UNIVERZITA PALACKÉHO V OLOMOUCI
KATEDRA MATEMATICKÉ INFORMATIKY

ROČNÍKOVÝ PROJEKT

Hra Agresoři

Srpen 2004

Pavel Kubát, Přemysl Šrubař
Informatika, III. ročník

Abstrakt

Tématem tohoto projektu je realizace tahové strategie Agresoři. Hru lze hrát jak proti počítači, tak proti jinému hráči a to jak na jednom počítači, tak i na více počítačích propojených síťí.
Samotná aplikace je tedy typu klient - server a všechny její výstupy kromě komunikace po síti jsou ve formátu XML.
Aplikace byla napsána v v jazyce C# ve vývojovém prostředí Microsoft Visual .Net 2003. Program je určen pro operační systémy Windows 2000/XP.

Obsah

1 Specifikace a analýza zadání

1.1 Pravidla hry

Hra probíhá na libovolně velké hrací ploše, která je rozdělena na políčka různého typu (terénu). Každé políčko vlastní jeden hráč, s vyjímkou terénu moře, které nemá ve vlastnictví ani jeden z hráčů. Každý hráč má na svých políčkách možnost zlepšovat svůj průmysl stavbou budov, vynalézat novější a lepší zbraně, vyrábět válečné jednotky, měnit svůj typ vlády a vést diplomatické rozhovory s jinými hráči. Cílem hry je získat všechny políčka ostatních hráčů a tím je vyřadit ze hry. Vyhravá ten hráč, který zůstal na hrací ploše poslední.
Podrobnější pravidla - viz Příloha A.

1.2 Minimální požadavky

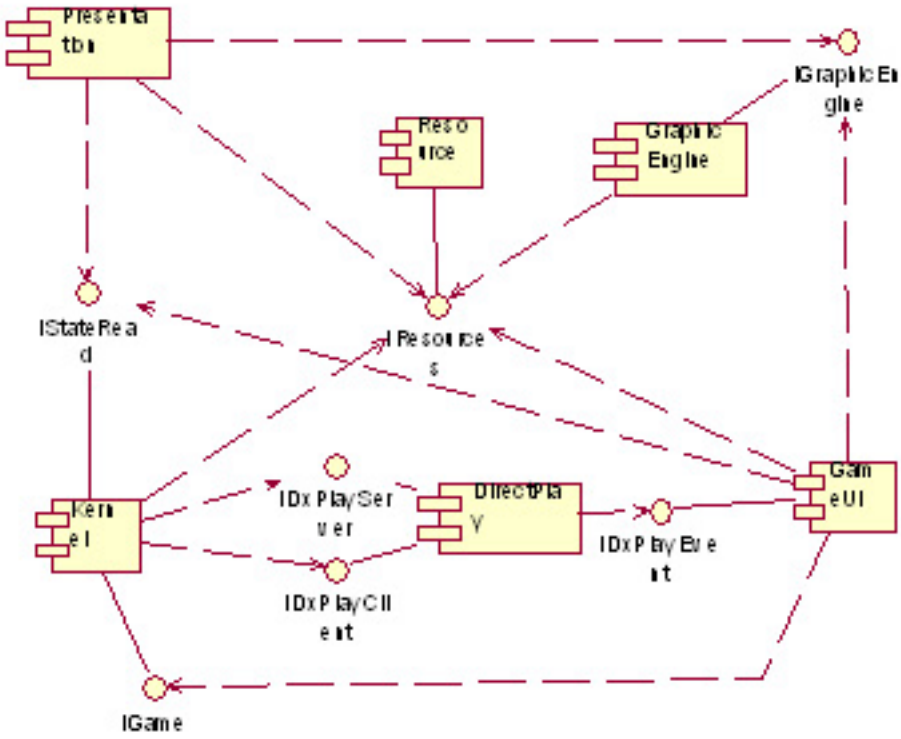
- Funkčnost aplikace
- Základní grafické rozhraní, které dokáže zobrazit hrací plochu a rozmístění jednotek
- Možnost vytvoření nové hry
- Kontrola jednotlivých tahů hráče podle daných pravidel hry (zabudování logiky hry)
- Možnost volby soupeřů, tj. výběr mezi hráčem a počítačovým hráčem
- Základní algoritmy, podle kterých dokáže počítačový hráč vytvářet nové tahy
- Uložení a znovuotevření rozehrané hry
- Možnost kdykoliv program i hru ukončit
- Výstup aplikace ve standartním formátu, nejlépe XML

1.3 Maximální požadavky

- Rozhraní důstojné pro veřejnou propagaci programu
- Aplikace typu server - klient
- Hru je možno hrát po síti a to nejen mezi více hráči,ale i s počítačovým hráčem, který je umístěn na serveru
- Nápověda obsahující pravidla hry, způsob ovládání programu a informace o programu
- Možnost zvolit styl zobrazení (hrací deska i figurky mají více podob)
- Několik stupňů inteligence počítačového hráče
- Několik algoritmů počítačového hráče
- Nápověda k ovládání uživatelského rozhraní
- Podpora více jazyků v programu
- Ošetření špatných vstupů a výpis chybových hlášení
- Dokumentace k celému programu
- Editor dat aplikace

1.4 UML diagramy a popis komponent

Prvotní návrh prošel řadou změn a nyní je aplikace tvořena šesti komponentami.



Obrázek 1: Diagram komponent

Presentation – Zobrazuje hrací plochu a jednotky

GraphicEngine – Tvoří grafické prvky hry

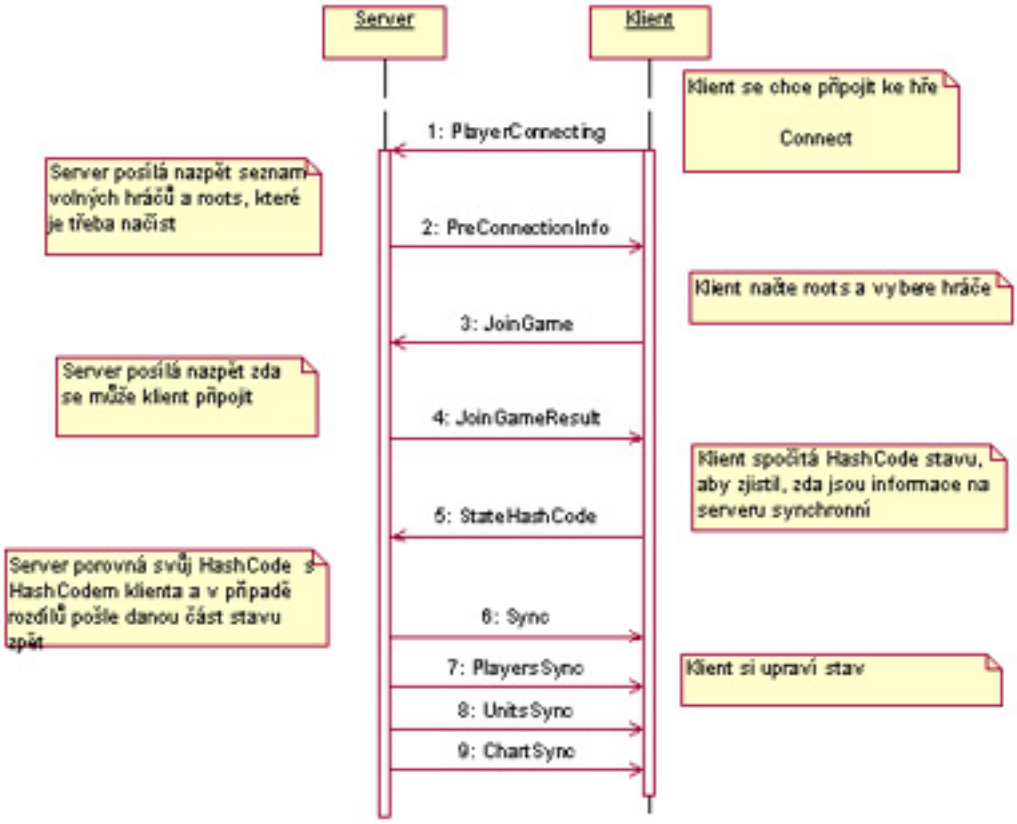
Kernel – Kontroluje správnost tahů, vytváření tahů počítačem a zodpovídá za stav hry

DirectPlay – Zodpovídá za komunikaci hry po síti

GameUI – Vytváří rozhraní pro komunikaci s hráčem

Resource – Zodpovídá za načítání a ukládání hry a přístup ke zdrojům

Sekvenční diagram popisující průběh stěžejních akcí při připojování k serveru



Obrázek 2: Připojení k serveru

2 Použité algoritmy

2.1 DefaultAI

Do aplikace je vložena implicitní počítačová inteligence, která se chová podle následujících algoritmů :

2.2 Informace ze stavu

Na začátku každého kola počítačový hráč (dále jen PH) zjistí určité informace ze stavu, které jsou pro něj důležité. K nim patří informace o tom, se kterými hráči sousedí (ať už na souši nebo do jisté míry i přes moře), hranice, které tvoří jeho stát, indexy celkové síly každého hráče (po diplomatické stránce, po stránce připravenosti na válku a po stránce síly jeho státu).
Z těchto informací si zmapuje situaci, ve které se nachází - tj. jak nebezpeční sousedé jsou a jak bezpečně má zajištěny hranice.

2.3 Diplomatické nóty

Z informací získaných ze stavu již ví, jak nebezpeční jeho sousedé jsou a kterým je potřeba se bránit, případně naopak nabízet mír.
Podle těchto informací a podle toho, jak s daným soupeřem vychází, se rozhoduje zda navrhne určité řešení (tj. uzavření míru, uzavření aliance, vypovězení války atd..), popřípadě na podobný návrh od soupeře zareaguje, pokud mu byl zaslán.

2.4 Algoritmus na stavbu budov

K zjišťění tahu, který PH provede se používají tzv. cíle. Cíl má určitou prioritu, aktéra a objekt.
PH prochází svůj stát a postupně přidává další cíle, které lze v daném tahu vystavět. K cílům patří jak stavba nových budov, tak oprava zničených budov nebo jejich vylepšení. Pro každou situaci má PH určité ohodnocení jednotlivých cílů tak, aby cíle, které nejsou v danou chvíli nutně potřeba vyplnit byly umístěny prioritně pod cíly které jsou akutní.
Např. ve chvíli, kdy z informací ve stavu PH zjistí, že má odkryté hranice proti soupeři, který mu vyhlásil válku, do seznamu cílů jsou přidány cíle na stavbu obranných bunkrů a děl v blízkosti hranic a jsou prioritně nad výstavbou průmyslu nebo zlepšováním již postavených budov.
Ve chvíli, kdy je seznam cílů hotov, vybere PH cíl s nejvyšší prioritou a dá ho stavět. Pokud ke stavbě z nějakého důvodu nedojde, cíl odkládá a zkouší vystavět následující. Pokud se stavba neprovede z důvodu nedostatku financí, je výstavba ukončena a zbytek zdrojů ušetřen do příšťího tahu, aby akutní cíl mohl být příště vystaven.

2.5 Algoritmus na pohyb jednotek

K zjišťování tahu jednotlivých figur se používá v zásadě stejný princip jako v algoritmu stavby budov.
Pro každou jednotku PH vygeneruje určitý seznam cílů s určitými prioritami. Největší priorita je dávána informacem ze stavu a celostátní strategii - tj. obrany hranic, míst útoku atd. . . , následně pak cílům lokálním - tj. obsazení cizí budovy, města, zničení nepřátelské jednotky, obrany vlastního státu tam, kde jeho důležitost zasluhuje více obranny nebo možnost nalodění se do transportu jednotek. Okolí jednotky je spirálovitě prozkoumáváno a tam, kde je některý z cílů nalezen, je přidán do seznamu cílů. Z nich je potom (stejně jako v algoritmu na stavbu budov) vybrán ten

s největší prioritou.
Pro cíl s největší prioritou je od pozice jednotky vybrána nejkratší možná cesta (kterou jednotka skutečně může projít) k místu cíle, a touto cestou se jednotka vydává. Pokud jednotka nenalezne vhodnou cestu k danému cíly, je cíl vynechán a jednotka se podobným způsobem zajímá o cíl následující.

3 Uživatelská dokumentace

Aplikace samotná nabízí dvě samotné části, které může uživatel používat.
První z nich je samotná hra, druhou je pak editor, kterým se dají veškeré zdroje hry prohlížet a ve většině případů i editovat.

Ihned po spuštění aplikace - hry se před uživatelem objeví úvodní obrazovka s hlavním menu celé hry.

Obrázek 3: Hlavní menu hry

Toto menu zpřístupňuje uživateli tyto možnosti :

Začít novou hru — vytvoření nové hry z vybraného scénária. Uživatel hraje pouze s počítačovými hráči, případně s dalšími uživateli, kteří hrají na stejném počítači

Síťová hra — vytvoření nové hry z vybraného scénária. Uživatel má možnost se jak připojit k již existujícím hrám - pokud je mu to dovoleno, tak si vytvořit hru novou. Může hrát jak s jinými uživateli na jiných počítačích propojených síťí, tak i s počítačovými hráči

Nahrát hru — načtení dříve uložené hry

Nastení grafiky — možnost zvolení grafického módu

Uložit hru — uložení aktuálně běžící hry

Konec — ukončení aplikace

Zpět — návrat do předchozí situace

3.1 Začít novou hru

Při rozhodnutí uživatele začít hrát hru na lokálním počítači pouze s počítačovými hráči (nebo s hráči se kterými se bude u počítače střídat) je nejdříve potřeba zvolit, které z daných scénářií a za který ze států ve scénáriu chce uživatel hrát. Po zvolení těchto možností se otevře uživateli samotné hrací okno celé hry.

Obrázek 4: Hrací okno

Ovládání hry je velmi intuitivní a jednoduché. Napravo jsou zobrazena dvě menu, ve kterých může uživatel vybírat možnosti, které se mu v dané situaci nabízí.

Menu hry

Na prvním z menu se nachází tyto možnosti :

Hlavní menu — návrat do hlavního menu

Konec kola — ukončení aktuálního kola uživatele

Stát — zobrazení, který ze států (hráčů) je v danou chvíli na tahu. Při kliknutí na tuto možnost se uživateli zobrazí jeho diplomatické možnosti - viz níže

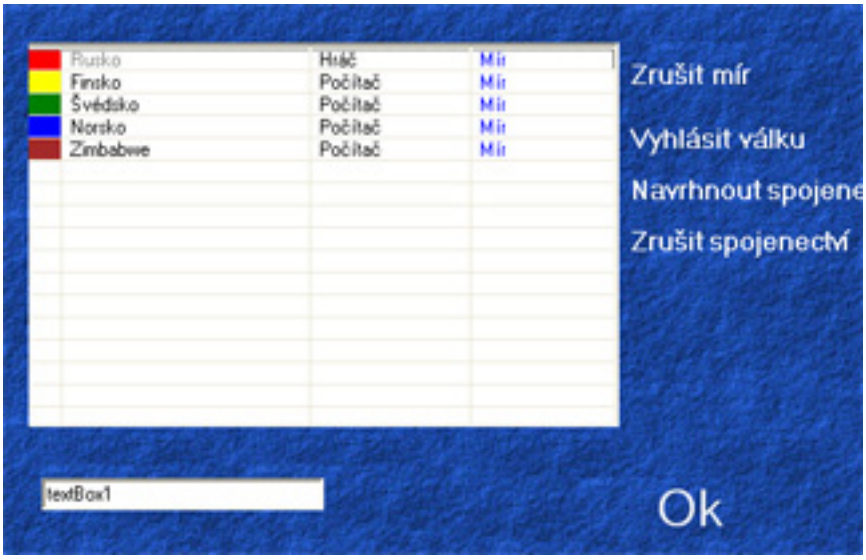
Změna administrace — možnost změnit státní správu svého státu - viz níže

Stavění — možnost vystavit některou z nových budov - viz níže

Výzkum — možnost vyzkoumat novou technologii - viz níže

Stát

Při kliknutí na možnost „Stát“ se uživateli otevře následující okno :



Obrázek 5: Diplomatické okno

Takto vypadající okna uživatele provází po celou dobu hry. Většinou mají tyto okna dvě tlačítka - OK a Zpět. Tlačítkem OK se možnost v daném okně potvrzuje, tlačítkem Zpět se daná možnost ruší.
Důležitá věc je, aby uživatel věděl, že zobrazení určité možnosti v daném okně ještě stoprocentně neznamená, že může dané možnosti využít. Znamená to pouze, že pokud uživatel splní všechny podmínky, které na něj jsou kladeny, pak může být tah proveden.

V okně „stát“ má uživatel možnost zhlédnout aktuální stav jeho diplomatických vztahů, zjistit situaci obsazení hráčů a případně vyslat diplomatické zprávy jinému hráči.
Při vyslání diplomatické zprávy je potřeba pouze označit příslušného hráče a vybrat některou z možností napravo. Pro opuštění diplomatického okna a návrat zpět slouží tlačítko OK.

Změna administrace

Při kliknutí na možnost změny administrace se uživateli otevře okno podobné oknu předchozímu, ale s tím rozdílem, že se místo výpisu jednotlivých hráčů vypíší do okna všechny, pro uživatele možné, změny administrace. Po kliknutí na vybranou změnu administrace a potvrzení kliknutím na OK se uživateli změní původní administrace na nově zvolenou.
Pokud chce vše nechat tak, jak je, stačí kliknout na Storno, čímž opusťí Diplomatické okno a vrátí se zpět do hry.

Stavění

Při kliknutí na tlačítko stavění se uživateli ukáže znovu podobné okno jako v předchozích případech, s tím rozdílem, že místo daných výpisů se v něm zobrazí seznam

budov, které může uživatel v danou chvíli vystavět.
Po zvolení některé z nabízených budov se okno znovu přepne do hracího okna, kde je třeba vybrat místo, kam bude budova vystavena. Při přejíždění jednotlivých políček na mapě se uživateli zobrazuje kurzor, který mu pomáhá dané místo najít. Při kliku na místu, kde budova může být vystavena se začne budova stavět (pokud má uživatel dostatek zdrojů na stavbu, atd...) a hra se vrací do normálního režimu.

Výzkum

Při kliknutí na možnost výzkumu se uživateli otevře okno s výpisem všech technologií, které lze v danou chvíli zkoumat. Při kliku na jednu z možností a potvrzení rozhodnutí tlačítkem OK se technologie začíná zkoumat.

Menu jednotky

Druhým menu na hrací ploše je menu jednotky. V tomto menu se vždy zobrazí možnosti, které aktuálně vybraná jednotka může provést.

V některých případech, kdy je potřeba na mapě pro úplnost zadaného tahu vybrat ještě některou doplňující informaci se uživateli zobrazují pomocné kurzory, které mu pomáhají danou informaci snadněji najít.

Během hry lze tedy shlédnout v menu tyto možnosti :

Posun jednotkou — zobrazuje se pro všechny pohyblivé figurky. Při kliknutí na zadaný směr se jednotka pokusí na dané místo přesunout

Útok — zobrazuje se pro jednotky, které mohou útočit

Vyrábět — zobrazuje se pro tzv.„výrobce“, kteří mohou vyrábět jiné jednotky. Při kliknutí začne daný výrobce vyrábět

Produkce — zobrazuje se pro výrobce. Při kliku se otevře okno, ve kterém je možné změnit výrobní model výrobce

Vyložit — zobrazuje se pro naloděné jednotky. Při kliknutí se snaží vyložit

Vyložit vše — zobrazuje se pro jednotky, ve kterých mohou být naloženy jiné jednotky. Při kliknutí se snaží svůj obsah vyložit

Vstoupit — zobrazuje se pro všechny pohyblivé jednotky. Při kliknutí se hledá jednotka, do které by mohla vstoupit

Drancovat — zobrazuje se pouze pro pohyblivé jednotky, které mohou drancovat. Při kliknutí a vybrání určitého města se snaží město vyplenit

Rozšířit — zobrazuje se pro budovy a města, které mohou být rozšířeny. Při kliknutí se zobrazí okno, ve kterém je možné vybrat typ rozšíření

Vylepšit — zobrazuje se pro jednotky, které mohou být vylepšeny na novější model.

Opravit — zobrazuje se pro jednotky, které mohou být opraveny

3.2 Síťová hra

Pokud chce uživatel hrát hru po síti s ostaními hráči, pak je třeba zvolit možnost síťové hry.
Po jejím zvolení se uživateli zobrazí okno, ve kterém je třeba vybrat typ připojení, které bude použito pro hru a které je možno na daném počítači využít.

Po jejím zvolení se pak zobrazí následující okno:

Obrázek 6: Okno výběru serveru

V něm je možné po kliknutí na „Založit nový server“ založit hru novou a tím se stát serverem, nebo je možné se po kliknutí na danou hru v seznamu her a potvrzením tlačítkem OK připojit ke hře již existující.
Seznam aktuálně hraných her se dynamicky aktualizuje. Aplikace sama najde všechny hry existující na lokální síti. Pokud chce uživatel hrát hru po internetu, je třeba adresu serveru vložit do příslušného boxu „IP adresa“ a zvoli správný port.
Po potvrzení daného okna a navázání spojení se uživateli zobrazí stejné okno Výběru státu jako při výběru nové lokální hry.
Po tomto výběru se pak pro server nabízí okno, ve kterém lze nastavit serverovou hru podle uživatelova přání (tj. nastavit, za které hráče mohou hrát jiní užitelé a za které hrát nemohou), případně se čeká na připojení ostatních uživatelů na nově založený server.
Pro připojujícího se uživatele se již další okna nezobrazují a namísto toho se synchronizují stavy s aktuální situací hry.

3.3 Nahrát hru

Pokud chce uživatel začít hrát již uloženou hru, pak je třeba zvolit tuto možnost.
Při jejím výběru se zobrazí seznam uložených her. Po kliknutí na některou z nich se hra nahraje a stav se aktualizuje podle daných dat.

3.4 Uložení hru

Pokud si chce uživatel aktuálně hranou hru uložit a nechce ji hrát znovu od původního stavu, pak je třeba zvolit uložení hry.
Při uložení může vybrat některou z již uložených her a přepsat ji, nebo je možné zadat název nový, který bude uložen do uživatelem zadaného místa.
Uživatel ovšem musí brát zřetel na to, že aplikace dokáže najít pouze takové uložené hry, které se nachází v některém poddadresáři dané hry a to jen ty, které začínají slovem „Save“.

3.5 Nastavení grafiky

Pokud se uživateli nelíbí aktuální nastavení rozlišení, barev, případně požaduje změnit zobrazový adaptér, pak při kliknutí na možnost nastavení grafiky se tyto vlastnosti dají změnit.

3.6 Konec

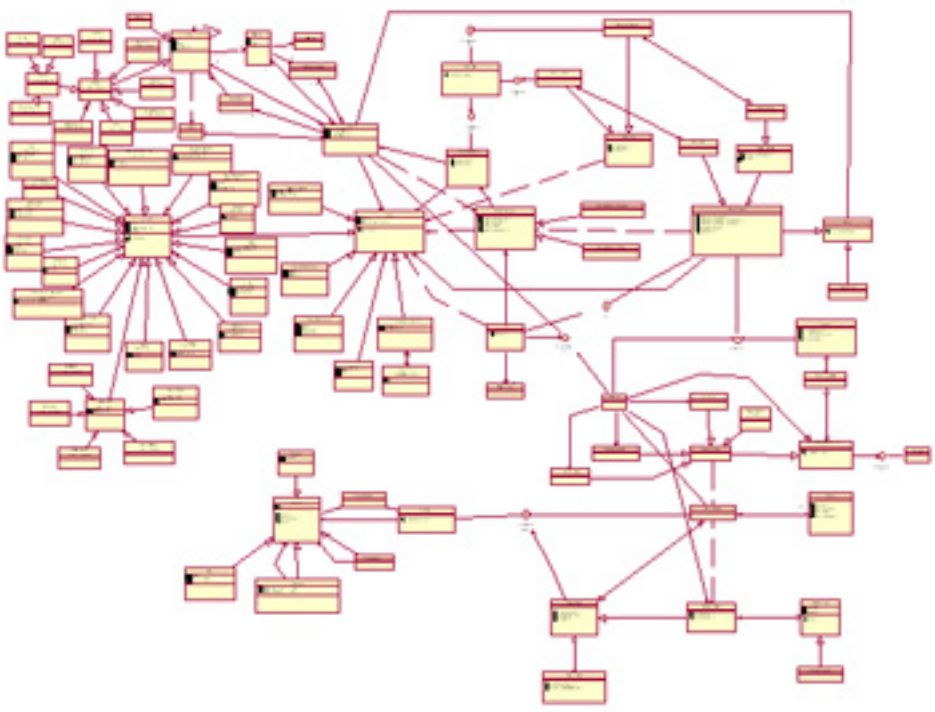
Vypnutí celé aplikace.

3.7 Zpět

Při zvolení této možnosti se uživatel vrátí do předchozí situace.

4 Programátorská dokumentace

Aplikace se skládá z šesti komponent, z nichž jedna je součástí DirectX.



Obrázek 7: Kompletní diagram hlavních tříd

4.1 Kernel

State

Popis

Uchovává obecné informace o stavu hry. Ostatní komponenty se na ni odkazují přes interface IStateRead

Data

```
UnitCollection units -- Kolekce jednotek ve hre
UnitModelCollection models  -- Kolekce modelu nacteného scénária
PlayerCollection players -- Kolekce hrácu ve hre
AdministrationCollection administrations -- Kolekce administrací scénaria
Chart chart -- mapa hry
UnitFactory unitFactory -- Továrna vytvárející tahy
int actualPlayerId -- Identifikacní císlo práve hrajícího hráce
int localPlayerId -- Identifikacní císlo hráce hrajícího na lokálním pocítaci
```

Hlavní metody

```
void Turn() -- Prepne aktuálního hráce na následujícího
```

```
void FillMemento(ref Mem.PatchType memento) -- Uložení stavu do mementa
void FillFromMemento(Mem.PatchType memento) -- nactení stavu z mementa
void Load(PatchEventArgs e) -- Nactení stavu
void Save(PatchEventArgs e) -- Uložení stavu
```

Unit

Popis

Základní abstraktní třída jednotek, ze které všechny ostatní jednotky dědí. Data obsahují informace, které se mění behem hry a nemohou být tím pádem načteny z UnitModelu.

Data

```
int id -- Identifikacní císlo jednotky
Position position -- Pozice jednotky
UnitModel model -- Model jednotky
int inBuild -- Délka trvání stavby jednotky
int abroadIn -- Identifikacní císlo jednotky, ve které je jednotka nalodena
int movement -- Aktuální pocet pohybových bodu
int attack -- Aktuální pocet útocných bodu
int defense -- Aktuální pocet obranných bodu
int ownedBy -- Identifikacní císlo hráce, který jednotku vlastní
UnitCollection unitsInside -- Kolekce jednotek nacházející se uvnitr jednotky
int freeCapacity -- Volné místo v jednoce
INNOVATION innovations -- Vystavené inovace jednotky
int currentPower -- Aktuální síla jednotky
string name -- Jméno jednotky
int numberOfShots -- Aktuální pocet volných strel
string buildModel -- Jméno modelu, který jednotka staví
bool buildNow -- Príznak, zda jednotka staví
```

Hlavní metody

```
virtual Unit FirstCollisionAt(Position at) -- Overení kolize jednotek
void FillMemento(ref Mem.DataUNIT data) -- Uložení jednotky do mementa
void FillFromMemento(Mem.DataUNIT data) -- nactení jednotky z mementa
virtual RESULTS TryDeployNear(Position pos, int maxRadius) -- pokus o vyložení
virtual RESULTS MoveInDirection(DIRECTION direction) -- Pohyb jednotkou
virtual RESULTS Upgrade() -- Vylepšení jednotky
virtual RESULTS CanBeUpgraded(int conteinerId) -- Pokusí se vylepšit jednotku
virtual RESULTS CanBeAt(Position at) -- Možnost výskytu jednotky na políčku
virtual RESULTS TouchUp(INNOVATION innovation) -- Rozšírení jednotky
virtual RESULTS CanUnitEnter(Unit u) -- Možnost vstupu jednotky do jiné
virtual RESULTS CanBuildUnit(Unit u) -- Možnost stavby zadané jednotky
virtual RESULTS CanAttackUnit(Unit u) -- Možnost útoku na zadanou jednotku
virtual RESULTS AttackByUnit(Unit attacker, out int attackPower) -- Útok
virtual RESULTS StartBuildUnit(Unit u) -- Stavba jednotky uvnitr zadané
virtual RESULTS LetUnitEnter(Unit u) -- Nechá zadanou jednotku vstoupit
virtual RESULTS CanBeUnloaded() -- Možnost vyložení jednotky
virtual RESULTS CanBePillaged(Unit unit) -- Možnost vydrancování jednotky
virtual RESULTS Unload() -- Vyloží se z jednotky
virtual RESULTS UnloadUnits() -- Vyložení všech jednotek
virtual void BeginTurn() -- Pocátek nového tahu
```

Chart

Popis

Uchovává informace o terénu, nerostných zdrojích a vlastnících políček. Singletone.

Data

```
ChartSquare[,] chart -- Dvojrozmerné pole uchovávající informace o mape
Position maxPosition -- Maximální pozice mapy
```

Hlavní metody

```
Position GetNext(Position at) -- Vrací následující posici na mape
void ChangeOwner(Position at, int newOwner) -- Zmení vlastníka políčka
bool TerrainNearPlace(Position at,int maxRadius,TERRAIN ter)
void FillFromMemento(Mem.PatchType memento) - Nactení mapy z mementa
void FillMemento(ref Mem.PatchType memento) - Uložení mapy do mementa
```

UnitModel

Popis

Uchává informace o neměnných nebo puvodních vlastnostech modelů. Invariantní.

Data

```
string id -- Identifikacní retezec modelu
UNIT_TYPE unitType -- Typ modelu
int buildTime -- Cas potrebný k výrobe modelu
int unitCapacity -- Kapacita modelu
int unitSize -- Velikost modelu
int movement -- Pohybové body modelu
int attack -- Útocné body modelu
int defense -- Obranné body modelu
int attackRange -- Dostrel modelu
ALTITUDE attackAltitude -- Nadmorská výška, do které lze útocit
ALTITUDE defenseAltitude -- Nadmorská výška, které se lze bránit
ALTITUDE altitude -- Nadmorská výška modelu
ALTITUDE containerForAltitude -- Nadmorská výška modelu schopných vejít
UNIT_CLASS containerForClass -- Trída modeluk, které dokáže naložit
UNIT_CLASS buildingClass -- Trída modeluk, které dokáže vystavet
UNIT_CLASS unitClass -- Trída modelu
UNIT_CLASS attackClass -- Trída modelu, na které dokáže zaútocit
UNIT_CLASS developingClass -- Trída modelu, které dokáže zkoumat
string[] needTechnogy -- Technologie nutné k vystavení modelu
int developmentTime -- Cas potrebný k vynalezení modelu
string upgradeModel -- Identifikacní retezec vylepšeného modelu modelu
int upgradingTime -- Cas potrebný k vylepšení na nový model
int fuel -- Velikost nádrží modelu
Cost buildCost -- Zdroje potrebné k výrobe modelu
Cost upgradeCost -- Zdroje potrebné k vylepšení modelu
int numberOfShots -- Pocet strel modelu
INNOVATION inovations -- Možné vylešení modelu
int developLevel -- Trída vynálezu
TerrainBehavior[] terrains -- Pole chování modelu na jednotlivých terénech
```

Hlavní metody

```
void FillMemento(ref Mem.DataUNIT_MODEL data) -- uloží model do mementa
void FillFromMemento(Mem.DataUNIT_MODEL data) -- nacte model z mementa
```

Move

Popis

Základní abstraktní třída tahu jednotek. Všechny ostatní tahy z ní děďí. Pomocí jí se předávají všechny informace o tazích hráčů. Zděděné třídy si přidávají potřebná data a překrývají nejduležitejší metodu DoMove(), která uskutečnuje daný tah pomocí stavu

Data

```
int id -- Identifikacní císlo zprávy
int playerFromId -- Identifikacní císlo hráce, od kterého zpráva pochází
RESULTS result -- Výsledek provedení tahu
```

Hlavní metody

```
override void Execute(State state) -- Vykoná tah
abstract RESULTS DoMove(State state) -- Samotné vykonání tahu
```

Msg

Popis

Základní abstraktní třída všech tahů ve hře. Všechny ostatní z děďí

Data

```
MSG_TYPE type -- typ zprávy
int playerFromNetId -- Identifikace místa odkud zpráva prišla
```

Hlavní metody

```
virtual void Dispatch(BossGame boss) -- Odbavení zprávy. Volá Execute()
virtual void Execute(State state)
```

Player

Popis

Udržuje informace o hráči

Data

```
int netId --  Identifikace místa, kde je hrác
int id -- Identifikacní císlo hráce
Cost resources -- Zdroje hráce
TechnologyCollection technologies -- Kolekce Technologií vyzkoumaných hrácem
ReportCollection reports -- Reporty urcené pro hráce
PLAYER_TYPE playerType -- Typ hráce
Administration myAdministration -- Aktuálne zvolená administrace hráce
int newSteel -- Aktuální výroba hráce
```

```
int invitation -- Aktuální výzkum hráce
Relation[] relations -- Pole obsahující vztahy k ostatním hrácum
string name -- Jméno hráce
int inBuild -- Pocet tahu potrebných ke zmene administráce hráce
```

Hlavní metody

```
int Develop(string modelId, int amount) -- Vyzkoumání modelu
bool CanDevelopModel(UnitModel m) -- Možnost výzkumu modelu
void ComputeSteelsAndInvitations() -- Propocítá nove pridanou ocel
RESULTS ChangeAdministration(string newName) -- Zmení administraci hráce
void HurryBuild(int plus) -- Zrychluje zmenu administrace
void FillFromMemento(Mem.DataPLAYER data) -- nacte hráce z mementa
void FillMemento(ref Mem.DataPLAYER data) -- uloží hráce do mementa
```

Ai

Popis

Abstrakntí třída počítačové inteligence. Ostatní z ní dědí. Strategy.

Data

```
MoveFactory moveFactory -- Továrna na tahy
IStateRead stateRead -- Interface state
int myId -- Identifikacní císlo pocítacového hráce
UnitFactory unitFactory -- Továrna na jednotky
```

Hlavní metody

```
abstract void StartTurn() -- Startovací procedura AI
virtual void InicializeAi(MoveFactory moveFactory, IStateRead stateRead)
```

DefaultAI

Popis

Třída zděděná z AI. Implicitně vložený počítačový hráč

Data

```
ArrayList coasts -- Seznam pobrežních pozic hráce
ArrayList ironOres -- Seznam nevyužitých železných rud hráce
ArrayList ironMines -- Seznam nevyužitých dolu hráce
ArrayList myState -- Seznam pozic státu hráce
int steelUsers -- Velikost výroby hráce
ArrayList myTowns -- Seznam mest hráce
ArrayList myEnemies -- Seznam neprátel hráce
int labs -- Pocet laboratorí hráce
PlayerInfos playerInfos -- Informace o jednotlivých hrácích
```

Hlavní metody

```
override void StartTurn() -- Startovací procedura AI
void ChooseNewTechnologies() -- Výber nové technologie
void GetDangerOfMyNeighbours() -- Zjištení nebezpecnosti sousedu
```

```
void GetWholeStrategy() -- Diplomatické tahy hráce
int MyDistanceFromHim(int hisId) -- Vrací nejkratší vzdálenost dvou hrácu
void BuildBuilding() -- Vstupní metoda, která vytvárí tahy pro budovy
ArrayList BuildBateriesAndDefenceBunkers() -- Stavba baterií a bunkru
ArrayList FilterMapInformation() -- Filtruje aktuální informace
ArrayList GetBuildingTargets() -- Vrací cíle typu stavení
void GetMapInformation() -- Zjištuje aktuální informace
void MoveWithFigures() -- Vstupní metoda pohybu jednotek
ArrayList FindGlobalTarget(Unit u) -- Vyhledá globální cíle jednotek
ArrayList FindLocalTargets(Figure figure,int maxRadius) -- Hledání cíle
ArrayList GetTarget(Unit figure, Position position) -- seznam cílu na pozici
```

Boss

Popis

Fasáda. Muže být aktivní maximálně jedna (jeden potomek). Děďí z ní specializovanejší Bossové

Data

```
State state -- stav hry
```

BossGame

Popis

Fasáda pro hru

Data

```
MoveFactory localPlayerFactory -- Továrna na tahy
```

Hlavní metody

```
virtual void DispatchMsgs() -- Vybere zprávy z MessageReceiveru a provede je
virtual void ForceEndTurn() -- Vynutí ukoncení tahu aktuálního hráce
```

BossLocal

Popis

Fasáda pro hru, lokální hráč bez síťového rozhraní s počítačovými hráči

Data

```
Ai[] ais -- Pole pocítacových hrácu
Thread aiThread -- Vlákno používané pro pocítacové tahy
```

Hlavní metody

```
void TryRunAi() -- Je-li aktuální hrác pocítacový hrác, spustí AI
void RestartAiThread() -- Vytvorí nebo obnoví vlákno aiThread
void KillAiThread() -- zruší vlákno aiThread
void AiLoop() -- Smycka bežící na vlákne aiThread blokována semaforem
void InicializeAis() -- Inicializace pocítacových hrácu
virtual void ChangePlayerType(int index, PLAYER_TYPE newType)
```

BossClient

Popis

Fasáda pro síťovou hru, hráč-klient

Data

```
RemoteClient remote
```

Hlavní metody

```
ServiceProvider GetProvider(int index) -- Informace o poskytovateli pripojení
int Connect(DirectPlayAdr hostAdr, DirectPlayAdr providerAdr)
int SendToServer(Msg msg) -- Pošle zprávu na server
void AskForSynchronization() -- Požádá server o synchronizaci stavu hry
void Disconnect() -- Odpojí lokálního hráce od serveru
```

BossServer

Popis

Fasáda pro síťovou hru, hráč-server s počítačovými hráči

Data

```
RemoteServer remote
ArrayList pendingPlayerNetId -- Id hracu, kteri cekaji na pripojeni
```

Hlavní metody

```
void RemovePendingPlayer(int netId) -- Odstraní hráce z cekajících hrácu
void SendPreconnectionInfo(int netPlayerId) -- Odešle hráci informace o hre
void Host(DirectPlayAdr adr) -- Založení serveru
void Kick(int playerId) -- Vynutí odpojení zadaného hráce
void SendTo(Msg msg, params int[] playerToNetIds) -- Rozešle zprávy klientum
void SendTo(Msg msg, int playerToNetId) -- Odešle zprávu klientovi
```

Remote

Popis

Bázová trída pro komunikaci s DirectPlay. Adaptér. Třídy RemoteServer a RemoteClient z ní dědí a podle svých potřeb metody překrývají

Data

```
DirectX8 directX
DirectPlay8Event dxEventHandler
BinaryFormatter binFormatter -- Serializující
```

Hlavní metody

```
virtual void FillAppDesc() -- Požadavek o aktualizaci ApplicationDescription
virtual void AppDescChanged() -- Zmena ApplicationDescription
```

MsgReceiver

Popis

Uchovává frontu zpráv a událostí spojených s připojováním (sítových) hráču. Singletone.

Data

```
MsgQueue msgQueue -- Fronta zpráv
```

MoveFactory

Popis

Abstraktní továrna na vytváření tahu.

Data

```
int forPlayerId Identifikacní císlo hráce, kterému je tah urcen
```

Hlavní metody

```
virtual Move CreateTurn(Target target) -- Vytvorí z cíle tah
virtual Move CreateSOMEMOVE(\dots) - Sada metod, vracejících Move
```

Administration

Popis

Udžuje informace o administrativách (státních správách).

Data

```
string name -- Identifikacní retezec administrace
double quantumOfProduction -- Velikost produkce v administraci
double quantumOfInvention -- Velikost výzkumu v administraci
double quantumOfEarning -- Velikost príjmu v administraci
int changeTime -- Doba zmeny na danou administraci
```

Hlavní metody

```
void FillFromMemento(Mem.DataADMINISTRATION data) -- Nactení z mementa
void FillMemento(ref Mem.DataADMINISTRATION data) -- Uložení do mementa
```

4.2 Resources

PatchManager

Popis

Stará se o režii spojenou s načítáním a ukládáním dat. Udržuje seznam načtených souborů (patch).

Data

```
StringCollection loaded -- Kolekce souboru nactených na požádání
StringCollection roots -- Kolekce souboru, které byly nacteny na požádání
string main -- Jméno hlavního datového souboru. Vždy maximálne jeden
XmlPatchCollection toApplay -- Kolekce souboru, které se mají nacíst
```

Hlavní metody

```
string MakeLanguageSpecificFileName(string neutralFileName)
XmlPatch CreateXmlPatch(string fileName) -- Vytvorí XmlPatch ze souboru
void CreateToApplyList(string fileName, StringCollection ignore)
void ApplyXmlPatch(XmlPatch patch, RES_TYPE_FILTER filter)
XmlPatch WadToApplyList() -- Spojí XmlPatche ze seznamu toApply
void Load(string fileName, RES_TYPE_FILTER filter) -- Nacte XmlPatch
void Save(Stream stream, PatchType memento, RES_TYPE_FILTER filter)
```

Strings

Popis

Udržuje dvojice textových řetězců a klíčů (taky řetězce).

Data

```
static Strings mainStrings -- Udržované retezce
```

Hlavní metody

```
void AddString(string Id, string String) -- Pridá textový retezec s daným Id
static string GetStringSafe(string id) -- Vrátí textový retezec podle Id
static string GetStringSafeFormat(string format, params object[] args)
void FillFromMemento(Memento.ResourcesType resources) -- Vyplní podle mementa
void FillMemento(ref PatchType patchType) -- Naplní memento nactenými retezci
```

XmlPatch

Popis

Stará se o spojování zdrojů (v xml)

Data

```
PatchType memento
```

Hlavní metody

```
static XmlPatch operator + (XmlPatch a, XmlPatch b) -- Spojí dva XmlPatche
XmlElement FindDataElement(XmlElement classElement, string id)
```

4.3 Presentation

SpriteViewer

Popis

Uchovává objekty typu Sprite

Data

```
SpriteCollection sprites -- Kolekci Spritu
IStateRead stateRead -- Stav
```

Hlavní metody

```
void RecreateUnitSprites() -- Aktualizace Spritu podle stavu hry
UnitSprite FindSpriteForUnit(Unit u) -- Najde Sprite pro danou jednotku
void UnitMoved(Unit u) -- Vyhledá Sprite a posune jednotku
```

Sprite

Popis

Viditelný grafický prvek (obrázek) na mapě

Data

```
Point location -- Pozice sprite na mape v bodech
Size size -- Vrátí rozmery Sprite v bodech
Point goingTo -- Pozice, na kterou se jednotka pohybuje
int id -- Identifikace Spritu
int level -- Výšková úroven Sprite. cím vyšší - tím pozdeji je vykreslen
```

MapCursor

Popis

Základní třída kurzoru. Ostatní kurzory z ní dědí a překrývají její metody

Data

```
Position pos -- Pozice kurzoru
```

Hlavní metody

```
virtual void Draw(Surface dest, Point at, int SSize) -- Vykreslení kurzoru
```

ChartFace

Popis

Vzhled mapy a hráčů

Data

```
IStateRead stateRead -- stav
TerrainVariation[] vars -- Pole Picture pro každý typ terénu
ChartSprite[,] face -- Pole Spritu pro danou mapu
PlayerColors colors -- Pole barev používané k odlišování hrácu
```

Hlavní metody

```
void Inicialize(IStateRead stateRead) -- Inicializace vzhledu mapy a hrácu
void RefreshFromState() -- Aktualizace vzhledu mapy ze stavu
void FillFromMemento(Mem.PatchType memento) -- Nactení ChartFace z mementa
void FillMemento(ref Mem.PatchType memento) -- Uložení ChartFace do mementa
```

MapViewer

Popis

Zobrazuje mapu a jednotky

Data

```
int squareSize -- Aktuální rozmery políček
Point viewPort -- Pozice obdelníku, ve kterém se zobrazuje mapa
Size viewPortSize -- Rozmery obdelníku, ve kterém se zobrazuje mapa
Point slideTo -- Pozice na mape levého horního rohu viditelné oblasti
int renderSkipFrames -- Kolik snímku vykreslit, než se znovu vyrenderuje
int followUnit -- Sledovaná jednotka
Surface renderPortSurface
IStateRead stateRead
Engine engine
SpriteViewer spriteViewer
ChartFace face
Point focusCenter -- Stred ovládacího prvku, pri centrování na jednotku
```

Hlavní metody

```
void Inicialize(IStateRead iStateRead, SpriteViewer spriteViewer)
void CreateSurfaces(int newWidth, int newHeiht) -- Vytvorí Surfcasy
void Slide() -- Posune viewPort smerem k SlideViewPortTo
virtual void DrawChart() -- Zobrazí cást vykreslené mapy
virtual void DrawTerrainSquare(Surface dest, Point at, Point from)
void RenderChar() -- Vykreslí oblast mapy urcenou pomocí renderPort
virtual void RenderSprite(Sprite s, Rectangle clip) -- Vykreslí jeden Sprite
virtual void UpdateRenderPortPosition() -- aktualizuje renderPort
```

4.4 Engine

PersistSurface

Popis

Rozšírení DirectDraw. Surface. Povrch je trvalého charakteru, umí se znovu načíst ze souboru, z kterého byl vytvořen

Data

```
string fileName -- Jméno souboru, ze kterého byl povrch vytvoren
string id -- Jednoznacná identifikace
Color colorKey -- Klícovací barva
Surface surface
```

Hlavní metody

```
DataSURFACE FillMemento()
void FillFromMemento(Resource.Memento.DataSURFACE memento)
DataSURFACE FillMemento() -- Uložení PersistSurface do mementa
void SetColorKey() -- Nastaví klícovací barvu
void Create() -- Vytvorí povrch
void Restore() -- Obnovení povrchu
void Recreate() -- Znovu vytvorí povrch. Pri zmene grafického modu
```

Mode

Popis

Reprezentuje grafický mód, který je v danou chvíli nastavený. State.

Data

```
SurfaceDescription desc -- Popis Surface
Control mainWindow -- Hlavní okno
Device device -- Vykreslovací zarízení
Surface front
```

Hlavní metody

```
virtual void Restore() -- Obnovení puvodního rozlišení
virtual void Switch() -- Prepnout mód
virtual Surface RenderSurface -- Povrch, na který se má zrovna renderovat
```

Engine

Popis

Grafický engine. Základní operace s grafikou

Data

```
Device device -- Výstupní zarízení
Mode currentMode -- Aktuální grafický mód
PersistSurfaceCollection surfaces -- Kolekce Surfacu
PictureCollection pictures -- Kolekce Pictures
```

Hlavní metody

```
void SetMode(Mode mode) -- Nastaví daný grafický mód
void SetFullScreen(SurfaceDescription desc, Control mainWindow)
void ToggleFullScreen() -- Prepne aktuální mód
void ShutDown() -- Uvolní adaptér
void RecreatePersistSurfaces() -- Znovuvytvorí trvalé povrchy
void FillFromMemento(Resource.Memento.PatchType data) -- Nactení z mementa
void FillMemento(ref Resource.Memento.PatchType data) -- Uložení do mementa
```

Picture

Popis

Obrázek, abstraktní trída. Ostatní vykreslované prvky z ní dědí

Data

```
string id -- Identifikace obrázku
```

Hlavní metody

```
abstract void Draw(Surface dest, Point at, Rectangle clip)
```

4.5 GUI

UnitPanel

Popis

Zobrazuje informace o jednotce a umožnuje hráči pro ni vybírat jednotlivé tahy

Data

```
int selectedUnitId -- Id vybrané jednotky
BossGame boss
MoveParams registeredMoveParams -- Parametry tahu vybrané jednotky
SelectUnitParams selectUnitParams Parametry vybrané jednotky
FullMapViewer fullMapViewer
```

Hlavní metody

```
override void Inicialize()
void FillPanel(Unit u) -- Vyplní panel pro zadanou jednotku
void SetCommandsVisibility(bool visible) -- Nastaví viditelnost príkazu
CheckContainerFormVisibility() -- Urcuje viditelnost panelu
void FillInfoPanel(Unit u) -- Vyplní info panel pro zadanou jednotku
```

NotesPanel

Popis

Panel zobrazující poznámky z probíhající hry (výsledky tahu, atd . . .)

Data

```
NOTE_CATEGORY Filter -- Filtr poznámek
NoteCollection Notes -- Kolekce Notu zobrazených v panelu
```

Hlavní metody

```
void AddToListView(Note n) -- Pridá poznámku do panelu
void ChangeDetail(NoteDetail newDetail) -- Prepne zobrazený detail
```

Game

Popis

Hlavní ovládací prvek, stará se o základní panely a zobrazuje mapu. Mediator.

Data

```
SpriteViewer spriteViewer
MiniMapMenu miniMapMenu
UnitPanel unitPanel
NotesPanel notes
BossGame boss
```

Hlavní metody

```
void CreatePanels() -- Vytvorí panely MiniMapMenu, UnitPanel a NotesPanel
void SetPanelsVisibility(bool visible) -- Schová nebo zobrazí panely
void SynchroStartedHandler() -- zobrazí dialog synchronizace klienta
```

GameMenu

Popis

Základní třída pro všechny ovládací prvky hry. Všechny ostatní z ní dědí. Obsahuje pouze tlačítka Ok a Zpět a nastavuje chování panelu

MoveParams

Popis

Základní třída pro všechny parametry tahu. Ostatní z ní dědí. Obsahuje informace o uživatelem vybrané akci určité jednotky. Při volání Do() je tah poslán do MsgReceiveru

Data

```
IUnitPanelContext context -- UnitPanel ze kterého cte informace
```

Hlavní metody

```
virtual bool VisibleOnUnit(Unit u) -- Možnost tahu pro danou jednotku
virtual bool NeedCheckUnit  -- Zmena parametru pri zmene jednotky
virtual bool NeedCheckSquare -- Zmena parametru pri zmene políčka
virtual RESULTS CheckUnit(Unit u) -- Možnost provedení tahu
virtual RESULTS CheckSquare(Position at) -- Možnost provedení tahu
virtual RESULTS Do(Position at, Unit target) -- Provede tah.
virtual MapCursor SOMECursor -- Sada metod vracejících správný kurzor
```

MoveParams

Popis

Hlavní ovládací prvek hry. Dynamicky se mění podle změn situace. Nachází se v ní smyčka hry

Data

```
IUnitPanelContext context -- UnitPanel ze kterého cte informace
```

Hlavní metody

```
void InicializeEngine() -- Inicializuje grafický engine
void LoadMainPatch() -- Nactení hlavního Patche
void GameLoop() -- Smycka hry
void AutoStartMainMenu() -- Zapnutí hlavního menu
void ActivateControl(LocalizableControl control) -- Zmena focusu
```

5 Příloha A - Návod hry

Hra Agresoři je tahová strategie odehrávající se v době druhé světové války. Hráči se zde ocitají v roli vůdců jednoho ze států tehdejší Evropy a snaží se docílit co největšího rozšíření svého státu.

Úvod

Po první světové válce, která zcela a nevratně změnila tvář Evropy nastala doba opětovného rozmachu. Lidé postupně zapomínali na hrůzy Velké války a očekávali lepší zítřky. Ohromný krach na burze v roce 1929 však nepřinesl nic než hladomor, bídu a strach. V mnoha státech se ujali moci diktátoři, kteří slíbili lidu jídlo a práci. Španělsko, Portugalsko, Itálie, S.S.S.R., Německo, Jugoslávie a jiné státy vedeni nacionálními diktátory pohlížejí za hranice a dychtí po dalším území. A v této chvíli - v únoru 1933 se ocitáte Vy - hráči této hry, aby jste jako diktátoři vedly své země za mocí. Lid je jen nástrojem války, a zbraně prostředkem, který ji vyhrají. Začíná nová epocha Evropy, začíná druhá kolonizace !

Cíl hry

Cílem hry je dovést svůj stát ke konečnému vítězství - obsadit celou mapu hry.

Jak začít hrát

V prvé řadě si musíte zvolit stát, za který budete hrát. Je dobré si vybrat pokud možno co největší nebo nejsilnější stát, se kterým máte větší pravděpodobnost výhry, ale pokud jste již hru za větší stát mnohokrát zkusili, určitě nebude od věci si jednou zahrát za některý z menších států a hru určitě uvidíte i z jiného úhlu. Po zvolení státu již dostáváte možnost spravovat celý stát podle svého uvážení. Hru začínáte hrát jako monarcha - tj. vaší státní správou je monarchie a máte vytvořen základní průmysl - tj. několik oceláren a továren.

Mapa

Každé políčko mapy má předem určené některé vlastnosti, které vysoce ovlivňují průběh celé hry. Typem terénu se rozumí, jaký terén se na políčku nachází. Může to být jeden ze sedmi terénů a to - nížiny, lesy, hory, pouště, moře, řeky, naleziště rudy a železné rudy. Terén určuje které figurky se zde mohou pohybovat a jak rychle.
Vlastníkem terénu se rozumí hráč, který toto políčko vlastní - tj. hráč, jehož pozemní figurka se na tomto poli nacházela jako poslední.
Přechody mezi políčky, které vlastní různí hráči se nazývají hranice. Hranice mohou figurky hráče překročit pouze pokud je hráč s vlastníkem pole ve válečném stavu.
Aby bylo možné vyrábět figurky, existují zde suroviny ze kterých lze vyrobit ocel a z té následně tyto figurky vytvořit. Jsou to železná ruda - s pomocí ocelárny přeměňuje železnou rudu v ocel, ze které je možné vyrábět a naleziště železné rudy, která se po výstavbě dolu na železo dá použít stejně jako železná ruda - tj. vystavit zde ocelárnu a pak vyrábět.

Pohyb figurek po mapě a jeho pravidla

Každý typ figurky má předem dáno, jak daleko se dokáže během jednoho kola přesunout a po jakém terénu se nejlépe pohybuje.
Většina pozemních figurek se nejlépe pohybuje po nížinách (vyjímky tvoří např.

pouštní nebo horské figurky), leteckým nezáleží na typu terénu a námořní se dokáží pohybovat pouze na moři. Na každém políčku mapy smí v určité nadmořské výšce stát pouze figurky jednoho hráče. Figurky se mohou pohybovat do všech 8 směrů. Útočí vždy pouze na figurky ve své bezprostřední blízkosti (tj. na figurky, které stojí vedle ní) nebo na figurky, na které se dostanou svým dostřelem (např. děla). Některé figurky (převážně letecké) mají povinnost se do určité doby vždy vrátit na svou základnu a doplnit benzín. Pokud se do té doby nevrátí, jsou zničeny.

Budovy a jejich stavba

Ve hře můžete stavět několik typů budov, které buď zvyšují objem výroby státu, jeho vynalézání nebo jeho příjem. Typy budov a jejich výhody jsou:

Laboratoř dá se postavit kdekoliv ve státě hráče a zvyšuje jeho rychlost vynalézání.

Ocelárna dá se postavit pouze na místě výskytu železné rudy, z níž vytváří ocel, ze které je možné postavit figurky.

Továrna dá se postavit kdekoliv ve státě hráče a z ocele vytvořené v ocelárnách vyrábí pozemní figurky. Každá továrna je vždy postavena pro jeden druh figurek, a tyto figurky vytváří, pokud je povolen hráčem příjem ocele do této továrny. Druh figurek, které továrna staví se dá změnit, ale přestavba továrny trvá určitou dobu a stojí další peníze. V továrnách se mohou pozemní figurky za určité peníze opravovat do původního stavu, případně i vylepšovat na novější model.

Obranný bunkr dá se postavit kdekoliv ve státě hráče a zvyšuje obranu všem pozemním „nemechanickým “figurkám stojící na tomto místě, kde je bunkr vystaven o polovinu obranu.

Důl na železo dá se postavit pouze na nalezišti železné rudy, které přetváří na železnou rudu. Z této železné rudy je možné vytvářet ocel v ocelárnách a dále přetvářet na figurky.

Baterie dá se postavit kdekoliv ve státě hráče a umožňuje ničení figurek a bombardování nepřítele v dosahu několika polí. V zásadě ji lze považovat za útočnou jednotku, pro kterou ovšem platí pravidla budov.

Loděnice dá se postavit kdekoliv na pobřeží a přetváří ocel z oceláren na námořní figurky. Platí pro ni stejné pravidla jako pro továrnu.

Letiště dá se postavit kdekoliv ve státě hráče. Umožnuje přetvářet ocel na letecké figurky a doplňuje jim benzín. Platí pro ně stejné pravidla jako pro továrnu.

Každá z těchto budov může být nepřítelem vybombardována. Ve chvíli, kdy je budova vybombardována, přestává její účinek fungovat a budova ztrácí jakýkoliv význam. Vybombardovanou budovu nelze jakkoliv měnit nebo jinak vybavovat. Vybombardované budovy se dají opravit vlastníkem budovy, ale oprava stojí určité peníze.
Každou z těchto budov budov lze vylepšit tzv. ochranou proti vybombardování. Vyráběcí budovy (továrnu, letiště a loděnice) lze také přizpůsobit na přetváření novějších modelů. Pokud přizpůsobena není, nelze v ní jednotky vylepšovat.

Města a jejich pravidla

Od začátku hry existují na mapě města. Jejich poloha se nedá měnit (stejně jako u budov),ale nelze je ani stavět. Každé město přidává hráči, který ho vlastní určité peníze do jeho kasy. Počet peněz závisí na vybavení města. Město je možné vybavit těmito vlastnostmi :

Ochrana proti vybombardování použití je stejné jako ochrana proti vybombardování u budov.

Hradby zvyšují obranu pozemních figurek v tomto městě o polovinu.

Banka znásobuje vydělávání města.

Každé město lze stejně jako budovu vybombardovat. Ve chvíli, kdy je město vybombardováno, ztrácí nejen jakékoliv výhody vymožeností, ale nevydělává ani žádné peníze.

Výroba figurek

Figurky se dají vyrobit v továrnách, letišťích nebo v loděnicích (dále jen výrobny). Výroba probíhá tak, že se výrobně vybere možnost „Vyrábět“. Od množství oceli, kterou hráčův stát produkuje se odečítá množství oceli potřebné na výrobu této figurky. Tento postup může hráč opakovat do té doby, než mu jeho množství oceli již nestačí v žádné výrobně na výrobu figurky.
Tyto vyrábějící se figurky se po dokončení jejich výstavby objeví ikolo této továrny a hráč již může s touto figurkou nakládat jak chce.

Výzkum nových technologií

Ve hře může hráč díky stavbě laboratoří využívat novější technologie, nové figurky a budovy.
Každá vystavěná laboratoř zvyšuje možnosti vynalézt další technologii.
Nové technologie se staví podle předem daného vynálezeckého stromu, kde nelze zkoumat technologie u které nejsou probádáni všichni technologičtí předci této technologie.

Státní správa

Každý hráč má možnost si zvolit podle svého uvážení státní správu. Státní správa má vliv na objem velikosti výroby, výzkumu i vydělávání. Jednotlivé vlivy na stát mohou být buď kladné (tj. objem se zvyšuje), stejné (objem zůstává neměnný) nebo záporné (objem se snižuje). Záleží na uvážení každého hráče, čemu dává přednost a co v aktuální době potřebuje nutněji.

	Výroba	Vydělávání	Vynalézání
Nacismus	+	0	-
Komunismus	+	-	0
Monarchie	-	0	+
Socialismus	0	0	0
Kapitalismus	-	+	0
Fašismus	0	+	-

Diplomacie

S ostatními hráči lze vést diplomatické debaty. V zásadě jsou dvojího druhu - ty na které není třeba odpovídat (vyhlášení války, zrušení aliance) a ty, na které je třeba druhým hráčem odpovědět (uzavření aliance, uzavření míru, zrušení míru s jiným hráčem). Jednotlivými možnostmi diplomacie jsou :

Zruš mír s požadavek zrušení míru s jiným hráčem

Vyhlašuji ti válku rozhodnutí vyhlásit válku druhému hráči

Uzavřeme mír nabídka uzavření míru

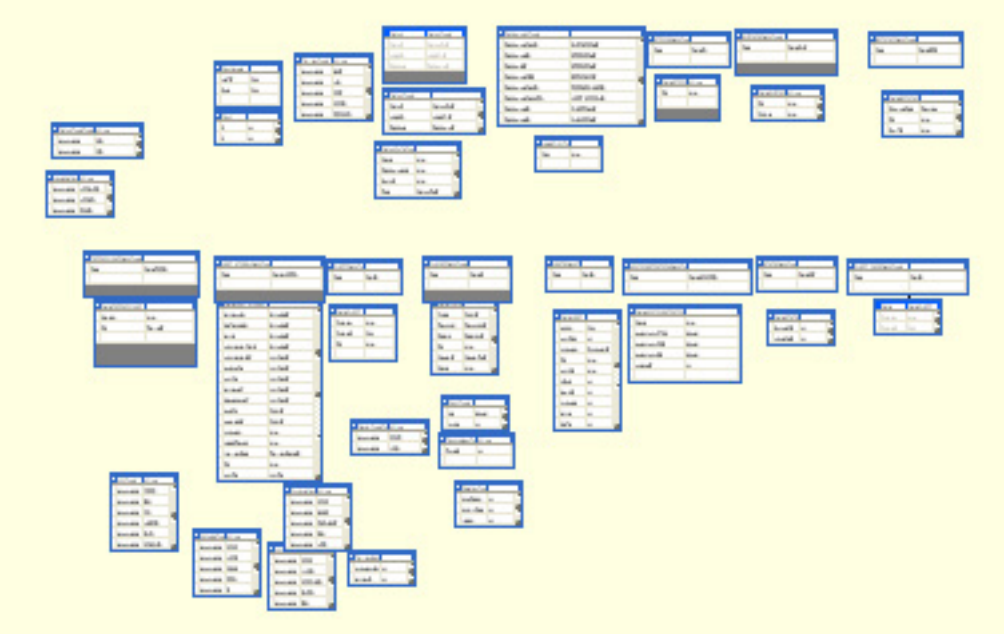
Uzavřeme alinaci nabídka uzavření obranné aliance

Ruším s tebou alianci rozhodnutí zrušení aliance

Přejeme hodně úspěchů ve hře !!

6 Příloha B - XML výstupy

Všechny vstupy i výstupy aplikace používají standartní formát XML. Tyto XML soubory mají definované schéma v souboru Resources.XSD jehož zmenšená grafická podoba je uvedena níže.



Obrázek 8: XSD schéma

Při načítání a ukládaní je využit vzor Mementa.
Při načítání je nejdříve naplněno memento v PatchManageru, tedy všechny třídy vygenerované příslušným schématem jsou naplněny pomocí XML serializace a uloženy do mementa. Potom, co je memento naplněno je vyvolána událost, že je memento naplněno a je třeba z něj příslušné data vybrat. Příslušné třídy, které mají nové data načíst (zjištěno filtrem, zda jsou data určena pro ně) si po vyvolání události z mementa příslušné data odeberou a aktualizují je.
Při ukládání se postupuje opačným způsobem. Nejdříve je vytvořeno prázdné memento a je vyvolána událost, že je třeba memento naplnit. Jednotlivé třídy událost odchytí a memento naplní. Poté, co je memento naplněno se uloží na příslušné místo pomocí PatchManageru.

Reference

[1] Bradley Bargen & Peter Donnelly: *Inside DirectX* Mircosoft Press, Redmond, Washington 1998

[2] Kosek, Jiří: *XML pro každého, podrobný průvodce* GRADA Publishing, Praha, 2000

[3] Jeffrey Richter: *Applied Microsoft .NET framework programming* Microsoft Press, Redmond, Washington 2002

[4] kolektiv autorů: *C# programujeme profesionálně* Wrox Press, překlad Praha 2003