračovat v dříve uložené hře* byla tlačítka *play* a *reset progress* nahrazena kombinací tlačítek *new game* a *continue*, která by uživatelům měla být známá z dalších her. Problém s nejasností přítomnosti uložené hry, který byl nalezen heuristickou analýzou, bude vyřešen signalizací skrze stav tlačítka *continue*, které bude dostupné pouze při existenci uložené hry. Návrh nového rozložení tlačítek si můžete prohlédnout na obrázku B.2.



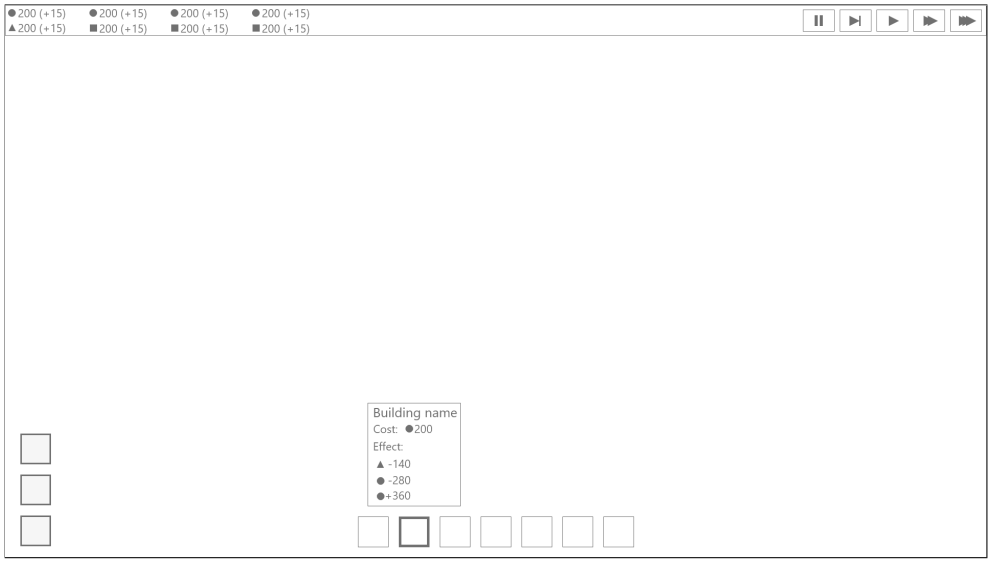
Obrázek 4.1: Návrh obrazovky s nastavením

Hlavní menu bylo dále rozšířeno o obrazovku s nastavením hry a tlačítkem *settings* pro navigaci na tuto obrazovku. Navržené rozložení této obrazovky naleznete na obrázku 4.1. Obrazovka s nastaveními je dělena do tří sekcí pro nastavení obrazu, hlasitosti různých druhů zvuků a zobrazení ovládání hry. Na levé straně obsahuje tlačítka pro navigaci mezi těmito sekcemi.

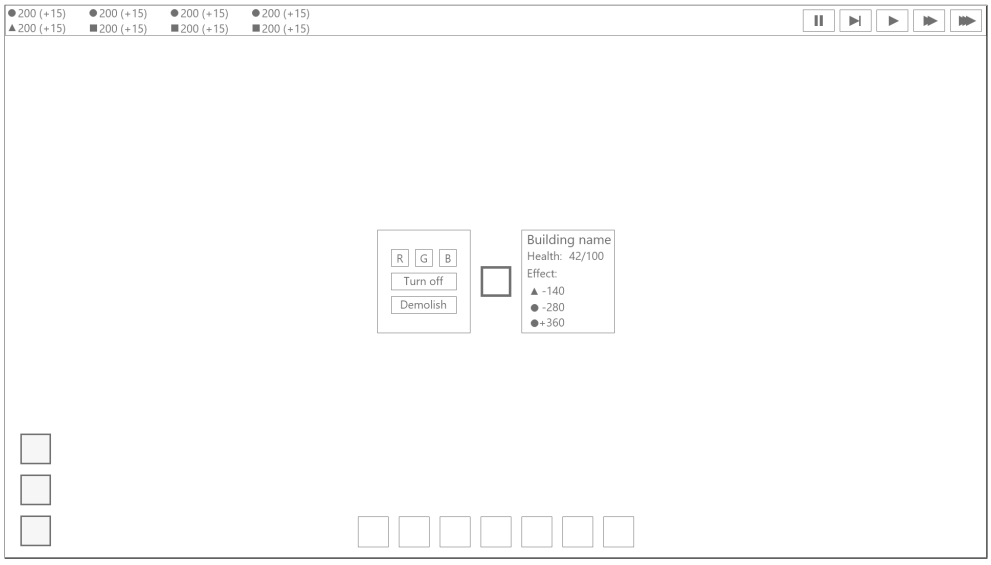
4.2.2 Rozhraní v průběhu úrovně

Nejobsáhlejší částí uživatelského rozhraní je rozhraní, které uživatel vidí přímo ve hře, které se skládá z vícero funkčních částí. Tato část rozhraní je zodpovědná za největší podíl případů užití z popsaných v sekci 3.5. Jeho základní podobu si můžete prohlédnout na obrázku B.5.

Stavění budov a používání schopností
Pro výběr budov k postavení či schopností k použití jsou určena tlačítka ve spodní, respektive levé spodní části obrazovky. Tlačítka pro výběr budov byla oproti předchozí podobě přesunuta do středu, zatímco tlačítka pro výběr schopností jsou zcela nová. Po přesunutí kurzoru nad tlačítko se zobrazí informace o relevantní budově či schopnosti, což je změnou oproti původní podobě, kdy bylo nutné pro získání informací na tlačítko kliknout. Ukázku zobrazení informací o budově si můžete prohlédnout na obrázku 4.2. Kromě krátkého popisu funkce obsahuje náhled také cenu budovy, či dobu trvání, než bude možné schopnost znovu použít.



Obrázek 4.2: Návrh rozhraní v hlavním herním režimu – informace o budově



Obrázek 4.3: Návrh rozhraní v hlavním herním režimu – interakce s budovou

Po zvolení budovy či schopnosti rozhraní nově obsahuje náhled, který se zobrazí pod hráčovým kurzorem na herní mapě v místě, kde bude schopnost použita či budova postavena. Náhled obsahuje též dosah zvolené budovy či schopnosti. Absence tohoto znázornění dosahu budov byla v heuristické analýze identifikována jako jeden ze dvou nejvíce závažných nedostatků původního rozhraní hry. Návrh podoby náhledu viz obrázek B.9 a původní vzhled rozhraní při stavění budovy pro srovnání viz obrázek B.10.

Rozhraní již postavené budovy
Po kliknutí na již postavenou budovu na herní mapě se hráči zobrazí informace a kontrolní panel vztahující se k této budově. Panel obsahuje informace o dané budově jako je její aktuální stav a produkce a také tlačítka pro interakci s danou budovou. Informace se zobrazí přímo u vybrané budovy čímž je jasně znázorněno, ke které budově se tyto informace váží. Hráč tak v porovnání se starším provedením nemusí pro získání informací zrakem putovat do spodní části obrazovky. Informační panel zároveň obsahuje i tlačítka pro různé interakce s budovou, jako je vypnutí budovy, přepnutí produkce na jinou barvu rudy nebo demolice této budovy. Díky přesunutí panelu přímo k budově jsou tato tlačítka rovněž snáze dostupná pro rychlou interakci. Návrh informačního panelu budovy si můžete prohlédnout na obrázku 4.3.

Na obrázku B.10 pak můžete vidět pro porovnání původní podobu informačního panelu v bakalářské práci.

Přehled surovin
Tabulka s přehledem stavu a produkce surovin byla pro lepší přehlednost přeskupena tak, aby rudy a krystaly stejné barvy byly pod sebou. Zkratky surovin byly nahrazeny ikonami. Produkce byla místo samostatného sloupce umístěna přímo k stavu dané suroviny a doplněna o znaménko plus či mínus.

Menu pozastavené hry
Jednoduchý vzhled menu při pozastavené hře je téměř shodný s předchozí verzí, pouze tlačítko pro zobrazení stručného návodu ke hře bylo nahrazeno tlačítkem pro zobrazení obrazovky s nastavením, viz obrázek B.11.

Závěrečné shrnutí
Vzhled panelu informujícího hráče o odnesených krystalech po opuštění úrovně zůstal zachován. Pouze v případě tutoriálu byl text panelu upraven tak, aby byl reflektován fakt, že z tutoriálu si hráč žádné krystaly odnést nemůže.

Tutoriál
Základem rozhraní v tutoriálu je rozhraní běžného herního režimu, které je obohaceno o panel s hráčovým aktuálním úkolem, viz obrázek B.12. Panel obsahuje krátký popisek a také body, které musí hráč splnit, aby dokončil daný

úkol. Po splnění všech bodů se zobrazí text vyzývající hráče, aby kliknutím na panel pokračoval k dalšímu úkolu. Hráč je jednotlivými úkoly postupně obeznámen s mechanikami hry a proveden tutoriálem.

Spolu s novými úkoly jsou hráči též odhalována jednotlivá tlačítka pro výběr budov a schopností tak, aby hráč nebyl na počátku zahlcen mnoha dostupnými možnostmi a byl s nimi seznámen postupně.

4.2.3 Rozhraní mezi úrovněmi

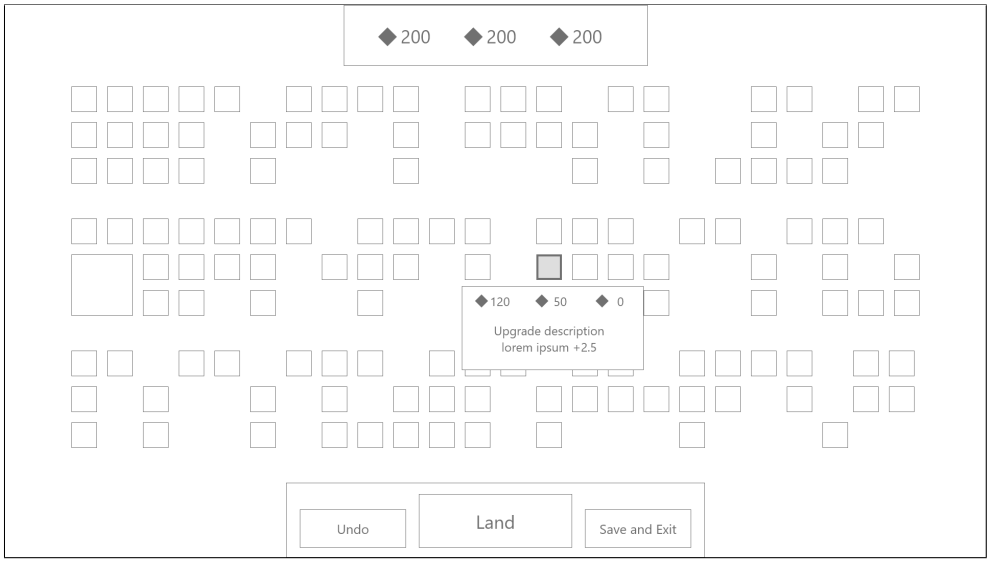
Rozhraní mezi přistáními sestává z obrazovky pro zakoupení vylepšení budov a schopností a nově též z obrazovky pro výběr úrovně. Jeho součástí jsou také scény prezentující příběh před začátkem jednotlivých úrovní.

Vylepšení
Obrazovka s vylepšeními nově vyobrazuje všechna vylepšení najednou. Tato vylepšení jsou uspořádána horizontálně dle příslušnosti k budovám či schopnostem a vertikálně dle jejich ceny, viz obrázek B.14. Informace o jednotlivých vylepšeních jsou pro přehlednost skryty a zobrazí se při přesunutí kurzoru nad dané tlačítko vylepšení, viz obrázek 4.4. Je důležité, aby tlačítka byla rozlišena dle toho, zda si hráč příslušné vylepšení může či nemůže dovolit. V podrobných informacích při najetí kurzorem na tlačítko by pak mělo být zvýrazněno, které z potřebných krystalů hráči chybí.

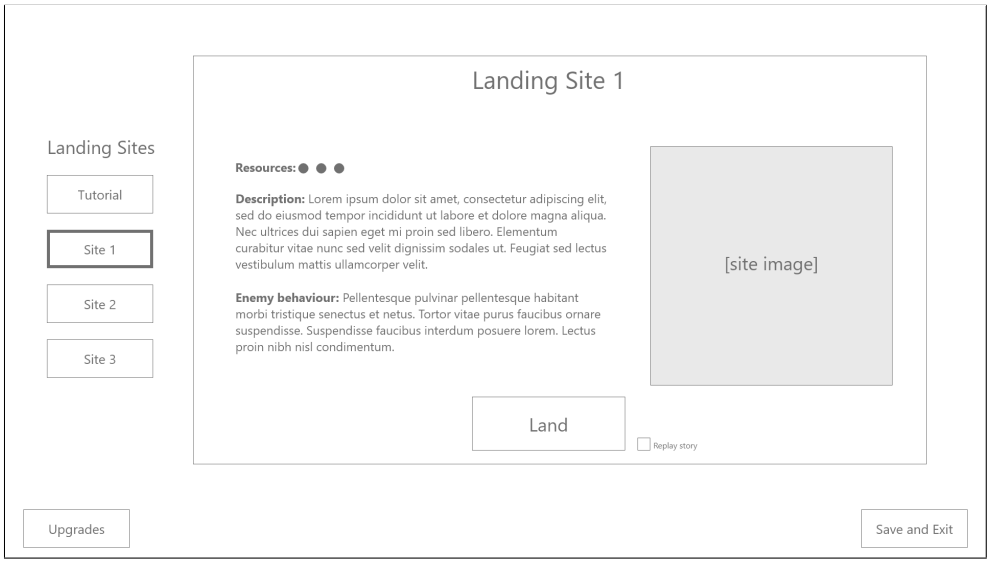
Obrazovka s vylepšeními je nově doplněna o tlačítko *undo* pro vrácení provedeného nákupu zpět. Absence možnosti zrušení nákupu vylepšení je druhým ze dvou nejvíce závažných nedostatků identifikovaných v heuristické analýze. Původní vzhled obrazovky s vylepšeními můžete vidět na obrázku B.13.

Výběr úrovně
Zcela novou částí UI je obrazovka pro výběr úrovně, jejíž návrh můžete vidět na obrázku 4.5. Obsahuje jeden panel pro každou úroveň, přičemž hráč může mezi jednotlivými panely přepínat tlačítky na levé straně. Každý panel obsahuje název úrovně, seznam rud v této úrovni, krátký popis této úrovně a odůvodnění proč ji hráč musí navštívit, popis chování nepřátel v této úrovni, náhledový obrázek a tlačítko pro přistání. Před prvním přistáním v každé úrovni je hráči odprezentována část příběhu. V případě zájmu si hráč může nechat příslušnou část příběhu zopakovat zaškrtnutím políčka vedle přistávacího tlačítka. Informace o chování nepřátelských agentů v úrovni by hráči měla být zpřístupněna až poté, co hráč v této úrovni alespoň jednou přistane.

Příběh
Hráči bude příběh hry prezentován jednoduchými monologovými scénami, které se skládají z velkého obrázku prostředí úrovně a spodního panelu s textem. Jejich návrh můžete vidět na obrázku B.17. V pravé části textového



Obrázek 4.4: Návrh obrazovky s vylepšeními – informace o vylepšení



Obrázek 4.5: Návrh obrazovky pro výběr úrovně

panelu se nachází obrázek osoby, která k hráči promlouvá. Text je prezentován postupně po menších úsecích, přičemž každý úsek musí hráč odkliknout myší či tlačítkem na klávesnici.

Po dohrání poslední úrovně bude hráči odprezentována závěrečná část příběhu, po které bude následovat jednoduchá obrazovka s titulky viz B.18.

4.3 Budovy

Tato část se zaměřuje na skripty dodávající funkcionalitu budovám ve hře. Základem každé z budov je objekt s jménem dané budovy. Různé součásti budovy jsou pak rozloženy do tří potomků tohoto objektu:

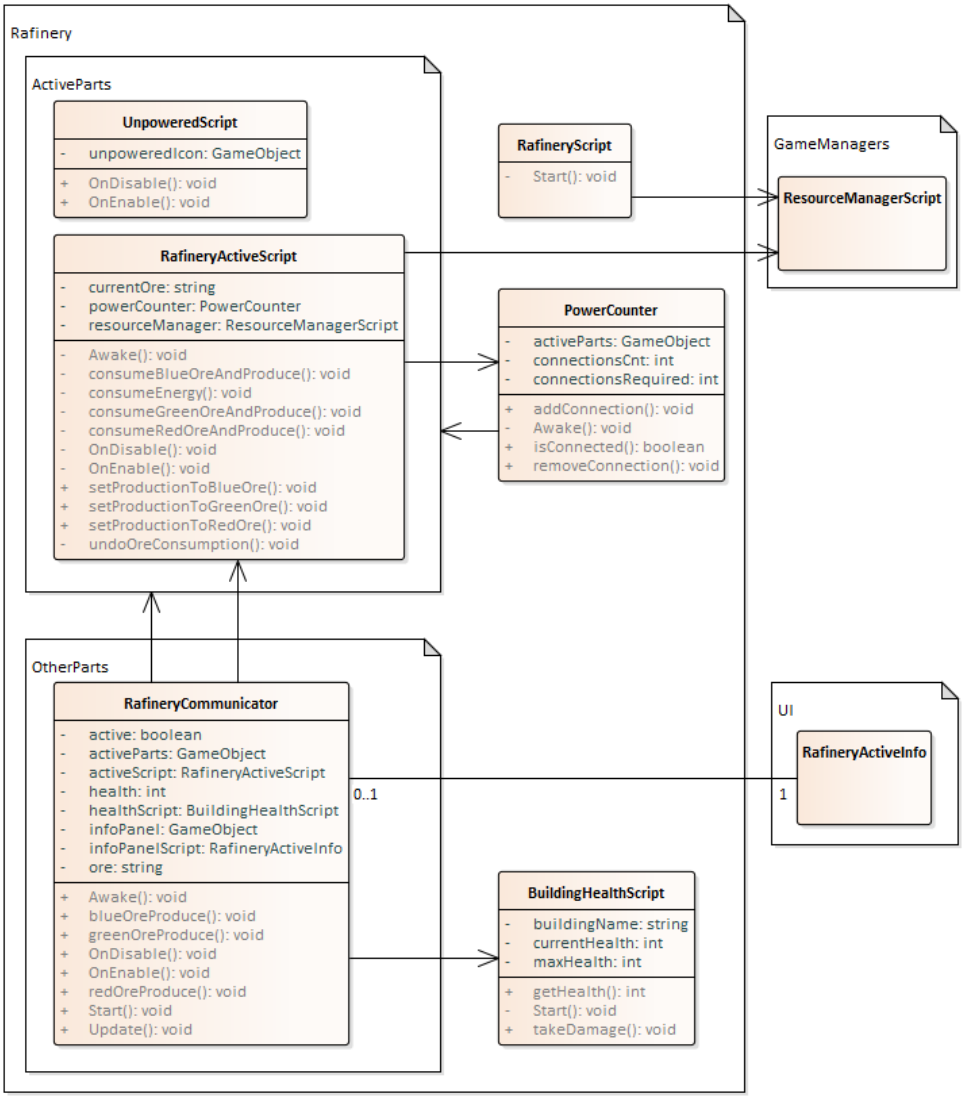
- `PassiveParts` obsahující součásti budovy, které jsou vždy aktivní, jako je renderer jejího vzhledu.
- `ActiveParts` pro součásti funkční jen když je budova spuštěna, tedy například skript pro generaci a spotřebu surovin či světlo poskytované budovou.
- `OtherParts` se součástmi, které jsou aktivní jen příležitostně, avšak nikoli v závislosti na spuštění budovy.

Tato struktura byla stanovena v prototypu hry a bude ponechána i pro plnou verzi. Vždy aktivní skripty obvykle na rozdíl od jiných typů komponent nejsou umístěny na objektu `PassiveParts`, ale pro snazší přístup zvenčí jsou součástí přímo hlavního objektu budovy.

Každá postavená budova mimo přistávacího modulu může být buď zapnutá, kdy pravidelně spotřebovává suroviny a provádí aktivně svoji funkci (světlo svítí, kulomet střílí atd.), nebo vypnutá, kdy je stále umístěna na herní mapě, avšak neprovádí žádnou akci a nespotřebovává ani nevytváří suroviny. Pro změny chování budovy v závislosti na těchto dvou stavech je využita schopnost Unity aktivovat a deaktivovat herní objekty, čímž je zároveň přerušena funkce na nich umístěných skriptů. Deaktivace budovy je prováděna deaktivací objektu `ActiveParts`.

Uspořádání a nedostatky skriptů pro budovy v prototypu hry a jejich plánovaná vylepšená struktura budou demonstrovány na příkladu rafinerie. Diagram tříd tvořících rafinerii v prototypu hry si můžete prohlédnout na obrázku 4.6. Příslušnost tříd k objektům (v terminologii Unity `GameObject`) umístěných v hierarchii scény je znázorněna poznámkami.

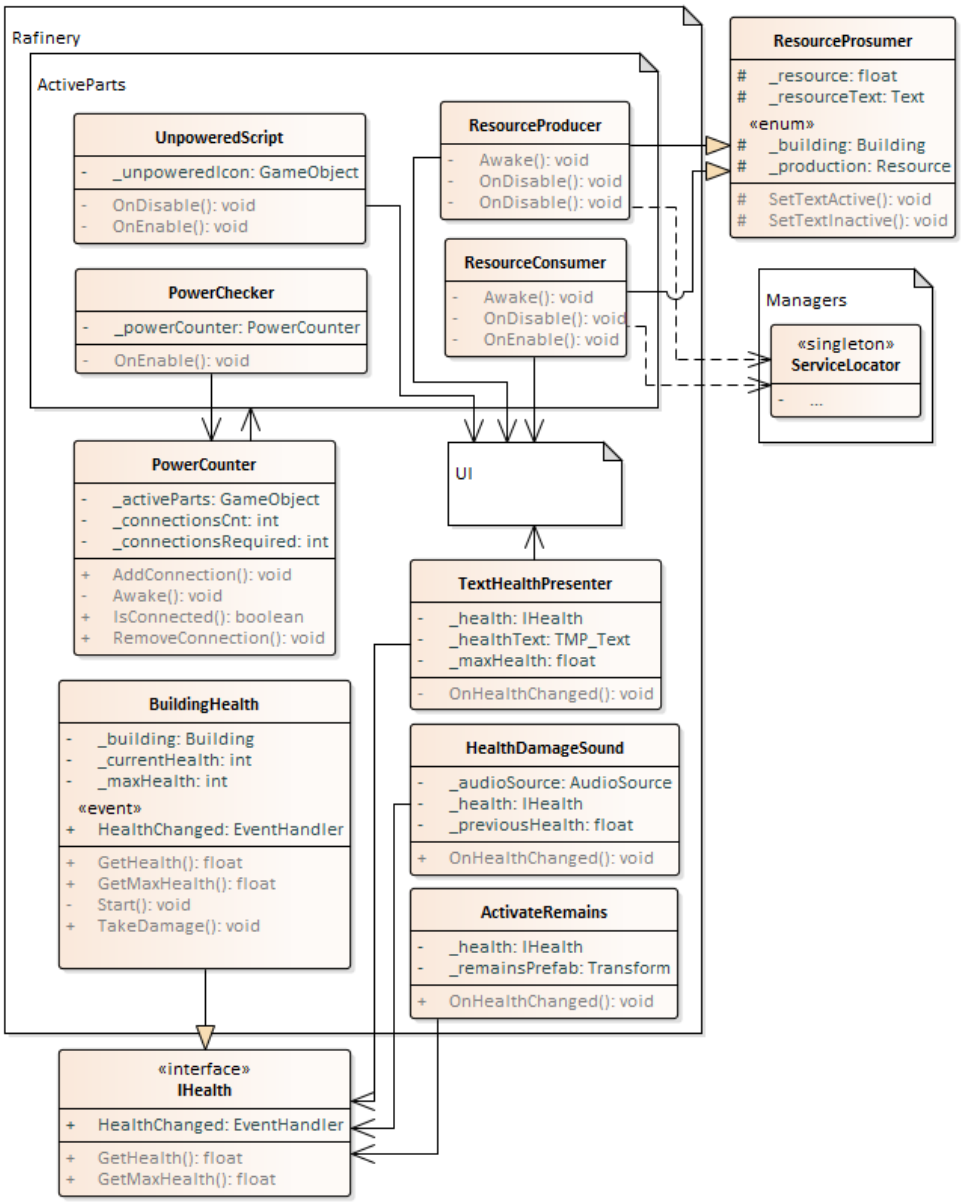
Valná většina chování budovy z pohledu herních mechanik byla v prototypu koncentrována do třídy `RafineryActiveScript`. Každá budova měla vlastní třídu `ActiveScript` pojmenovanou po dané budově, tedy existovalo celkem osm těchto tříd. Toto snadné a rychlé řešení vložením všeho chování budovy do dlouhé třídy je výrazným porušením principu SoC a prvních dvou principů SOLID. V kombinaci se samostatnou třídou pro každou budovu pak docházelo často ke zbytečným duplicitám a lineárnímu růstu počtu těchto tříd



Obrázek 4.6: Diagram tříd pro rafinerii v prototypu hry

s počtem dostupných budov. V nové verzi hry je proto chování třeba rozdělit nikoliv dle budov, ale dle funkcionalit. Obsluhu prvků UI, která je v prototypu koncentrována do jedné třídy `RafineryCommunicator`, pak bude rozdělena obdobně. Třídy `Communicator` existovalo v původní verzi stejně jako v případě `ActiveScript` osm variant, pro každou budovu jedna.

Nové rozložení tříd v rámci budovy rafinerie můžete vidět na obrázku 4.7. Dříve přítomné velké třídy pro chování budovy a její UI jsou přeorganizovány dle funkcionalit do menších tříd. Z třídy `RafineryScript` a jejích variant zodpovídajících za zaplacení postavené budovy a nastavení jejích vlastností dle vylepšení je placení budovy vyčleněno mimo samotnou budovu, což v případě



Obrázek 4.7: Diagram tříd pro rafinerii v nové podobě

rafinerie vede k možnosti odstranit původní třídu, jelikož neobsahovala žádné další chování.

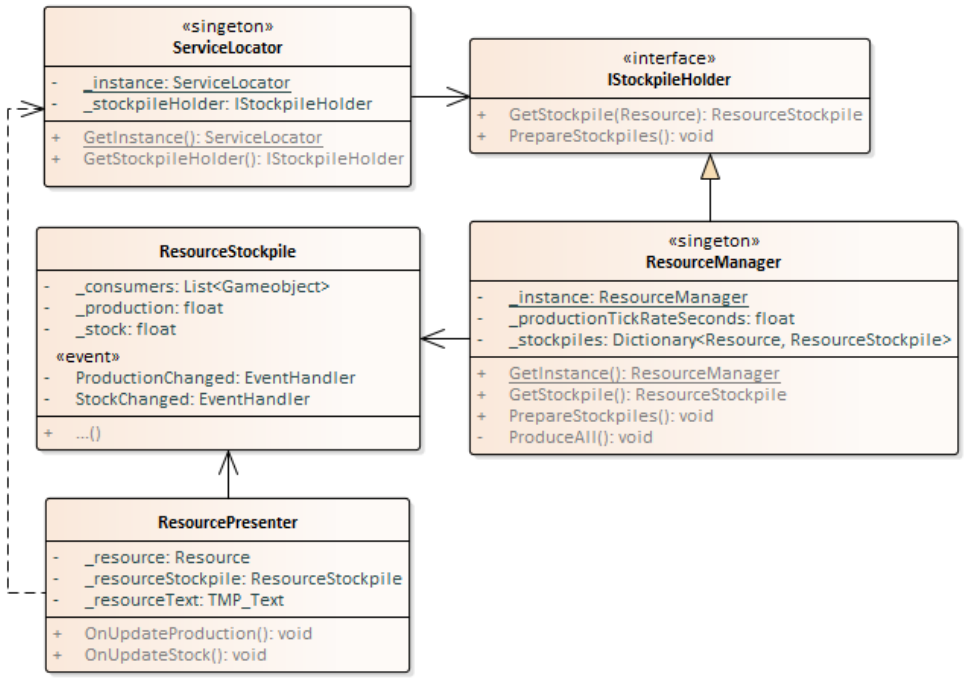
Ve vhodných případech, jako je zdraví budovy a produkce se spotřebou surovin, bude dále provedeno rozdělení příslušných tříd dle strukturálního vzoru *MVP*. V případě zdraví budovy je *modelem* třída `BuildingHealth`, *prezentérem* je třída `TextHealthPresenter` a *view* je zastoupen UI komponentami, které poskytuje Unity. Propojení skrze vzor *Observer* implementovaný `C#` událostmi umožňuje snadno zdraví prezentovat vícero způsoby a případně na jeho změny reagovat v dalších komponentách bez nadbytečných závislostí či potřeby měnit kód zodpovědný za zdraví budovy. Použití událostí je zároveň šetrnější z pohledu výkonu v porovnání s dříve užívaným neustálým kontrolováním změn v herní smyčce. Zdraví je zároveň zobecněno rozhraním `IHealth`, díky kterému je možné komponenty reagující na změny zdraví využít i jinde než pouze u budov.

Další budovy jsou pak tvořeny znovupoužitím výše popsaných komponent a případně doplněny komponentami zprostředkovávajícími chování unikátní pro tyto budovy. Příkladem jedné z takovýchto nových komponent je třída `PowerGrid` pro rozvodny energie a přistávací modul, která s pomocí algoritmu DFS detekuje, zda rozvodny tvoří souvislý graf. V případě že tomu tak není, jsou všechny rozvody energie, které nejsou součástí komponenty souvislosti, jež obsahuje přistávací modul, vypnuty. Tím je vyřešena známá chyba z prototypu, kdy se dvě či více rozvoden dokázalo napájet navzájem, ačkoli nebyly připojeny k základně.

4.4 Správa surovin

V prototypu hry byla správa surovin zajišťována poměrně obsáhlou třídou `ResourceManagerScript` a samotnými budovami, které skrze statické metody této třídy produkovaly a konzumovaly suroviny a v případě nedostatku surovin se samy vypnuly. Tato třída obsahovala samostatnou metodu pro každou ze surovin ve hře, a kvůli tomu se v ní nacházelo mnoho duplicit a byla velmi obtížně udržovatelná. Zároveň musel v každé budově běžet samostatný cyklus pro produkci a spotřebu surovin, což zbytečně zvyšovalo nároky na výkon s narůstajícím počtem budov ve scéně.

V návrhu nové verze hry je správa surovin (obrázek 4.8) rozdělena mezi třídu `ResourceManager` a třídu `ResourceStockpile`. Zásobu jednoho druhu suroviny reprezentuje třída `ResourceStockpile`, jež nabízí funkce k manipulaci s ní a registraci producentů a konzumentů. Dále má na starosti deaktivaci konzumentů při nedostatku spravované suroviny a poskytuje též události vyvolávané při změně množství suroviny či její produkce. Tyto události umožňují nejen snadno prezentovat stav daných surovin v uživatelském rozhraní, ale rovněž na ně můžou reagovat i další související prvky rozhraní. Například je možné tlačítko pro zakoupení vylepšení aktivovat či deaktivovat na základě



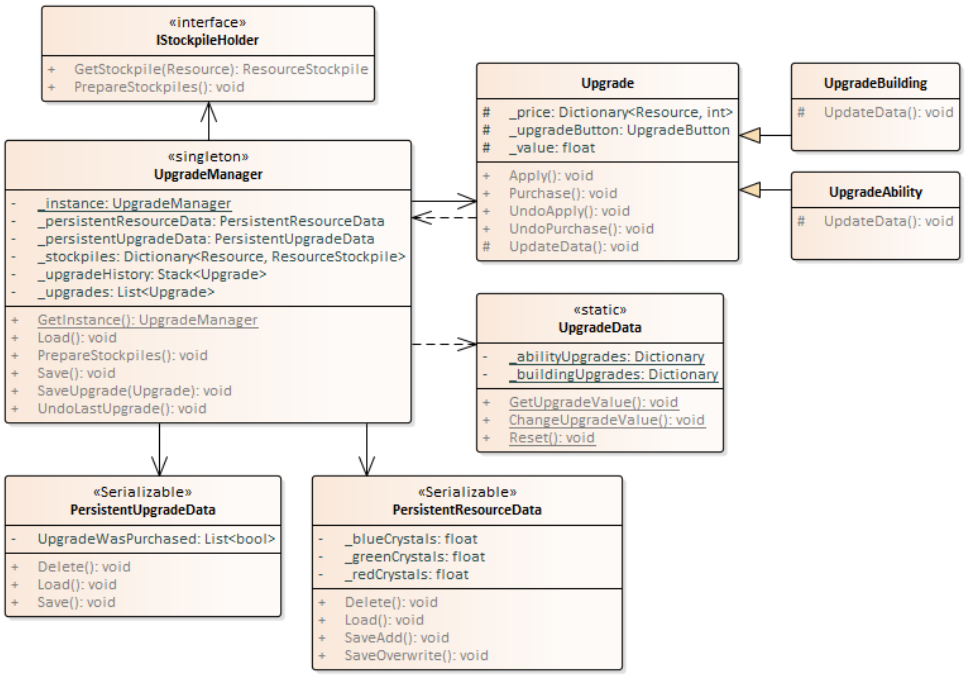
Obrázek 4.8: Diagram tříd pro správu surovin v nové podobě

toho, zda máme dostatek potřebných surovin, aniž by bylo třeba tento stav aktivně kontrolovat ze strany tlačítka.

Třída `ResourceManager` implementující návrhový vzor *Singleton* spravuje zásoby surovin pro danou úroveň a zajišťuje cyklus produkce a spotřeby surovin v závislosti na uplynulém čase. Rozhraní `IStockpileHolder` reprezentuje obecného správce zásob surovin. Využití tohoto rozhraní umožňuje třídu `ResourceManager` snadno nahradit jinou třídou implementující totéž rozhraní v situacích, kdy je vyžadována odlišná logika práce se surovinami. Globální přístup k aktuálnímu správci surovin nezávisle na jeho aktuální podobě poskytuje třída `ServiceLocator`, která je provedením návrhového vzoru *Service Locator*. Za reprezentaci stavu a produkce suroviny v uživatelském rozhraní je pak zodpovědná třída `ResourcePresenter`.

4.5 Vylepšení a persistence dat

Další velkou softwarovou změnou oproti prototypu je kompletní přepracování systému vylepšení budov a schopností. Navrhovanou strukturu tohoto systému můžete vidět na obrázku 4.9. Zakoupení vylepšení je v novém návrhu reprezentováno třídou `Upgrade`, respektive jejími potomky `UpgradeBuilding` a `UpgradeAbility`. Tyto třídy představují návrhový vzor *Command* a implementují metody pro zakoupení vylepšení i případné zrušení tohoto zakoupení.



Obrázek 4.9: Diagram tříd pro vylepšení v nové podobě

O správu zakoupených vylepšení se stará třída `UpgradeManager`, která pomocí zásobníku udržuje historii nákupů a umožňuje provedené nákupy ve zpětném pořadí rušit.

Persistence pořízených vylepšení a krystalů potřebných k jejich kupování je nově místo využití třídy `PlayerPrefs` zajištěna serializací dat a uložením do souboru ve formátu JSON. Zatímco třída `PlayerPrefs` poskytovaná Unity umožňuje ukládat pouze jednotlivé dvojice klíče a hodnoty typu *int*, *float* a *string*, serializace do formátu JSON umožňuje ukládat libovolné primitivní datové typy, a především i jejich pole.

Kromě širších možností ukládání nabízí též formát JSON možnost obfuskace ukládaných dat pro znesnadnění jejich úpravy ze strany uživatele. Různé druhy dat lze pak rozdělit do samostatných souborů, a tím umožnit budoucí změny struktury částí těchto dat, aniž by byla ovlivněna ostatní ukládaná data. Uvolnění třídy `PlayerPrefs` pak umožní využít ji k jejímu primárnímu účelu, kterým je ukládání změn nastavení zadaných uživatelem.

Pro uložení dat do formátu JSON je třeba data vložit do specializovaných tříd opatřených atributem *Serializable*. Pro poskytnutí snadného přístupu budovám a schopnostem jsou data o vylepšeních zapisována rovněž do statické třídy `UpgradeData`. Krom dat o vylepšeních a krystalech budou dále ukládána taktéž data o hráčově postupu hrou. Ta umožní například rozpoznat, zda je

vhodné hráči před spuštěním úrovně spustit její příběhový úvod, protože ji spouští poprvé.

4.6 Nepřátelé

Základem nové struktury nepřátelských agentů ve hře je třída `EnemyHealth`, která bude podobně jako třída pro zdraví budov implementovat rozhraní `IHealth`. To umožní nepřátelské agenty dále rozšířit komponentami pro zvuky při poškození `HealthDamageSound` a pro zobrazení mrtvoly po smrti agenta `ActivateRemains`. Tyto komponenty jste již mohli vidět v návrhu struktury budov na obrázku 4.7. Na rozdíl od budov nebudou životy agentů prezentovány textovou formou s využitím třídy `TextHealthPresenter`. Tato třída bude nahrazena třídou `SpriteHealthPresenter`, která představuje alternativní metodu prezentace životů skrze změny v textuře agenta.

Agenti se dále skládají z komponent zprostředkovávajících jejich chování vytvořených již v rámci bakalářské práce [19] a z komponent pro pohyb prostorem, které jsou součástí balíčku *A* Pathfinding Project* [26], taktéž ponechaných již z prototypu hry. Skripty pro chování agentů vnitřně obsahují konečný automat, bylo proto zváženo jejich přepsání dle návrhového vzoru *State*. Nakonec tato možnost nebyla zvolena. Důvodem je použití relativně malého množství stavů, jejichž počet se již dále nebude měnit. Stavy navíc pracují s poměrně vysokým množstvím společných informací, které by v případě použití vzoru *State* bylo mezi jednotlivými stavy nutné složitě předávat, což by mělo výrazně negativní vliv na přínos využití tohoto vzoru z pohledu čitelnosti kódu a využití vzoru by tím pádem bylo kontraproduktivní.

Agenti v jednotlivých úrovních se od sebe liší nastavenými hodnotami komponenty `EnemyHealth` a výběrem komponenty pro jejich chování. Komponenta pro zdraví agentů umožňuje nastavit nejen hodnotu jejich základního zdraví, ale taktéž míru, jakou jejich počáteční zdraví roste v závislosti na tom, kdy se konkrétní agent zrodil. To umožňuje s postupem času zvyšovat obtížnost úrovně a tím donutit hráče, aby se v úrovni v rámci jednoho přistání nezdržoval příliš dlouho. Pro tutoriál byli zvoleni agenti bez jakéhokoli chování, kteří mají sloužit pouze jako cvičné cíle. V první plnohodnotné úrovni bude využito včelích agentů s vyváženým chováním prioritizujícím podobně sběr surovin i útočení na hráče. V druhé úrovni bude využito mravenčích agentů s vysokou komunikační vzdáleností vedoucí k tvorbě pravidelných cestiček spojujících suroviny, které hráč může pozorovat. Ve třetí úrovni pak budou použiti kooperující agenti s agresivním nastavením. Pořadí bylo zvoleno dle subjektivní obtížnosti jednotlivých modelů chování.

Kapitola 5

Realizace

Tato kapitola popisuje všechny implementované části vyvíjené hry kromě částí, které byly přidány až v reakci na výsledky závěrečného testování. Pořadí ve kterém jsou popisovány bylo zvoleno dle toho, v jakém pořadí se s nimi hráč ve hře přibližně setká.

Autorem vytvořené C# třídy jsou v Unity označovány jako skripty a do hry jsou přidávány v podobě komponent uvnitř herních objektů. Slova *třída*, *skript* a *komponenta* jsou proto v rámci této kapitoly užívány zaměnitelně.

5.1 Příběh a jeho prezentace

První část hry, se kterou se hráč po spuštění nové hry z hlavního menu setká, je úvodní příběhová scéna před spuštěním tutoriálu. Příběh hry, jehož krátké shrnutí si můžete přečíst v části 3.3.2, je hráči prezentován v podobě pěti krátkých monologových scén. Čtyři jsou spuštěny před prvním navštívením každé úrovně, pátá pak po úspěšném odnesení artefaktů z poslední úrovně. Po poslední příběhové scéně následuje obrazovka s titulky.

Každá ze scén obsahuje ilustraci pozadí reprezentující příslušnou úroveň a portrét umělé inteligence, která na hráče promlouvá. Tyto ilustrace a portrét byly generovány s pomocí modelu hlubokého učení Stable Diffusion[3], který umí vytvářet obrázky na základě textového vstupu. Po drobných úpravách pak byly obrázky převedeny do pixel artové podoby softwarem Pixelator[4]. Ukázku výsledného vzhledu jedné z příběhových scén můžete vidět na obrázku 5.1. Aby scény nepůsobily staticky, byla vytvořena jednoduchá komponenta jménem `SmoothBackgroundZoom`, která zajišťuje plynulý přechod mezi dvěma různými velikostmi pozadí a navozuje tak pocit přibližování či oddalování.

Základem každé scény je třída `StoryText`, která je zodpovědná za prezentování textu po krátkých úsecích. Text se v rámci každého úseku objevuje

[3] `https://stability.ai/blog/stable-diffusion-public-release`
[4] `http://pixelatorapp.com/`

Obrázek 5.1: Ukázka příběhové scény

postupně po jednotlivých písmenech, čímž je hráčův zrak upoután na postupně se odvíjející text a zároveň je tímto způsobem naznačeno, že text reprezentuje mluvenou řeč, která se taktéž odvíjí postupně. Aby byla prezentace textu plynulá, jsou přeskakovány mezery a speciální znaky. Spolu s každým textovým úsekem je zároveň spuštěna syntetizovaná hlasová nahrávka daného textu (více viz část 5.6.3). Postupné odhalování textu může hráč přeskočit mezerníkem či levým tlačítkem myši. Důležité informace jsou v textu vyznačeny odlišnými barvami. Například mimozemský artefakt, který se hráč snaží získat, je označen světle žlutou barvou, zatímco nepřátelé jsou označeni jasně červenou barvou.

5.2 Tutoriál

Po úvodní příběhové scéně následuje tutoriál, který má hráče seznámit se základními koncepty a mechanikami hry. Tutoriál se skládá celkem z dvanácti úkolů reprezentovaných třídou `Quest` s textovým panelem na pravé straně obrazovky. Ukázku jednoho z nich můžete vidět na obrázku 5.2. Každý úkol se pak skládá z jedné či více drobných úloh reprezentovaných třídou `Task`. Velká rozmanitost úloh je řešená dědičností, přičemž potomci se liší v tom, jakým způsobem kontrolují své splnění. Třída `Task` tak mimo jiné obsahuje potomky kontrolující postavení budovy na určitém místě, stav produkce surovin, zabití nepřátel v určité oblasti, či pohyb kamery. Splněním úlohy se barva jejího textu změní na sytě zelenou. Po splnění všech úloh daného úkolu může hráč kliknutím přejít na další úkol. V průběhu tutoriálu je hráč seznámen s následujícími koncepty:

Obrázek 5.2: Panel jednoho z úkolů v tutoriálu

1. **Pohyb**
   Hráči jsou krátce představeny možnosti ovládání kamery. Pro splnění úkolu musí hráč pohnout kamerou a přiblížit či oddálit obraz, aby si ovládání vlastnoručně ozkoušel.
2. **Stavební materiály**
   Hráči je odemčena první z budov, kterou je těžební vrták, a je mu dáno za úkol postavit vrtáky na dvě dostupná ložiska bílé rudy. Tím hráč získá zkušenost se stavěním budov a zároveň je seznámen s funkcionalitou budovy vrtáku, využitím bílé rudy a způsobem získávání surovin ve hře.
3. **Distribuce energie**
   Hráči je zpřístupněna budova rozvodny energie a odhalena nová část mapy, která obsahuje neaktivní budovy. Hráč má za úkol budovy zapojit postavením několika rozvoden, a tím se seznámit s touto budovou a mechanikou rozvádění energie budovám.
4. **Demolice budov**
   Hráč má za úkol zdemolovat předem připravenou budovu kulometu. Tím je seznámen s rozhraním pro interakci s budovami a možností demolovat nevhodně umístěné budovy.
5. **Produkce energie**
   Hráči je odemčena budova generátoru. Za úkol má postavit několik těžebních vrtáků a generátor a tím se seznámit s využitím barevné rudy pro generaci energie.
6. **Reflektor**
   Hráč má za úkol s pomocí nově dostupných budov reflektoru a kulometu zničit nepřítele na mapě. Tím se seznámí s funkcemi těchto dvou budov a tím, že nepřátelské agenty je třeba nasvítit, aby po nich kulomety mohly střílet.
7. **Optimalizace produkce**
   Hráč musí dosáhnout požadovaných úrovní produkce energie a červené

rudy tím, že vypne některé nepotřebné budovy. Tím je upozorněn na fakt, že budovy spotřebovávají energii, a že nepotřebné budovy lze skrze interaktivní panel pro úsporu energie deaktivovat.

8. **Bodové světlo**
   Hráč má zlikvidovat skupinu nepřátel s použitím nově odemčeného bodového světla a tím se naučit, že bodové světlo má sice oproti reflektoru menší dosah, ale dokáže osvítit více nepřátel najednou.

9. **Láva**
   Hráč je seznámen s alternativním způsobem získání energie skrze těžbu lávy, která byla odhalena zabitím nepřátel při plnění předchozího úkolu.

10. **Schopnosti**
    Pro seznámení s dostupnými schopnostmi je hráči zpřístupněna další část mapy, která obsahuje vícero nepřátel. Zároveň je deaktivováno pomocné globální světlo, které doteď hráči usnadňovalo orientaci na mapě. Hráč musí využít všechny tři schopnosti, aby nové cíle nalezl a zlikvidoval. Tím se seznámí s jednotlivými schopnostmi a mechanikou odhalování herní mapy.

11. **Krystaly**
    V předposledním úkolu je hráč seznámen s budovou rafinérie a má za úkol vyprodukovat malé množství krystalů každé barvy. Tím se kromě porozumění funkci rafinérie též naučí měnit barvu rudy, kterou rafinérie zpracovává.

12. **Dokončení úrovně**
    Posledním úkolem hráče je opustit herní mapu. Tím je seznámen se způsobem, jak dokončit úroveň a s faktem, že úkolem je z úrovně odnést barevné krystaly pro vylepšení. Splněním tohoto úkolu hráč zároveň tutoriál dokončí a je převeden na obrazovku s vylepšeními.

5.3 Uživatelské rozhraní

Jednotlivé části uživatelského rozhraní ve hře byly vypracovány dle wireframů vytvořených v části 4.2. V této sekci budou popsána především vylepšení a rozšíření oproti základní navržené struktuře. Nebudou zde popisovány obrazovky, jejichž jediné úpravy spočívaly ve změně struktury dle návrhu či drobných úpravách barev, jako je hlavní menu, menu při pozastavené hře, či informační obrazovka po dokončení úrovně. Rovněž v této sekci není popsána obrazovka s příběhem, které byla věnována pozornost v samostatné podkapitole 5.1 a rozhraní v tutoriálu, které je popsáno v podkapitole 5.2.

Obrázek 5.3: Ukázka rozhraní úrovně

5.3.1 Rozhraní v průběhu úrovně

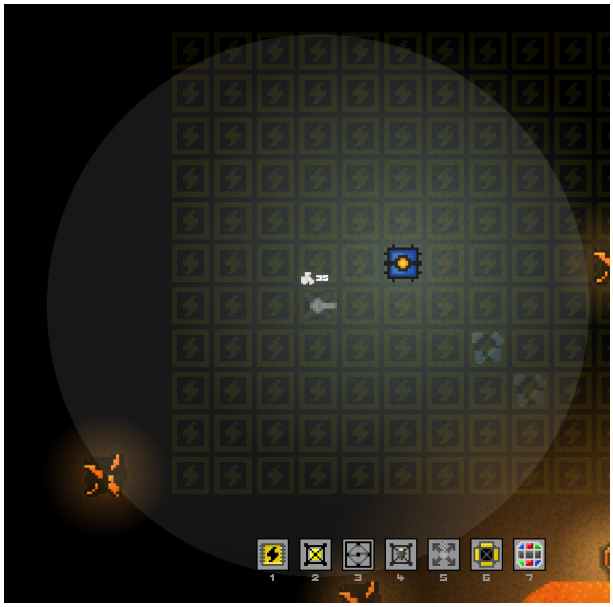
Suroviny

Ukázku vzhledu UI v průběhu hry můžete vidět na obrázku 5.3. Všechny texty rozhraní popisující suroviny mají jejich příslušnou barvu. Oproti prototypu byly zvoleny méně syté barvy, které jsou lépe čitelné na tmavém pozadí. Dříve používané zkratky pro názvy surovin byly nově nahrazeny autorem vytvořenými ikonami. Ve vhodných případech pak texty surovin mohou měnit barvu v závislosti na kontextu. Například když si hráč budovu nemůže dovolit, změní se barva její ceny na tmavě červenou. Totéž platí i v případě negativní produkce. Změny textu surovin v přehledové tabulce obstarává pro každou surovinu jedna instance třídy `ResourcePresenter`. Změny textů s cenami pak obstarává drobná komponenta `CostText`, která reaguje na události vyvolávané jí příslušející zásobou surovin.

Texty s produkcí a spotřebou surovin u budov disponují jednoduchým skriptem, který jejich hodnoty aktualizuje na začátku úrovně v závislosti na zakoupených vylepšeních. Totéž platí i pro texty popisující čas, po kterém je použitá schopnost opět dostupná. Budovy zpracovávající barevné suroviny byly vybaveny identifikátorem, se kterou barvou suroviny právě pracují, aby nebylo pro zjištění druhu produkce budov nutné na každou z nich klikat.

Náhledy

Každá budova disponuje náhledem vyobrazujícím její podobu na místě, kde bude postavena. U vhodných budov je rovněž vyobrazen jejich dosah. Náhled dosahu budov je aktualizován dle hráčových zakoupených vylepšení. Totéž platí i pro náhled dosahu schopností. Ukázku náhledu budovy kulometu si



Obrázek 5.4: Náhled kulometu při jeho stavění

můžete prohlédnout na obrázku 5.4. Tlačítko schopnosti je po jejím použití zakryto posuvníkem, který značí dobu, než bude schopnost znovu dostupná k použití.

Interakce s budovami
Každá budova disponuje vlastním interakční panelem. Barvy tlačítek v panelu byly upraveny v závislosti na jejich funkcích pro lepší přehlednost. Panel zároveň obsahuje informace o produkci a životech budovy. Tyto informace se automaticky aktualizují při změně stavu budovy.

Závěrečný úkol
Rozhraní indikující stav závěrečného úkolu v poslední úrovni bylo minimalizováno a začleněno do volného místa na liště se surovinami, aby hráči co nejméně překáželo při hraní. Pro jeho implementaci byla využita stejná kombinace tříd `Quest` a `Task`, jako v případě tutoriálu. Rozhraní indikuje stav každého ze čtyř pilířů, které je nutné napájet, a čas, který zbývá pro získání mimozemského artefaktu. Po získání artefaktu je rozhraní nahrazeno textem indikujícím, že artefakt byl úspěšně získán a hráč může úroveň opustit.

5.3.2 Rozhraní mezi úrovněmi

Po dokončení úrovně je hráč přesunut do menu skládajícího se z obrazovky pro nákup vylepšení a obrazovky pro výběr další úrovně k přistání.

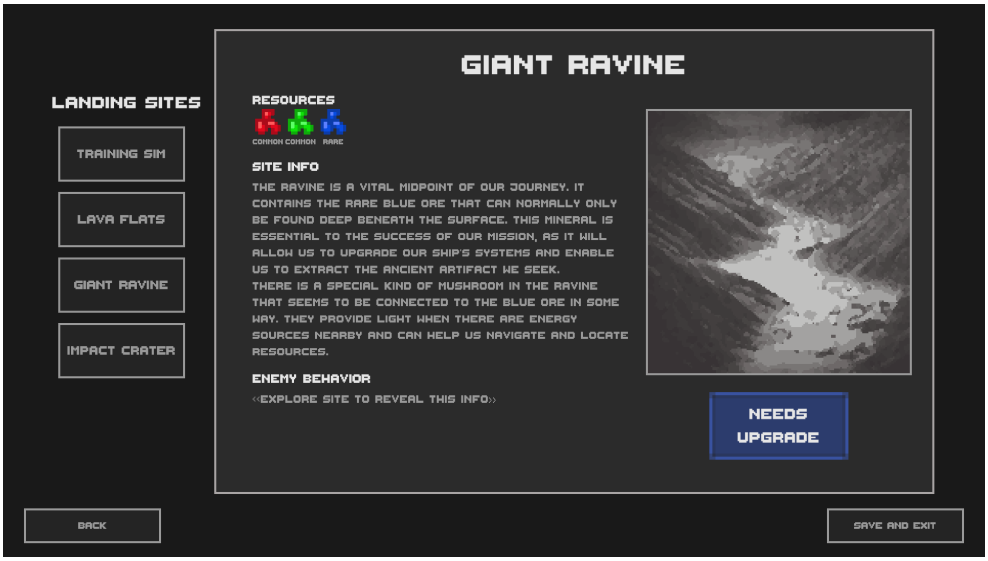
Obrázek 5.5: Obrazovka s vylepšeními

Nákup vylepšení
Finální vzhled obrazovky s vylepšeními si můžete prohlédnout na obrázku 5.5. Zakoupená vylepšení jsou označena zelenou značkou zaškrtnutí, dostupná vylepšení mají normální barvu a nedostupná vylepšení jsou ztmavená. Ztmavení tlačítek s vylepšeními aktivně reaguje na změny množství dostupných krystalů. Obdobně jako při za kupování budov je i v případě vylepšení barva jejich ceny jasně červená, pokud hráč nemá dostatek příslušné barvy krystalů.

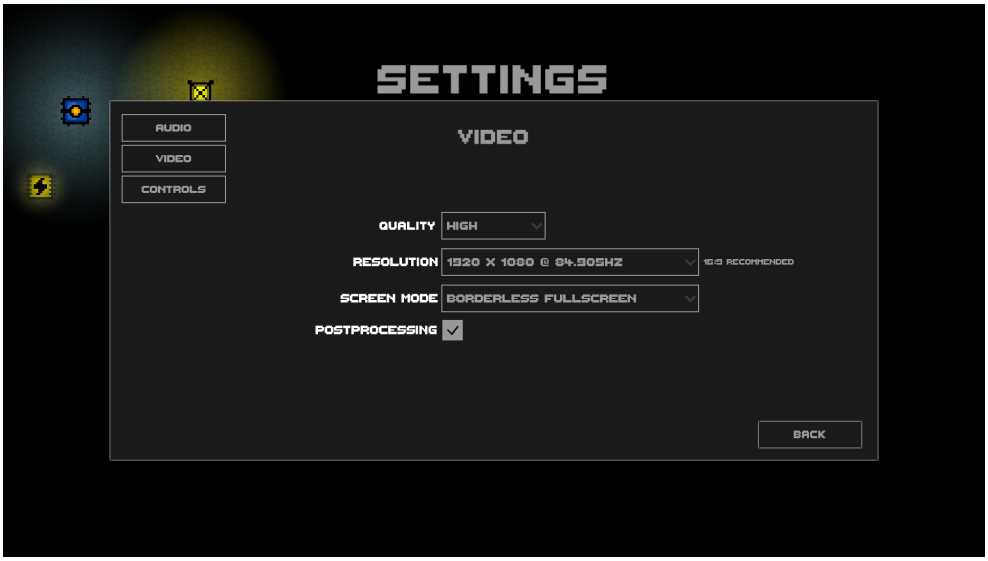
Výběr lokace k přistání
Obrazovka pro výběr místa přistání sestává ze čtyř informačních panelů pro jednotlivé úrovně doplněných ilustracemi vytvořenými s pomocí AI obdobně, jako v případě ilustrací použitých při prezentaci příběhu (viz sekce 5.1). Jeden z těchto panelů můžete vidět na obrázku 5.6. Zaškrtávací pole pro zopakování příběhové scény před začátkem úrovně se zobrazí pouze v případě, že hráč již scénu jednou viděl. V opačném případě je scéna spouštěna automaticky. Po první návštěvě každé úrovně se zobrazí popis chování agentů na dané mapě. Třetí a čtvrtou úroveň lze spustit pouze se zakoupeným vylepšením, na což je hráč v případě absence tohoto vylepšení textově upozorněn a s tlačítkem pro přistání v dané úrovni nebude možné interagovat.

5.3.3 Nastavení

Obrazovka pro nastavení zvuku obsahuje pět posuvníků pro jednotlivé druhy zvuků ve hře (viz sekce 5.6). Při otevření nastavení zvuku jsou pozice posuvníků automaticky nastaveny dle aktuálních hodnot hlasitostí. Při změnách



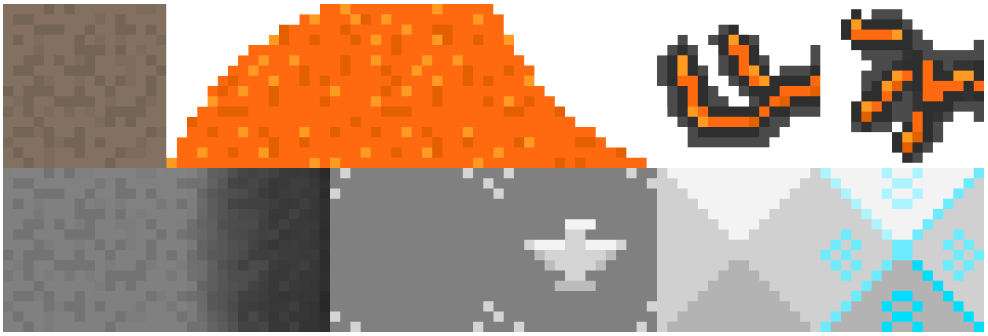
Obrázek 5.6: Panel pro přistání na mapě s trhlinou



Obrázek 5.7: Nastavení obrazu

polohy posuvníků hráčem jsou automaticky přehrávány ukázkové zvuky nastavovaného druhu, aby měl hráč představu o výsledné hlasitosti zvuků, které nastavuje. Nastavení obrazu nabízí rozbalovací menu pro volbu grafické kvality, rozlišení a módu celé obrazovky. Jeho podobu můžete vidět na obrázku 5.7. Rovněž byla přidána možnost deaktivovat postprocessing. Stránka s ovládáním hráči poskytuje přehled o všech dostupných klávesových zkratkách ve hře.

Aplikaci nově nastavených hodnot zajišťují jednotlivé skripty umístěné přímo na prvcích uživatelského rozhraní. Persistence nastavení zvuku i obrazu



Obrázek 5.8: Nové statické textury ve hře

je zajištěna využitím funkcí třídy `PlayerPrefs` poskytované Unity, která je k tomuto účelu určena. Pro aplikaci uložených hodnot po opětovném spuštění hry byla vytvořena třída `Settings`.

5.3.4 Ovládání

Ovládání hry bylo rozšířeno o možnost pohybu kamerou pohybem myší při držení kolečka. Byly rovněž přidány zkratky pro výběr schopností klávesami Q, E a F, výběr budov klávesami 1 až 7, změnu rychlosti času tabulátorem a pozastavení hry mezerníkem. Tím byl adresován nedostatek klávesových zkratek popsaný v heuristické analýze (viz sekce 3.6).

5.4 Úrovně

Obecné zasazení jednotlivých úrovní již bylo stručně popsáno v sekci 3.3.3. V této části bude konkrétněji představena jejich podoba a prvky, které byly pro účely jednotlivých úrovní vytvořeny.

Tutoriál představuje úroveň malé rozlohy, která se postupně s novými úkoly (viz sekce 5.2) rozrůstá. Aby bylo zřejmé, že se jedná pouze o simulaci, byla kamenná textura podlahy nahrazena dlaždicovitou texturou s pravidelnými vzory, kterou můžete vidět v dolní části obrázku 5.8. Nepřátelé mají rovněž v tutoriálu podobu terčů, aby byl více zřejmý jejich účel a nebyla hned v tutoriálu prozrazena skutečná podoba nepřátel ve hře.

Dominantou úrovně *Lava flats* je lávová řeka v pravé části mapy. Ta hráči poskytuje záchytný bod a velký zdroj přírodního světla. Zároveň díky ní mohou nepřátelé přicházet pouze z levé strany mapy. V okolí řeky se vyskytuje mnoho ložisek lávy, které lze využít pro zisk energie. Pro větší rozmanitost bylo do hry přidáno více textur těchto ložisek. Celá mapa je pokryta žílami červené rudy, jejichž velikost mírně roste se vzdáleností od hráčova přistávacího modulu. Poblíž hnízd nepřátel se pak vyskytuje rovněž zelená ruda, kterou hráč potřebuje k postupu do další úrovně.

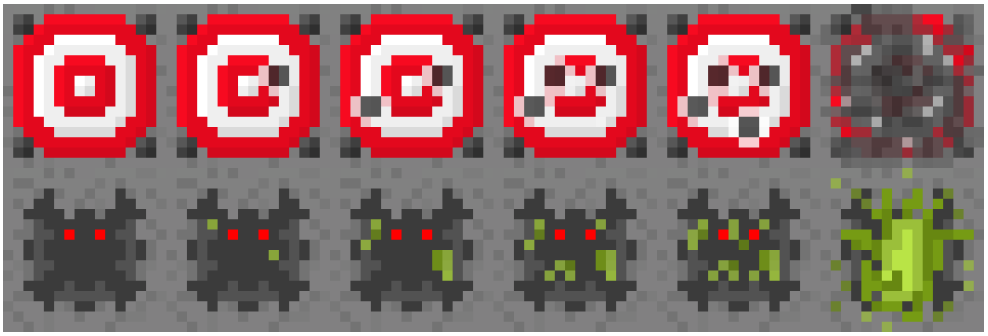
Úroveň *Giant ravine* je specifická útesy na pravé i levé straně mapy. Ty způsobují, že hráč může rozšiřovat základnu pouze na sever či na jih a nepřátelé rovněž mohou přicházet pouze z těchto dvou světových stran. Úroveň je hojně pokryta červenou i zelenou rudou, modrá ruda se pak nachází pouze na horním a dolním okraji poblíž hnízd nepřátel. V úrovni se rovněž nachází velké množství houbovitých koberců, které při zapojení do sítě začnou svítit, a tím hráči usnadní objevování mapy a obranu před nepřáteli. Struktura hub byla upravena tak, aby pro aktivaci celé skupiny hub stačilo zapojit pouze jedno políčko této skupiny. V prototypu oproti tomu bylo každé políčko s houbami aktivováno zvlášť.

Povrch úrovně *Impact crater* je nejvíce podobný původní mapě použité v prototypu hry. Jsou po něm hojně rozsety všechny tři druhy barevných rud a neobsahuje žádné překážky, nepřátelé tudíž mohou na hráče útočit ze všech směrů. V úrovni se rovněž nacházejí čtyři prastaré pilíře, které hráč musí aktivovat na určenou dobu, aby překonal sílu zabraňující mu v získání artefaktu potřebného k dokončení hry.

5.5 Vizuální efekty

Pro lepší vizuální požitek ze hry bylo přidáno vícero vizuálních efektů. Jejich popisu se věnuje tato sekce.

- **Plynulé rozsvěcení světel**
  Světla vyzařovaná budovami byla opatřena skriptem zajišťujícím jejich postupné rozsvěcení s využitím Unity *Coroutines*, aby jejich umisťování ve scéně působilo přirozeněji a plynuleji.
- **Přistání**
  Začátek úrovní odehrávajících se na povrchu toulavé planety byl doplněn efektem přistání hráčova základového modulu. Ten se skládá ze čtyř částicových systémů představujících plameny šlehající z trysek modulu v kombinaci s hlasitým hučením a je zakončen zvukem dopadu modulu na povrch. Až po dopadu se začne světlo modulu rozsvěcet a zároveň je postupně odhaleno uživatelské rozhraní s užitím Unity komponenty *Canvas Group*.
- **Pokládání budov**
  Stavění budov bylo doplněno zaduněním a jednoduchým částicovým efektem představujícím prach odlétající z místa, na kterém byla budova umístěna.
- **Schopnosti**
  Podobně jako světla budov jsou i světla orbitálního laseru a světlic rozsvícena postupně a na závěr jejich působení opět postupně ztmavována. Síla orbitálního laseru pak byla umocněna využitím částicového efektu.



Obrázek 5.9: Změny textury nepřátel v závislosti na zdraví

- **Výbuchy**
  Zničení budov bylo zvýrazněno hlasitým výbuchem, který doprovází částicový efekt reprezentující explozi. Zničené budovy zanechávají texturu ruiny, která časem sama mizí.

- **Životy nepřátel**
  Zdraví nepřátel je ve hře reprezentováno změnami jejich textur. Za tímto účelem byla do hry přidána řada textur znázorňujících různé úrovně poškození (viz obrázek 5.9). Po smrti nepřátelé zanechávají krvavé fleky.

- **Postprocessing**
  Do hry byla přidána postprocessing vrstva skrze Unity komponentu *Volume*. Po experimentaci s různými dostupnými efekty bylo rozhodnuto využít pouze efekt vinětace a bloomu, protože ostatní efekty působily spíše rušivé. Jelikož může být bloom náročný na výkon, byla do nastavení přidána možnost postprocessing deaktivovat.

5.6 Zvuky

Zvuky ve hře byly s pomocí Unity audio mixerů rozděleny do tří základních skupin, a to na hudbu, zvukové efekty a řeč. Každá z těchto skupin má na obrazovce pro nastavení zvuku vlastní posuvník. Střelba pak byla v rámci zvukových efektů vyčleněna do samostatné podskupiny, jelikož mohou být zvuky střelby při vysokém množství kulometů nepříjemné, a tak je vhodné hráčům poskytnout možnost ztlumit tuto skupinu zvuků zvlášť. Veškeré zvukové efekty a hudba ve hře pochází ze stránky pixabay[5], která poskytuje obsah k volnému užití.

[5] `https://pixabay.com/`

5.6.1 Zvukové efekty

Hra obsahuje poměrně rozmanitou škálu zvukových efektů, z nichž některé již byly zmíněny v sekci 5.5. Zpětná vazba tlačítek v rozhraní všech scén je podpořena zvuky klikání, splnění úkolu v tutoriálu je zase oznámeno zacinkáním. Zvuky vydává též aktivace a stavění budov. Spuštěné těžební vrtáky hučí, kulomety hlasitě střílejí a při zásahu nepřítele se ozve čvachtavé plesknutí. Každá schopnost má taktéž vlastní specifický zvuk, přičemž obzvláště v případě orbitálního laseru byla při výběru zvuku věnována velká pozornost tomu, aby dostatečně umocnil efekt dané schopnosti.

V Unity jsou zvuky do hry přidávány skrze komponentu *Audio Source*. Při vkládání zvuků do hry bylo dbáno na to, aby žádný z nich nebyl neúměrně hlasitý či tichý. Spouštění zvuků je obvykle buď navázáno na aktivaci objektu, do kterého byla zvuková komponenta vložena, nebo prováděno skrze skript. Hru tak bylo rovněž třeba doplnit o skripty pro spouštění zvuků, a to obvykle v reakci na události, jako je například změna životů agenta.

5.6.2 Hudba

Hudba ve hře je spouštěna skriptem `MusicManager` v závislosti na aktuální scéně a tom, zde již nějaká hudba hraje. Pro hlavní menu byla zvolena více energická hudba, zatímco závěrečná scéna obsahuje hudbu s emocionálním tónem. Pro ostatní scény pak bylo vybráno několik ambientních skladeb, které jsou spouštěny náhodně. Pro zajištění plynulých přechodů mezi scénami bez utnutí hudby byla využita kombinace návrhového vzoru *Singleton* a funkce Unity `DontDestroyOnLoad`.

5.6.3 Řeč

Jak již bylo zmíněno v sekci 5.1, scény s příběhem obsahují syntetické hlasové nahrávky. Nahrávky byly vygenerovány s pomocí stránky TTSMaker[6] pro syntetizaci řeči na základě textů, která umožňuje vytvořené nahrávky využívat k libovolným účelům. Aby hlas zněl roboticky, bylo na příslušný Unity audio mixer aplikováno několik filtrů (angl. attenuations). Konkrétně byly využity filtry *Flange*, *Echo* a *Lowpass*.

[6] `https://ttsmaker.com/`

Kapitola 6

Testování

Pro otestování hry bylo zorganizováno kvantitativní uživatelské testování, které umožní objevit nejen chyby v kódu, ale i nedostatky uživatelského rozhraní, problémy s vyvážeností hry a funkčnost na zařízeních s různými specifikacemi. Budou jím také získána data o tom, jak reální uživatelé hodnotí jednotlivé aspekty hry. Před veřejným testováním byla hra průběžně testována autorem s pomocí Unity editoru a rovněž odeslána dvěma dalším dobrovolníkům pro ověření funkčnosti.

Pro účely tohoto testování byl vytvořen dotazník, který bude detailněji popsán v části 6.1. Hra byla publikována na stránce itch.io[7]. Ta umožňuje nezávislým herním vývojářům snadno a zdarma vydávat hry bez složitých registrací či kontrol. Uživatelé si pak mohou stránku hry snadno prohlédnout v prohlížeči a hru stáhnout i bez účtu na této stránce. Na stejné stránce pak bude po dokončení testování a potřebných úpravách dle jeho výsledků vydána i finální verze hry.

Odkazy na stránku hry a dotazník byly spolu s krátkým komentářem a ukázkovými snímky ze hry sdíleny v několika studentských a hráčských skupinách skrze sociální síť Facebook a komunikační platformu Discord. Hře byla rovněž vytvořena stránka v archivu československých her a vývojářů *Visiongame*[8].

6.1 Dotazník

Dotazník, který měli hráči vyplnit po vyzkoušení hry, byl strukturován tak, aby pro jeho vyplnění nebylo nutné psát dlouhé odpovědi, jejichž časová náročnost by mohla mnoho vyplňujících odradit, ale zároveň poskytoval vždy dostatek místa pro vyjádření. Všechny povinné otázky jsou proto pouze za-

[7] `https://strnny.itch.io/rogue-planet`
[8] `https://visiongame.cz/hra/rogue-planet/`

klikávací, avšak bývají doplněny i nepovinnými rozšiřujícími otázkami s textovým polem. Dotazník se skládá z několika sekcí s různým účelem:

1. **Úvodní otázky**
   Na začátku dotazníku mají hráči zvolit, jakou verzi hry ozkoušeli a kolik jí věnovali času. Pokud hráči v úvodní sekci vyplnili, že se jim hru podařilo spustit, pokračují do sekce pro hodnocení tutoriálu. V opačném případě jsou dotázáni, proč se jim hru nepodařilo spustit. Mohou vybrat jednu z předpřipravených možností, či zadat vlastní. Rovněž jim je dán prostor pro detailnější vysvětlení.
2. **Srozumitelnost tutoriálu**
   Hráči mají na čtyřbodové škále ohodnotit jasnost jednotlivých vysvětlení různých konceptů v tutoriálu. Dále jsou pak dotazováni, zda jim tutoriál připadal příliš volný a nejednoznačný, nebo naopak příliš omezující. Taktéž mají možnost vyjádřit se, zda jim v tutoriálu chybělo vysvětlení nějaké mechaniky či funkce. Nakonec jsou dotázáni, zda se jim podařilo tutoriál dokončit. V případě úspěchu pokračují na hodnocení jednotlivých úrovní, v případě neúspěchu pak na doplňující sekci pro dovysvětlení problému.
3. **Obtížnost jednotlivých úrovní**
   Pro každou úroveň ve hře je hráčům poskytnuta možnost ohodnotit obtížnost úkonů v ní prováděných na pětibodové škále. Hodnotit mají například obtížnost obrany před nepřáteli, získávání rudy pro vylepšení, či hledání rudy pro pokrok do další úrovně. V závislosti na tom, zda hráči dokázali získat dostatek krystalů pro odemčení další úrovně, pokračují buď na doplňující otázky ohledně důvodů neúspěchu, nebo na hodnocení odemčené úrovně.
4. **Hodnocení různorodých aspektů hry**
   V případě, že hráči dohráli alespoň tutoriál, jsou požádáni o zhodnocení různých aspektů hry. Pokud hráči tutoriál nedohráli, předpokládá se, že nemají dostatek informací na to, aby tyto aspekty hodnotili.

   Hráči mají nejprve s pomocí čtyřbodové škály postupně ohodnotit následující vlastnosti: *zábavnost*, *ovládání*, *uživatelské rozhraní a interakce s ním*, *vzhled a grafický styl*, *příběh*, *zasazení a jeho prezentace*, *hudba a zvukové efekty*, *možnosti nastavení* a *výkon a optimalizace*. Dále pak mohou popsat nalezené chyby, pokud na nějaké v průběhu hraní narazili, a navrhnout, co dalšího by do hry rádi přidali, pokud by měli tu možnost.
5. **Informace o hráči**
   Závěrečný segment dotazníku je zaměřen na sběr informací o samotných hráčích, kteří dotazník vyplňují. Cílem je zjistit, zda vyplňující patří do cílové skupiny hry, či nikoliv. Hráči zde musí vyplnit, kolik času denně průměrně tráví hrami a v jakém hrají rozlišení. Dále mají ze seznamu

her podobných Rogue Planet vybrat, které z nich již hráli, a případně doplnit další. Seznam obsahoval následující hry: *Faster Than Light*, *Loop Hero*, *Duskers*, *Creeper World* (libovolný díl), *Factorio*, *Mindustry*, *Cities: Skylines*, *Workers & Resources: Soviet Republic*, *Prison Architect*, *Rimworld*, *They Are Billions* a *Rise to Ruins*. Na závěr pak hráči mají uvést, odkud se o hře a jejím testování dozvěděli a je jim poskytnuta poslední možnost pro doplnění libovolné zpětné vazby.

6.2 Verze a změny hry v průběhu testování

Hra již v průběhu testování prošla drobnými opravami a změnami na základě zpětné vazby obdržené od hráčů. Během testování tak byly hráčům v různých časových rozmezích dostupné následující verze hry:

- **RoguePlanet_v1.0.0**
  Verze 1.0.0 byla přístupná první den testování a za tuto dobu byla stažena celkem 14krát.
- **RoguePlanet_v1.0.1**
  Verze 1.0.1 po dni testování nahradila verzi 1.0.0. Dostupná byla jen několik hodin v noci a stažena byla za tuto dobu pouze 2krát. Tato verze přinesla opravu problému, kdy hru provázely krátké záseky při spouštění nové hudební skladby. Tato chyba byla opravena přenastavením způsobu načítání hudebních klipů do operační paměti. Dále pak byla poupravena cena některých vylepšení. Nejvýrazněji byla změněna cena vylepšení pro odemčení poslední úrovně, která byla v původní verzi přehnaně vysoká.
- **RoguePlanet_v1.0.2**
  Poslední verzí hry dostupnou v rámci testování byla verze 1.0.2. Ta byla dostupná téměř týden, tedy až do konce prováděného testování. Za tuto dobu byla stažena celkem 54krát.

  Verze přinesla opravu nepříjemné chyby, kdy občas zdánlivě náhodně nedošlo k postavení zvolené budovy po kliknutí na mapu. Bylo rovněž opraveno několik drobných překlepů a některé texty byly přepsány pro lepší výstižnost. Z komunikace s jedním z testerů například vyplynulo, že se tester obával v menu s vylepšeními kliknout na tlačítko „Land“, jelikož si nebyl jistý, zda bude jeho postup uložen, a proto vždy nejprve klikal na tlačítko „Save and exit“, a až poté se vrátil a pokračoval na obrazovku s výběrem úrovní. Tlačítko bylo z tohoto důvodu přejmenováno na „Save and land“.

  Jelikož si vícero hráčů stěžovalo, že si nedostatku energie všimnou v zápalu boje často až ve chvíli, kdy energie klesne na nulu a jejich budovy se začnou hromadně deaktivovat, byl do hry v této verzi přidán jednoduchý varovný systém, který hráče upozorní na nedostatek suroviny několik sekund před jejím vyčerpáním. Přidán byl rovněž text sig-

nalizující pozastavení hry nad rámec již existujícího textu zobrazujícího rychlost herního času.

- **RoguePlanet_v1.0.2_experimental_linux**
  Na žádost několika testerů došlo k pokusu o vytvoření experimentální verze hry pro OS Linux. Tvorba verze pro Linux v Unity se ukázala být po doinstalování potřebného balíčku poměrně jednoduchá. Tato verze tak byla zveřejněna souběžně s verzí 1.0.2 určenou pro OS Windows, od které se liší pouze cílovou platformou, a stažena byla celkem 14krát.
- **RoguePlanet_v1.0.2_experimental_macOS**
  Ve spolupráci se dvěma testery využívajícími zařízení s macOS došlo na jejich žádost k pokusu o vytvoření verze pro macOS. Bohužel vyšlo najevo, že hra nebude na macOS spustitelná bez absolvování procesu „notářského ověření“ (angl. notarization) od společnosti Apple. Ten je na operačním systému Windows dle Unity dokumentace [27] možné provést pouze s využitím placené služby *Unity Cloud Build*, a zároveň by vyžadoval mimo jiné Apple ID a členství v programu pro vývojáře od Apple. Z těchto důvodů verze hry pro macOS nebyla zveřejněna a její další vývoj není v plánu.

6.3 Výsledky testování

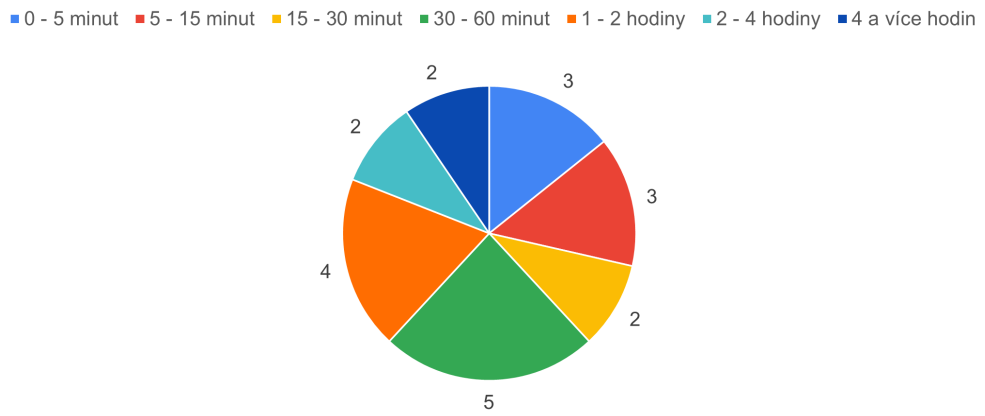
V průběhu testování byla hra stažena celkem 83krát. Dá se předpokládat, že někteří hráči hru stáhli vícekrát vzhledem k zpřístupnění nových verzí v průběhu testování, a reálný počet účastníků testování tak bude nižší. Dotazník vyplnilo celkem 21 z nich. Mnoho dalších účastníků se pak rozhodlo dotazník ignorovat a svoji zpětnou vazbu předat skrze komentáře či zprávy.

6.3.1 Úvodní otázky

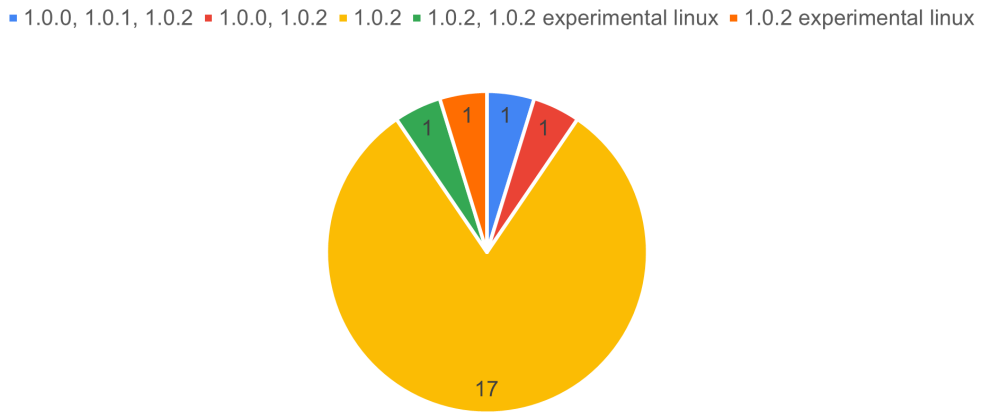
Hráči u hry strávili poměrně různorodé množství času. Jak můžete vidět na obrázku 6.1, někteří u hry strávili několik minut, jiní zase i vícero hodin. Valná většina hráčů vyzkoušela verzi 1.0.2, někteří pak i předchozí verze, viz obrázek 6.2. Překvapením bylo, že žádný z respondentů neozkoušel pouze první verzi hry. Všem respondentům kromě jednoho se podařilo hru úspěšně spustit. Z doplňujícího komentáře vyplynulo, že důvodem selhání bylo, že se respondent pokoušel spustit verzi pro OS Windows na macOS.

6.3.2 Srozumitelnost tutoriálu

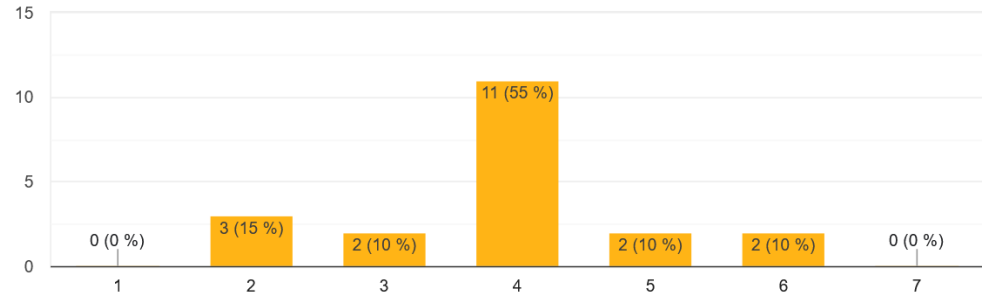
Srozumitelnost vysvětlení konceptů v tutoriálu hodnotili hráči převážně pozitivně, viz tabulka 6.1. Jako problematická se ukázala položka „Účel různých surovin“. Cílem tutoriálu bylo hráčům v tomto ohledu sdělit, že bílá rudá slouží k stavění budov, zatímco barevné rudy slouží ke generování energie a tvorbě



Obrázek 6.1: Čas strávený hraním Rogue Planet



Obrázek 6.2: Verze hry, které hráči testovali



Obrázek 6.3: Volnost tutoriálu

Tabulka 6.1: Jasnost konceptů v tutoriálu

	jasné	spíše jasné	spíše nejasné	nejasné
Pohyb Kamery	18	2	0	0
Jak získávat suroviny	16	4	0	0
Že budovy potřebují napojení na energii	13	5	2	0
Funkce jednotlivých budov	10	8	2	0
Že je třeba nepřítele osvětlit, aby po něm šlo střílet	16	4	0	0
Účel různých surovin	9	5	5	1
Jak lze budovy vypínat, přepínat a demolovat	13	7	0	0
Funkce schopností	12	5	3	0
Že cílem je získávat krystaly pro vylepšení	11	7	2	0
Jak ukončit úroveň	14	3	2	1

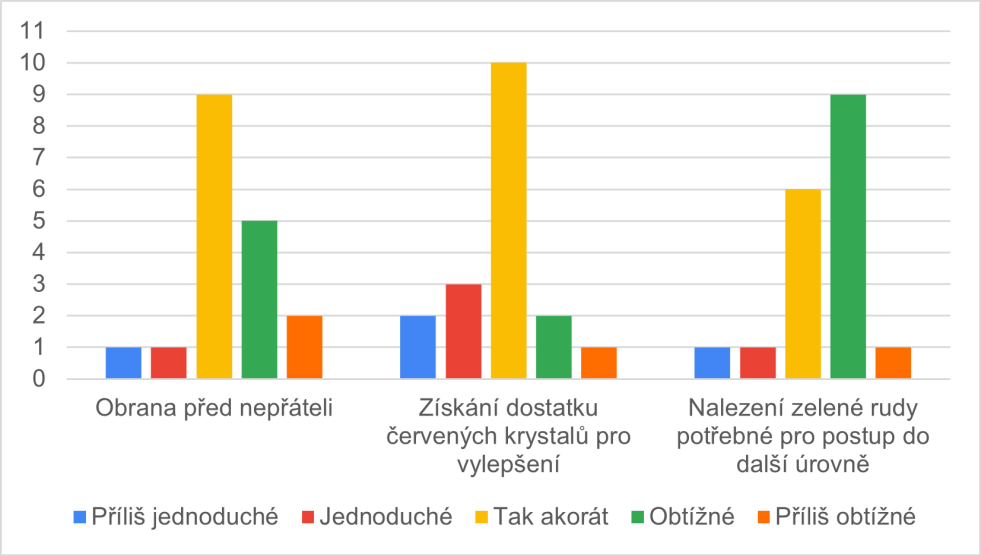
krystalů. Hráči dle doplňujících komentářů tuto položku označovali jako spíše nejasnou, jelikož nebyl vysvětlen rozdíl mezi jednotlivými barevnými surovinami, o což však tutoriál neusiloval. Spíše než o chybu tutoriálu se tedy jedná o nevhodně položenou otázku.

Hráči rovněž na stupnici s hodnotami 1 až 7 hodnotili, zda je tutoriál příliš omezoval, nebo naopak spíše vedl příliš málo. Hodnota 1 značí, že hráče tutoriál příliš omezoval, vedl je za ruku a nemohli dělat, co chtěli. Hodnota 7 naopak znamená, že byl tutoriál příliš volný a hráči nevěděli, co mají dělat. Jak můžete vidět na obrázku 6.3, hráči volnost tutoriálu hodnotili převážně jako optimální s drobnými výkyvy na obě strany.

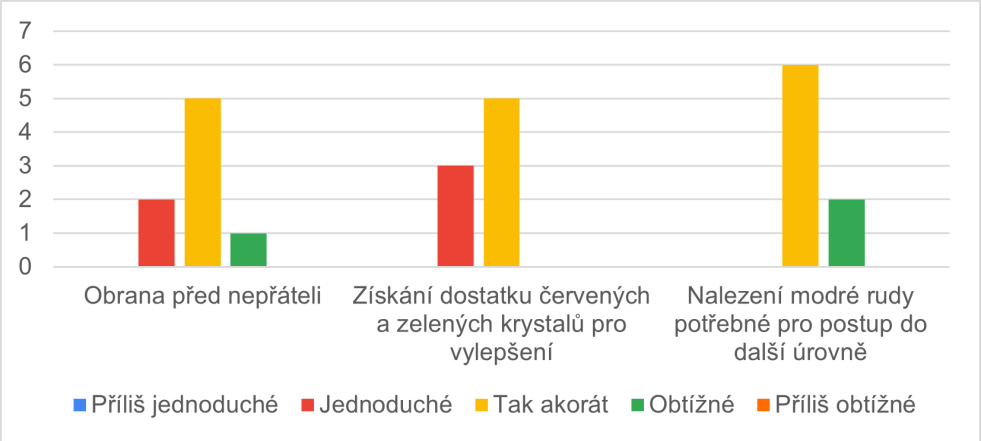
2 hráčům z 20 se nepodařilo tutoriál dokončit. První z nich uvedl, že mu tutoriál připadal příliš dlouhý, druhý se pak dle komentáře ztratil v úkolu zaměřeném na optimalizaci příjmu surovin a tutoriál vzdal. V obou případech hráči dle poslední sekce dotazníku neměli zkušenost ani s jednou ze zmíněných her podobných Rogue Planet.

6.3.3 Obtížnost jednotlivých úrovní

Hodnocení obtížnosti úrovně *Lava Flats* si můžete prohlédnout na obrázku 6.4. Obtížnost získávání červených krystalů se zřejmě podařilo vyvážit správně, hledání zelené rudy a obrana před nepřáteli však hráčům činila problémy. 10 hráčů zakončilo své testování v této úrovni a nepokračovalo dále. 4 z nich uvedli, že si chtěli hru pouze vyzkoušet a jako ukázka jim úroveň stačila. Další 4 snahu o pokročení do další úrovně vzdali s odůvodněním, že je příliš



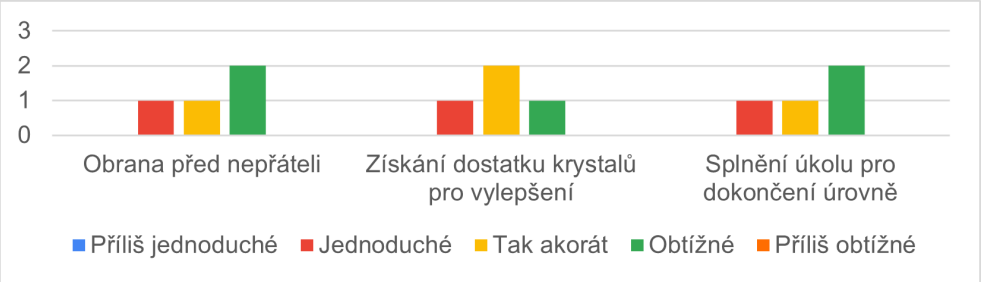
Obrázek 6.4: Hodnocení obtížnosti úrovně Lava Flats



Obrázek 6.5: Hodnocení obtížnosti úrovně Giant Ravine

obtížné bránit se před útoky nepřátel. Problém někteří z nich vidí především v příliš vysoké odolnosti nepřátel. 2 hráči pak nemohli pokračovat dále, jelikož nedokázali nalézt zelenou rudu potřebnou pro odemčení další úrovně.

Hodnocení obtížnosti úrovně *Giant Ravine* můžete vidět na obrázku 6.5. Obtížnost této mapy hráči hodnotí poměrně vyrovnaně ve všech třech aspektech. Polovině z hráčů, kteří se na tuto mapu dostali, se podařilo získat dostatek surovin pro pokračování do poslední úrovně. Jako důvod neúspěchu dva hráči uvádí, že si již hru dostatečně ozkoušeli a rozhodli se dále nepokračovat, přičemž jeden z nich zmiňuje, že mu v pokračování brání nedostatek času, ale rád by hru v budoucnosti dohrál. Další hráč uvedl, že je cena pro odemčení



Obrázek 6.6: Hodnocení obtížnosti úrovně Impact Crater

další úrovně příliš vysoká. Poslední pak jako důvod konce v této úrovni uvedl potřebu pokračovat v psaní nejnovější progtestové úlohy.

Hodnocení obtížnosti poslední úrovně zvané *Impact Crater* můžete vidět na obrázku 6.6. Jeden hráč ohodnotil všechny aspekty této mapy jako jednoduché, ostatní hráči pak hodnotili různé aspekty jako vyrovnané či obtížné. Jednoho hráče, který úroveň hodnotil jako převážně obtížnou, hra v této úrovni přestala bavit. Ostatním se podařilo úroveň dokončit, a tím dohrát celou hru.

6.3.4 Hodnocení různorodých aspektů hry

Hráči měli možnost vyjádřit svoji spokojenost s různými vlastnostmi hry. Souhrn těchto hodnocení naleznete v tabulce 6.2. Hru hodnotili hráči vesměs pozitivně. Negativní odpovědi pak vzhledem k nižšímu počtu byly analyzovány individuálně. Kroky provedené na základě nalezených chyb naleznete v sekci 6.4.

Zábavnost

Se zábavností hry byly spíše nespokojení čtyři hráči. Jeden z nich hodnotil negativně rovněž ovládání a uživatelské rozhraní. Z doplňujících komentářů vyplynulo, že hráč hru testoval na širokoúhlém monitoru, což způsobilo, že rozhraní hry nefungovalo správně a některé jeho prvky nebyly vůbec vidět. Hráčův zážitek tak byl touto chybou výrazně negativně ovlivněn. Druhý z testerů spíše nespokojených se zábavností hry hodnotil všechny ostatní aspekty hry pozitivně. V ostatních částech dotazníku se taktéž vyjadřoval příznivě a své negativní hodnocení zábavnosti nedoplnil žádným komentářem. Příčinu nespokojenosti tak nelze jednoznačně určit.

Zbylí dva testeři s negativním hodnocením zábavnosti hodnotili spíše negativně rovněž uživatelské rozhraní. Jeden z nich hodnotil zábavnost negativně, jelikož byl v první úrovni hry po delším hraní převálcován nepřátelskými agenty i přes důkladnou obranu, což pro něj bylo frustrující. Později se ukázalo, že příčinou byla nejspíše chyba ve škálování životů agentů. Druhý tester pak nedokázal v první úrovni nalézt zelenou rudu a hra ho proto přestala bavit.

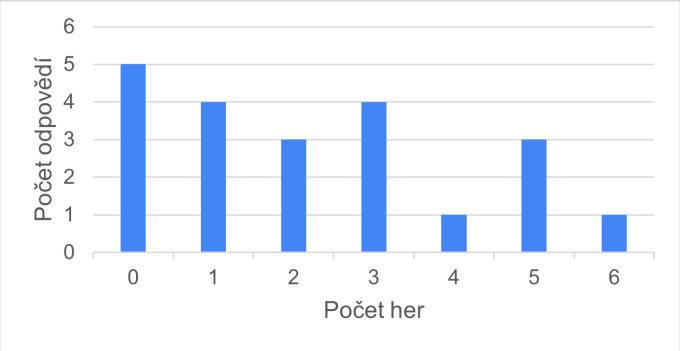
Tabulka 6.2: Hodnocení různorodých aspektů hry

	Spokojen	Spíše spokojen	Spíše nespokojen	Nespokojen
Zábavnost	9	5	4	0
Ovládání	6	10	1	1
UI a interakce s ním	6	7	4	1
Vzhled a grafický styl	6	10	2	0
Příběh a zasazení	7	10	1	0
Hudba a zvukové efekty	10	7	1	0
Možnosti nastavení	8	7	3	0
Výkon a optimalizace	15	3	0	0

Ovládání
Kromě již zmíněného hráče s problémem s širokoúhlou obrazovkou hodnotil ovládání negativně ještě jeden další hráč. Tomu se nelíbila kombinace real-time strategie a možnosti pozastavení času. Argumentuje, že tato kombinace hráče nutí k výběru, zda hru neustále pozastavovat a optimalizovat produkci surovin aktivací a deaktivací budov, což je velmi pracné, nebo hrát hru neoptimálním způsobem. Kvůli zmíněné pracnosti pak hodnotí spíše negativně i uživatelské rozhraní hry.

Uživatelské rozhraní
Kromě již zmíněných tří hráčů hodnotili uživatelské rozhraní negativně ještě další dva. Jeden z nich neuvádí konkrétní důvod nespokojenosti, avšak na otázku ohledně dalších součástí hry, které by rád viděl, odpověděl obsáhlým seznamem funkcí a mechanik. Ohledně UI zmiňuje například healthbar, ukazatele zranění, či vyobrazení propojenosti rozvodem energie. Zcela negativní hodnocení UI pak zanechal hráč, pro kterého bylo rozhraní dle komentáře nepřehledné vzhledem k jeho barvosleposti.

Ostatní
Zbylé aspekty hry byly hodnoceny vesměs pozitivně. Případná negativní hodnocení pak obvykle neobsahovala žádný komentář, co konkrétně se hráčům nelíbilo. Jeden hráč zmínil, že pro něj bylo obtížné rozpoznat jednotlivé budovy. Jiný hráč pak ohodnotil jako spíše neuspokojivé možnosti nastavení, hudbu, zvuky, i prezentaci příběhu a zasazení, avšak jeho závěrečný zhodnocující komentář byl velmi pozitivní.



Obrázek 6.7: Počet podobných her, které testeři hráli

6.3.5 Návrhy na další obsah

Po ohodnocení hry měli hráči taktéž možnost napsat, co dalšího by si do hry přáli přidat. Žádané funkcionality naleznete seřazené od nejčastěji žádaných v následujícím seznamu:

1. Ovládání více budov najednou
2. Vylepšování budov během úrovně
3. Zvýraznění zničené budovy
4. Více druhů obranných budov
5. Snadněji rozlišitelné textury budov
6. Zobrazení propojení rozvoden energie
7. Ovládání směru reflektorů
8. Možnost zobrazit dosah všech kulometů najednou
9. Více map
10. Další animace

6.3.6 Informace o respondentech

Všichni respondenti byli muži. Pět z nich ve věkovém rozmezí 19–21 let, deset v rozmezí 22–25 let, dva v rozmezí 26–30 let a čtyři v rozmezí 31–40 let. Osmnáct respondentů uvedlo, že hry obvykle hrají ve FullHD rozlišení, dva ve 2K rozlišení a jeden v rozlišení UW-QHD. Respondenti měli zvolit ze seznamu dvanácti her podobných Rogue Planet ty, které hráli, a případně doplnit další. Kolik podobných her respondenti hráli můžete vidět v grafu na obrázku 6.7.

6.4 Změny provedené po testování

V reakci na nalezené chyby a nedostatky hry byly provedeny následující změny v uvedeném pořadí:

1. **Opravy ukotvení UI**
   Při diskusi s testery se ukázalo, že některé prvky uživatelského rozhraní nefungují na širokoúhlých monitorech správně, jelikož jsou kvůli nezvyklému poměru stran roztaženy mimo obrazovku. Tato chyba byla opravena změnou ukotvení těchto prvků UI. Hra je stále primárně určena pro obrazovky s poměrem stran 16:9, avšak nyní by měla správně fungovat i na obrazovkách s jinými poměry.
2. **Velikost nepřátel**
   V reakci na nejasnosti ohledně síly nepřátel byla hra upravena tak, aby velikost nepřátel mírně rostla v závislosti na jejich maximálním zdraví.
3. **Poděkování testerům**
   Jakožto poděkování za odvedenou práci bylo testerům, kteří hru dohráli až do konce, nabídnuto jejich zmínění v rámci závěrečných titulků. Dva ze tří úspěšných testerů souhlasili a byli přidáni.
4. **Zkratky v tutoriálu**
   Některým testerům ve hře chyběly zkratky pro výběr budov a schopností navzdory tomu, že se tyto zkratky ve hře již vyskytují. Z tohoto důvodu bylo na jejich přítomnost kromě již existujících nápisů pod tlačítky upozorněno rovněž novými nápisy v tutoriálu.
5. **Oprava chyby s krystaly**
   Někteří testeři se setkali s chybou, kdy jim po zakoupení vylepšení nebyly odečteny krystaly. Na vině se ukázala být chyba způsobená neplatnými referencemi při opakovaném načtení scény pro nákup vylepšení. Chyba byla odstraněna.
6. **Množství rudy**
   Jelikož vícero hráčům připadalo hledání rudy pro postup do další úrovně příliš obtížné na to, aby se dostali dále, bylo množství zelené rudy na mapě *Lava Flats* a modré rudy na mapě *Giant Ravine* zvýšeno. Zároveň byly upraveny ceny některých vylepšení.
7. **Chování nepřátel v úrovni Giant Ravine**
   Pozorovatelné chování mravenčích agentů v úrovni *Giant Ravine* hodnotili hráči jako nezajímavé, jelikož všichni nepřátelé po určitém čase začali následovat stále stejnou cestu. To hráčům velmi usnadňovalo obranu základny, protože stačilo stavět velké množství kulometů na konci této cesty. Obtížnost se pak odvíjela pouze od odolnosti těchto nepřátel. Komunikační vzdálenost agentů v této úrovni tak byla snížena, aby agenti cestu časem zapomněli a hledali novou.

8. **Přidání chybějících zvuků UI**
   Do hry byly přidány zvuky při otevření interakčního UI budov a mačkání tlačítek v něm, které dříve chyběly.
9. **Oprava chyby se škálováním životů nepřátel**
   Větší množství hráčů uvedlo, že jim nepřátelé připadají příliš odolní. Nejprve se zdálo, že je to způsobeno tím, že hráči nevědí, že mají po delším čase opustit úroveň, jelikož zdraví nepřátel s délkou pobytu v úrovni neustále roste. Později se však ukázalo, že problém byl ve skutečnosti způsoben chybou ve hře, kdy zdraví nepřátel nezáviselo na čase od začátku scény, ale na čase od spuštění samotné hry. Chyba byla opravena a snížení obtížnosti způsobené jejím odstraněním bylo kompenzováno úpravami v rychlosti růstu životů agentů.
10. **Zrychlení interakce s budovami**
    Vícero hráčům připadalo přepínání stavu většího množství budov příliš pracné. Pro usnadnění práce s budovami bylo chování interakčního UI budov upraveno tak, aby tlačítka akcí po provedení akce automaticky UI zavřela a hráč ho tak nemusel zavírat manuálně.

Po odstranění všech nalezených chyb, které by mohly bránit ve vydání hry, byla nejnovější verze hry pro OS Windows zveřejněna opět skrze platformu itch.io[9]. Dodána byla též nejnovější verze hry zkompilovaná pro OS Linux, primární cílovou platformou jsou však stále Windows a verze pro Linux je pouze doplňková.

[9] `https://strnny.itch.io/rogue-planet`

Závěr

Hlavním cílem této práce byla realizace strategické hry Rogue Planet na základě výstupu předešlé bakalářské práce a její následné vydání. Na základě analýzy stavu již existujícího prototypu hry, případů užití budoucího rozhraní a person uživatelů byly zvoleny vhodné cesty dopracování hry a vytvořen návrh její finální herní i softwarové podoby s využitím nástrojů softwarového inženýrství představených v rešerši.

Uživatelské rozhraní hry bylo z výrazné části přepracováno a dále rozšířeno pro podporu nových funkcí hry. Při jeho vývoji bylo dbáno na jeho použitelnost a zachování původního grafického stylu. Velká část kódu hry byla rovněž přepracována pro lepší rozšiřitelnost a srozumitelnost, čehož bylo využito při přidávání nových funkcí a mechanik hry a jejím dalším rozšiřování. Pro lepší využití modelu chování nepřátelských agentů vytvořených v bakalářské práci byla hra rozdělena do tří úrovní, přičemž každá z nich využívá jedno z existujících chování. Hra byla též rozšířena o schopnosti, které hráčům umožňují nepřátelské chování lépe pozorovat a využívat. Pro lepší přístupnost novým hráčům byla hra rozšířena o podrobný tutoriál, který hráče seznámí s podstatnými mechanikami a prvky hry. Rozšířen byl i vizuál hry. Hra byla obohacena o další autorem vytvořené textury a nově také o ilustrace vytvořené s využitím generativní umělé inteligence. Přidány byly rovněž nové vizuální a částicové efekty. Byl vytvořen jednoduchý příběh provázející hráče celou hrou. Pro lepší uživatelský zážitek byla textová forma prezentace příběhu rozšířena o syntetické hlasové nahrávky. Většina akcí ve hře byla též obohacena zvukovými efekty a atmosféra hry je umocněna přidanou hudbou.

Výsledný softwarový produkt byl otestován kvantitativním uživatelským testováním, které přispělo nejen k odhalení případných chyb, ale rovněž i k posouzení srozumitelnosti vytvořeného tutoriálu, použitelnosti uživatelského rozhraní a vyvážení herní obtížnosti. Testování také poskytlo cenné informace o spokojenosti hráčů s různými aspekty hry a informace o tom, jaké další možnosti budoucího vývoje budou pro hru z pohledu hráčů nejvíce přínosné. Výsledky provedeného testování byly důkladně zanalyzovány a všechny nale-

zené chyby bránící v úspěšném vydání hry byly odstraněny. Hru lze v jejím současném stavu považovat za kompletní softwarový produkt, avšak možností pro další zlepšení v podobě nových mechanik, dalších funkcí uživatelského rozhraní či dodatečného herního obsahu je stále mnoho.

Výsledkem práce je plnohodnotná strategická hra, která byla pro hráče vydána skrze digitální distribuční platformu *itch.io*. Hra může posloužit jako propagační materiál FIT ČVUT a do budoucna má potenciál pokračovat v podobě komerčního projektu.

Bibliografie

1. COOPER, Alan. *Inmates Are Running the Asylum: Why High Tech Products Drive Us Crazy and How to Restore the Sanity*. 2nd ed. USA: Sams Publishing, 2004. ISBN 978-0-672-32614-1.
2. NIELSEN, Lene. Personas. In: SOEGAARD, Mads; DAM, Rikke Friis (ed.). *The Encyclopedia of Human-Computer Interaction* [online]. 2nd ed. The Interaction Design Foundation, 2014, kap. 30 [cit. 2023-01-30]. Dostupné z: `https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed`.
3. JACOBSON, Ivar; SPENCE, Ian; KERR, Brian. Use-Case 2.0: The Hub of Software Development. *Queue*. 2016, roč. 14, č. 1, s. 94–123. ISSN 1542-7730. Dostupné z DOI: `10.1145/2898442.2912151`.
4. COCKBURN, Alistar. *Writing Effective Use Cases*. 1st ed. USA: Addison-Wesley Professional, 2000. ISBN 978-0201702255.
5. NIELSEN, Jakob. *Usability Engineering*. USA: Morgan Kaufmann Publishers Inc., 1994. ISBN 978-0-08-052029-2.
6. NIELSEN, Jakob. *10 Usability Heuristics for User Interface Design* [online]. Nielsen Norman Group, 2020 [cit. 2023-02-04]. Dostupné z: `https://www.nngroup.com/articles/ten-usability-heuristics/`.
7. WHARTON, Cathleen; RIEMAN, John; LEWIS, Clayton; POLSON, Peter. The cognitive walkthrough method: a practitioner's guide. In: NIELSEN, Jakob (ed.). *Usability inspection methods*. USA: John Wiley & Sons, Inc., 1994, s. 105–140. ISBN 978-0-471-01877-3.
8. MARTIN, Robert Cecil. *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. 1st ed. USA: Pearson, 2017. Robert C. Martin Series. ISBN 978-0134494166.
9. HUNT, Andrew; THOMAS, David. *The Pragmatic Programmer: From Journeyman to Master*. 1st ed. USA: Addison-Wesley, 1999. ISBN 978-0201616224.

10. SHEIKHA, Hanna. *Keep It Simple, Stupid — The KISS Principle Guide to Developers* [online]. 2022. [cit. 2023-02-12]. Dostupné z: https://medium.com/sliitwif/keep-it-simple-stupid-the-kiss-principle-guide-to-developers-d6ad83145955.

11. MAKABEE, Hayim. *Separation of Concerns* [online]. 2012. [cit. 2023-02-12]. Dostupné z: https://effectivesoftwaredesign.com/2012/02/05/separation-of-concerns/.

12. FOWLER, Martin with contributions from; BECK, Ken; BRANT, John; OPDYKE, William; ROBERTS, Don. *Refactoring: Improving the Design of Existing Code*. USA: Addison-Wesley Professional, 1999. ISBN 978-0201485677.

13. REFACTORING.GURU. *Code Smells* [online]. 2023. [cit. 2023-03-24]. Dostupné z: https://refactoring.guru/refactoring/smells.

14. GAMMA, Erich; HELM, Richard; JOHNSON, Ralph; VLISSIDES, John. *Design Patterns: Elements of Reusable Object-Oriented Software*. 1st ed. USA: Addison-Wesley Professional, 1994. ISBN 978-0201633610.

15. NYSTROM, Robert. *Game Programming Patterns* [online]. 2014. [cit. 2023-02-13]. ISBN 978-0990582908. Dostupné z: https://gameprogrammingpatterns.com/.

16. LIN, Wilmer; KROUGH-JACOBSEN, Thomas; ANDREASEN, Peter; BILAS, Scott. *Level up your code with game programming patterns* [e-book]. Unity Technologies, 2022 [cit. 2023-02-13]. Dostupné z: https://resources.unity.com/games/level-up-your-code-with-game-programming-patterns.

17. TERRA, John. *Entity Component System: An Introductory Guide* [online]. 2023. [cit. 2023-02-28]. Dostupné z: https://www.simplilearn.com/entity-component-system-introductory-guide-article.

18. UNITY TECHNOLOGIES. *Entities overview* [online]. 2023. [cit. 2023-03-03]. Dostupné z: https://docs.unity3d.com/Packages/com.unity.entities@1.0/manual/index.html.

19. STRNAD, Ladislav. *Rogue planet: využití multiagentních systémů ve 2D hře* [online]. Praha, 2021 [cit. 2023-02-17]. Dostupné z: http://hdl.handle.net/10467/95014. Bakalářská práce. České vysoké učení technické v Praze, Fakulta informačních technologií. Vedoucí práce Radek RICHTR.

20. WIKIPEDIA. *Rogue planet* [online]. 2023. [cit. 2023-02-21]. Dostupné z: https://en.wikipedia.org/wiki/Rogue_planet.

21. MISFITS ATTIC. *Duskers* [soft.]. 2016. [cit. 2023-02-21]. Dostupné z: https://store.steampowered.com/app/254320/Duskers/.

22. THE CREATIVE ASSEMBLY LIMITED. *Alien: Isolation* [soft.]. Sega Corporation, 2014 [cit. 2023-02-21]. Dostupné z: https://store.steampowered.com/app/214490/Alien_Isolation/.

23. ŠIMEČEK, Ivan. *Posudek oponenta závěrečné práce* [online]. 2021. [cit. 2023-02-22]. Dostupné z: https://dspace.cvut.cz/bitstream/handle/10467/95014/F8-BP-2021-posudek-Simecek_Ivan.pdf.

24. GONZALEZ, Leandro. *How to Write a Game Design Document* [online]. 2016. [cit. 2023-03-19]. Dostupné z: https://www.gamedeveloper.com/business/how-to-write-a-game-design-document.

25. VALVE CORPORATION. *Hardwarový a softwarový průzkum: January 2023* [online]. 2023. [cit. 2023-02-23]. Dostupné z: https://store.steampowered.com/hwsurvey/Steam-Hardware-Software-Survey-Welcome-to-Steam.

26. GRANBERG, Aron. *A* Pathfinding Project* [online]. 2020. Ver. 4.2.15 [cit. 2023-04-22]. Dostupné z: https://arongranberg.com/astar/features#.

27. UNITY TECHNOLOGIES. *Notarizing your macOS application* [online]. 2023. [cit. 2023-04-30]. Dostupné z: https://docs.unity3d.com/Manual/macos-building-notarization.html#notarization-using-unity-cloud-build.

Příloha **A**

Seznam použitých zkratek

UI User Interface

SI Softwarové Inženýrství

JSON JavaScript Object Notation

LTS Long Term Support

MVC Model View Controller

MVP Model View Presenter

OOP Objektově Orientované Programování Entity-Component

EC Entity-Component

ECS Entity Component System

DOTS Data Oriented Technology Stack

DFS Depth-first Search

AI Artificial Intelligence

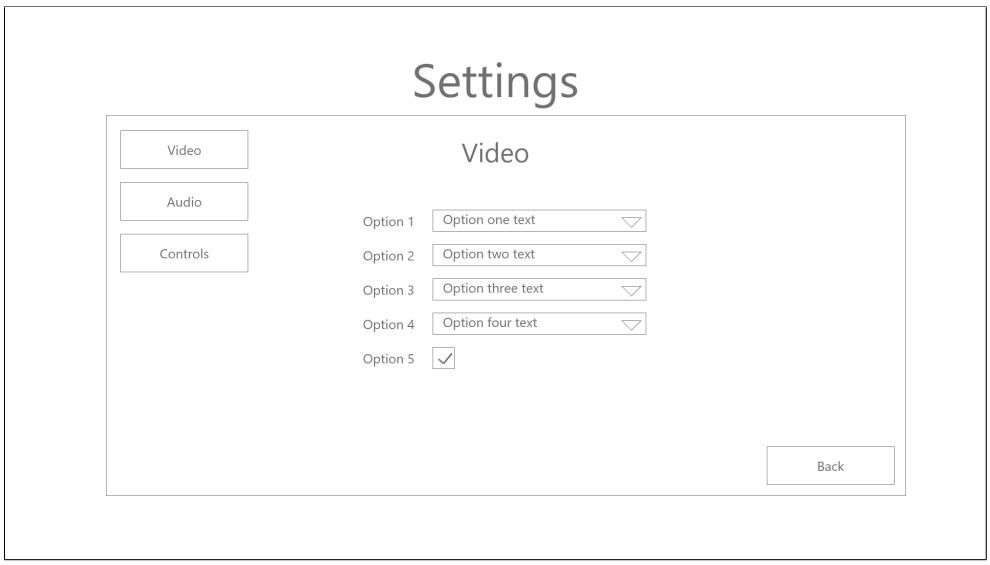
Příloha B

Obrázky návrhu UI

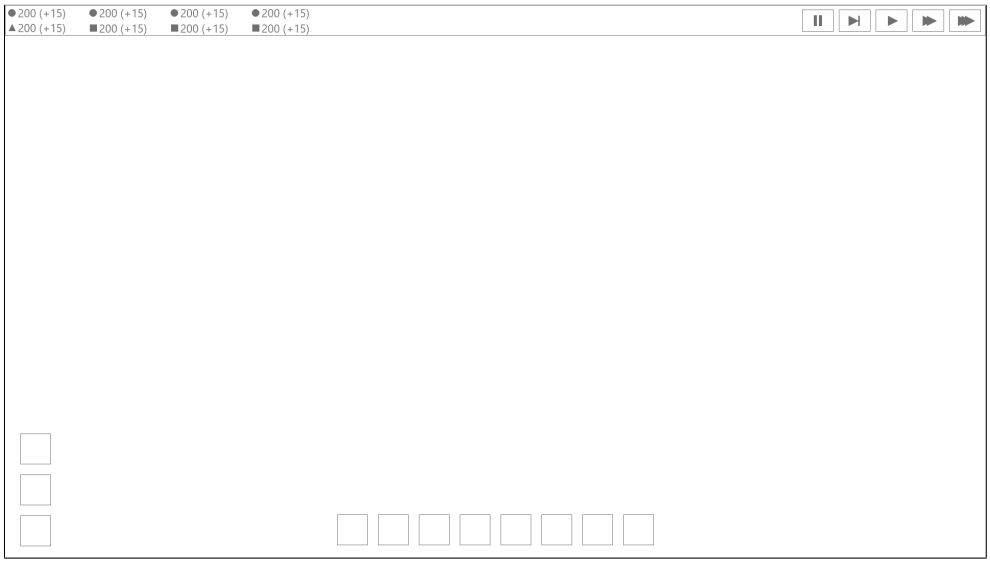
Obrázek B.1: Vzhled hlavního menu v prototypu hry

Obrázek B.2: Návrh hlavního menu

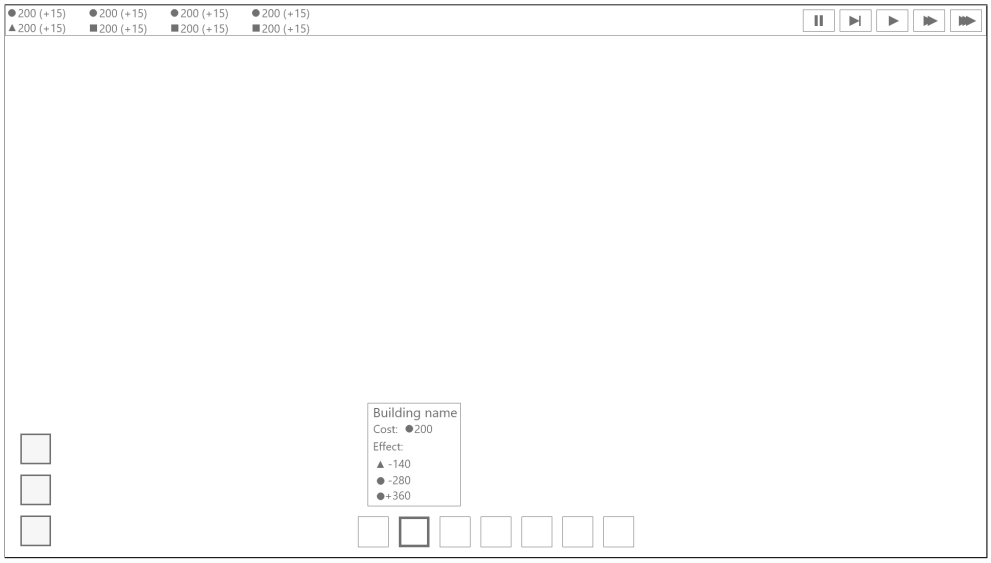
Obrázek B.3: Návrh rozhraní pro spuštění nové hry



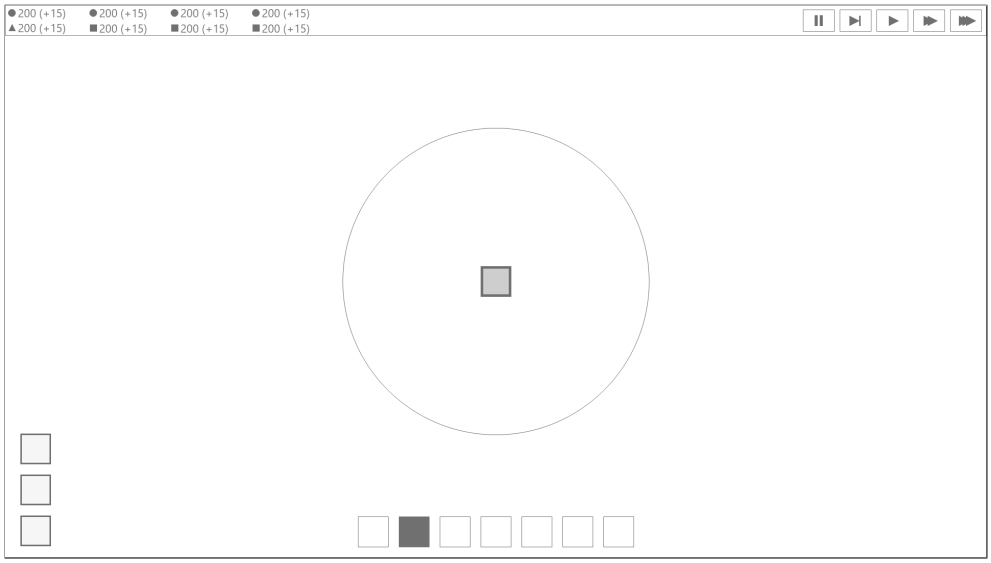
Obrázek B.4: Návrh obrazovky s nastavením



Obrázek B.5: Návrh rozhraní v hlavním herním režimu

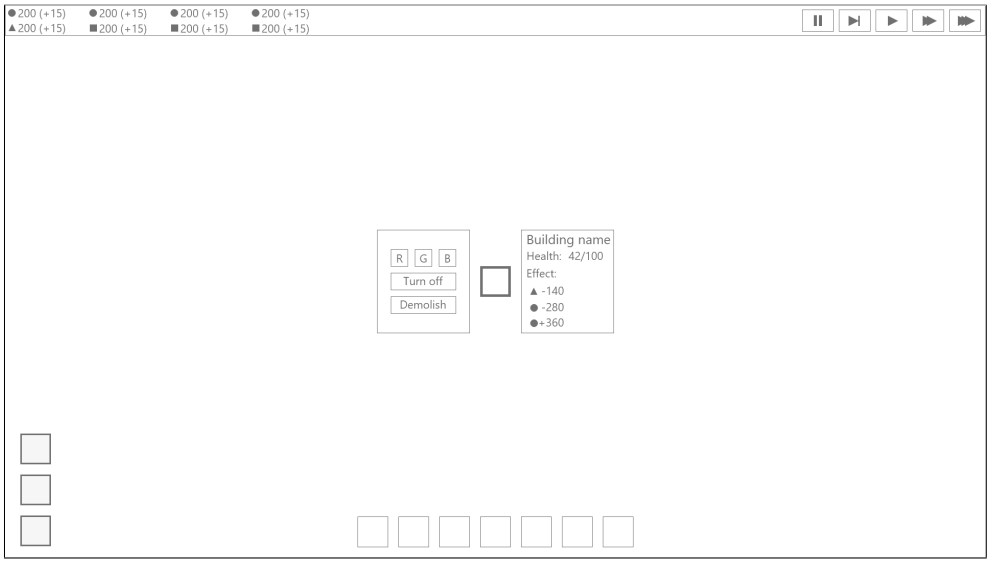


Obrázek B.6: Návrh rozhraní v hlavním herním režimu – informace o budově



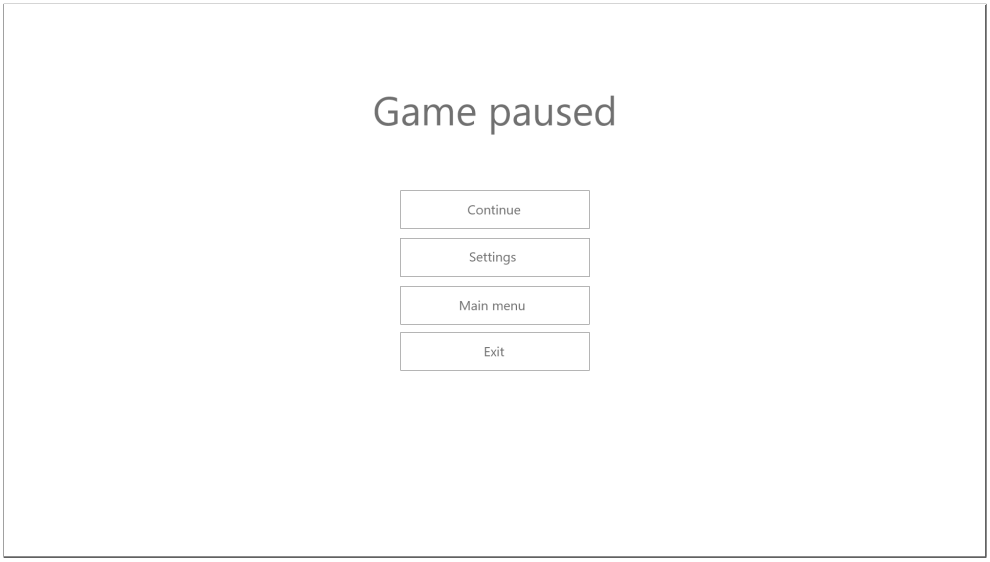
Obrázek B.7: Návrh rozhraní v hlavním herním režimu – náhled při stavění budovy

Obrázek B.8: Vzhled rozhraní v prototypu hry při stavění budovy

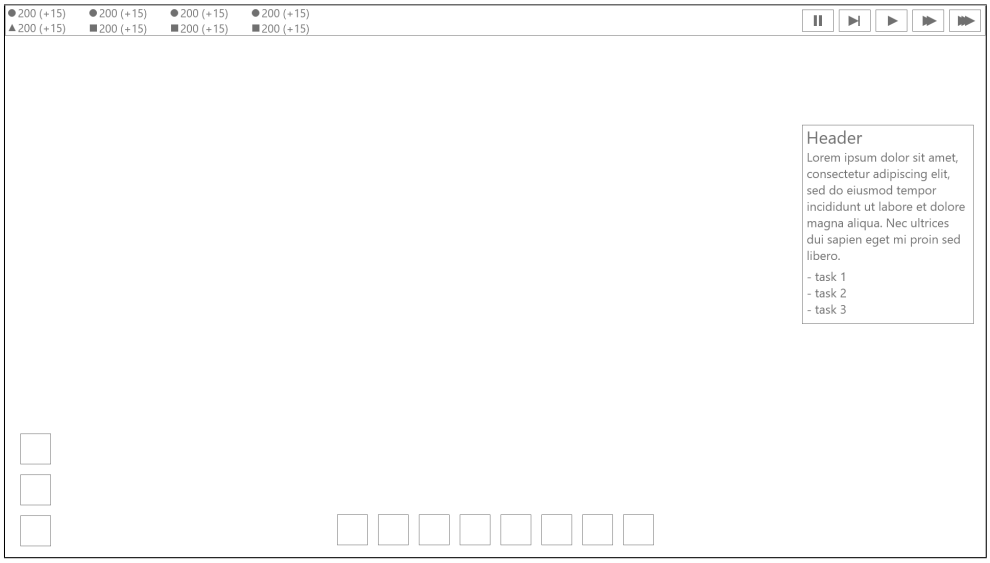


Obrázek B.9: Návrh rozhraní v hlavním herním režimu – rozhraní pro interakci s budovou

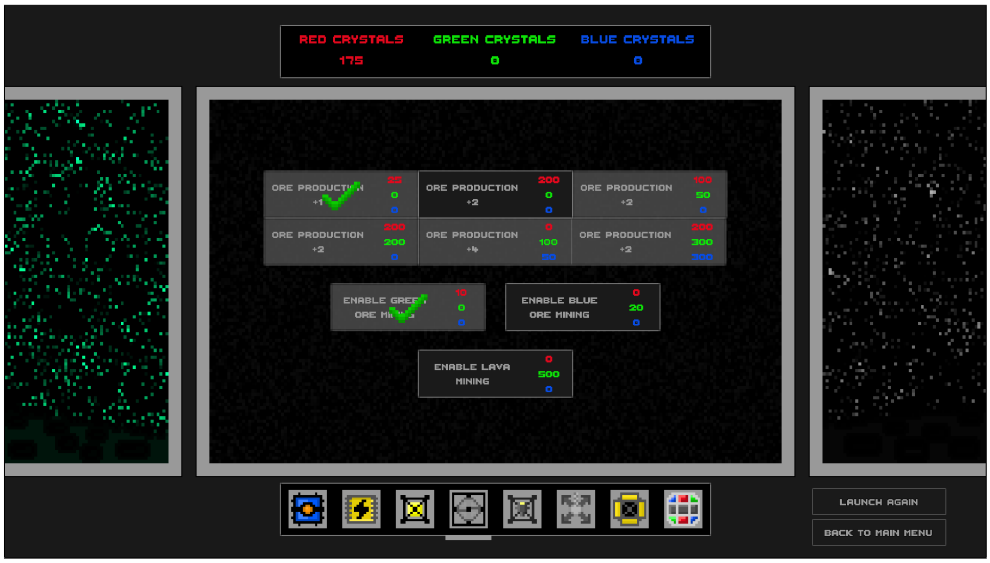
Obrázek B.10: Vzhled rozhraní v prototypu hry při zvolení postavené budovy



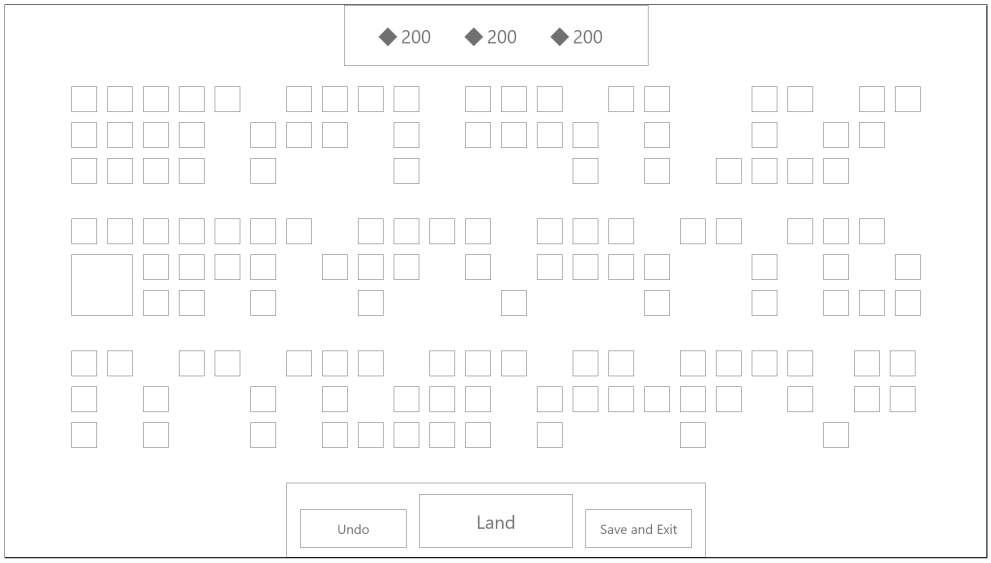
Obrázek B.11: Návrh rozhraní při pozastavené hře



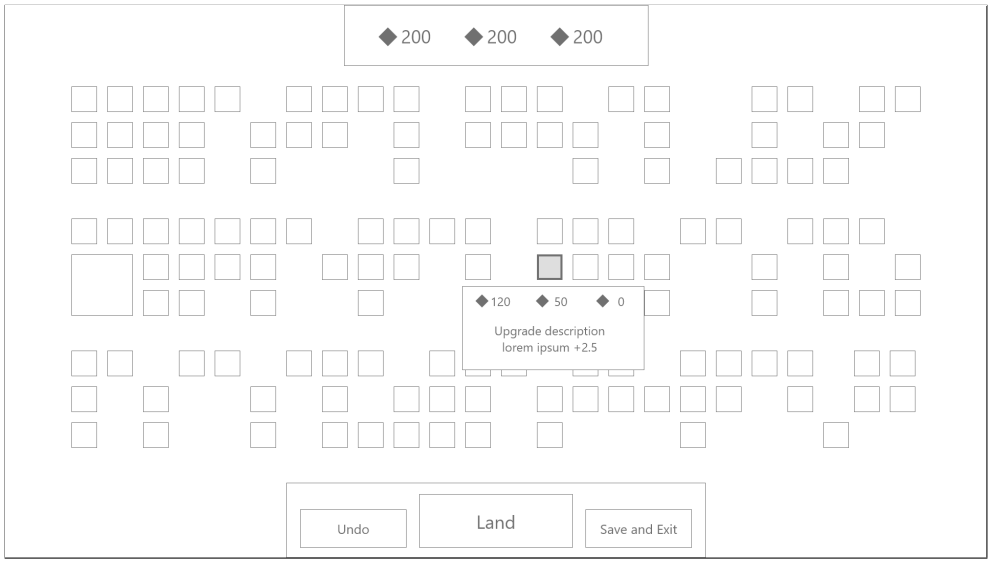
Obrázek B.12: Návrh rozhraní tutoriálu



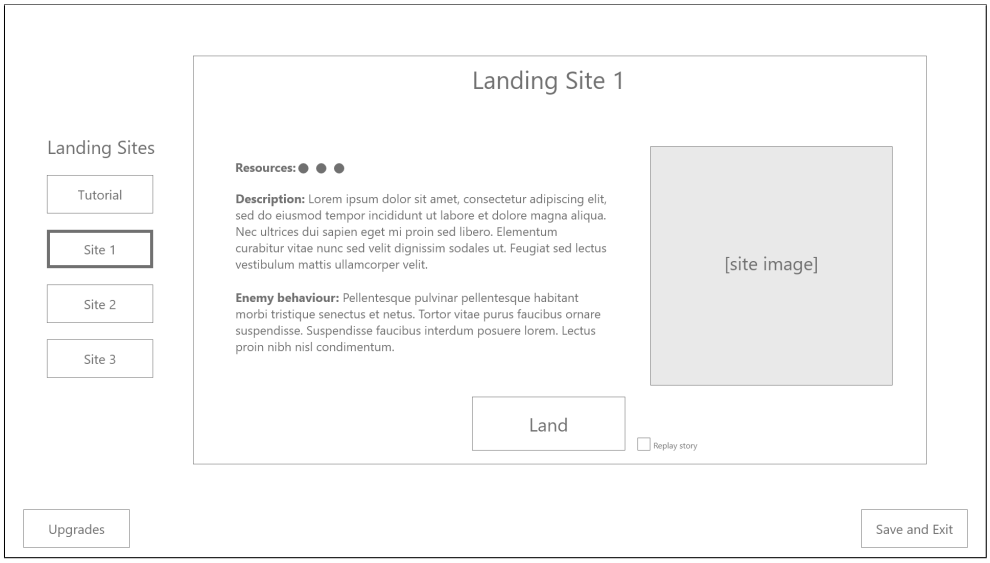
Obrázek B.13: Vzhled obrazovky s vylepšeními v prototypu hry



Obrázek B.14: Návrh obrazovky s vylepšeními



Obrázek B.15: Návrh obrazovky s vylepšeními – informace o vylepšení



Obrázek B.16: Návrh obrazovky pro výběr úrovně

Obrázek B.17: Návrh obrazovky s příběhem

Obrázek B.18: Návrh obrazovky s titulky

Příloha C

Obsah přiloženého média

- `readme.txt` ..... stručný popis obsahu média
- `RoguePlanet_v1.0.6` ... adresář se spustitelnou formou hry pro Windows
- `RoguePlanet_UnityProject` ..... adresář s Unity projektem hry
- `DP-strnalad` ..... adresář se zdrojovou formou práce ve formátu LaTeX
- `DP-strnalad.pdf` ..... text práce ve formátu PDF