**Universitatea tehnică din Cluj-Napoca Facultatea de Electronică,Telecomunicații și Tehnologia Informației**

**Specializarea:**

Master: Tehnologii Multimedia

**Structura jocurilor si tehnologiile folosite**

(Activitate de cercetare și proiectare Semestrul II)

**Profesor, Student masterand, Prof.Dr.Ing. Mircea Vaida Ing.Marius Bogdan Rad**

**2016**

Cuprins

**1.Introducere3**

2. Despre prototipuri3

3.Testarea4

**4.Limbaje de programare5**

5.Structura unui joc6

6.Concluzii8

**7.Referinte9**

**Programarea jocurilor si tehnologiile folosite**

1. **Introducere**

Dezvoltarea profesională a jocurilor începe de obicei cu un design care poate avea mai multe origini. Ocazional procesul de dezvoltare a jocului începe fără un plan concret, ci în urma unor serii de experimente.

Un game developer este un inginer software care creează în principal jocuri video sau software-uri aferente(cum ar fi instrumente de programare a jocurilor). Cel mai probabil fiecare joc are o strucutură de bază după care se începe fiecare joc. Această structură este creeata de un programator sau o echipă de programatori pentru a ușura munca pentru următoarele jocuri.

1. **Despre prototipuri**

Programatorii trebuie de obicei sa producă mai multe prototipuri pentru a ușura munca necesară pentru realizarea jocurilor. Acestea sunt create rapid și nimeni nu se așteaptă să funcționeze perfect, dar sunt folosite pentru a testa idei și a implementa task-uri noi.

Crearea prototipurilor este o activitate importantă care permite programatorilor și designerilor de jocuri să experimenteze diferiți algoritmi și scenarii pentru un joc.

Prototipurile sunt în esentă dovada de concept a jocului și acestea pot fi redate și evaluate. Acestea pot implica utilizarea unui limbaj de programare sau chiar un prototip pe hartie. Acest stadiu poate dura între o luna și șase luni, cu mici echipe care lucrează la mai multe prototipuri, urmând ca unul dintre ele să fie ales ca baza jocului ce va fi creat. Unele jocuri mai micuțe pot dura o saptămăna in funcție de dificultatea jocului. O data creeat un prototip, o structură de inceput a unui joc, timpul de realizare a unui joc scade foarte mult deoarece la bază exista clasele si obiectele cele mai folosite în crearea oricărui joc.[1]

1. **Testarea**

Procesul formal de testare a asigurării de calitate a fiecărui joc, efectuat de testerii de jocuri, începe încă din timpul producției. Testerii pot să înceapă să testeze primele variante ale jocului, în timp ce jocurile mai mici așteaptă până la lansare pentru a primi candidați. Sarcina programatorilor este de a corecta erorile descoperite de echipele de asigurare a calității jocului.[1]

Testarea este vitală pentru jocurile moderne, complexe, având în vedere că orice modificare simplă poate duce la consecințe catastrofale.

Printe sarcinile finale se numără și „lustruirea” jocului, deci fixarea erorilor de către programatori de la erori minore la erori catastrofice, care pot apărea în ultimile faze ale testării.

Producătorii de jocuri pot avea o perioadă de testare, dar acest lucru variază de la un producător la altul. Adesea, o variantă beta poate conține toate procese jocului, dar poate avea câteva greșeli sau conținut incomplet.[1]

Majoritatea jocurilor de azi necesită o perioadă cuprinsă între 1 și 3 ani pentru a fi terminate(cele complexe). Durata de producție depinde de numeroși factori, dar programarea este necesară de-a lungul tuturor fazelor, exceptând primele etape de design.

*Alpha* este etapa în care un joc poate fi jucat, el conținând toate caracteristicile majore. *Beta* este perioada în care jocul este complet, corectâdu-se doar erorile.

Corectarea greșelilor este o sarcină foarte importantă în procesul de dezvoltare. Unele limbaje de programare sunt mai predispuse anumitor defecte. Eliminarea acestora se face cu ajutorul unor IDEs(Integrated Development Environment sau Interactive Development Environment) precum Eclipse, Visual Studio, Kdevelop, NetBeans și Code::Blocks. Pentru realizarea jocurilor mai simple se poate folosi și Adobe Flash împreuna cu Action Script 3. În partea de Flash se adaugă evenimentele, grafică și se pot crea animații. In Action Script 3 se scrie propriu zis clasele și funcțiile utilizate pentru a crea funcționalitățile fiecarui obiect în parte.

Ca alte software-uri, programele de creare a jocurilor sunt generate de la codul sursă în programul actual(denumit executabil) de către un compilator. Codul sursă poate fi creat cu aproape orice editor de text, dar cei mai mulți programatori folosesc un mediu de dezvoltare complet integrat(IDE). Cele mai populare pentru Xbox si Windows sunt Microsoft Visual Studio și CodeWarrior.

Companiile producătoare de jocuri sunt adesea dornice să cheltuie mii de dolari pentru a se asigura că programatorii sunt bine echipați cu cele mai bune instrumente de lucru.

### Limbaje de programare

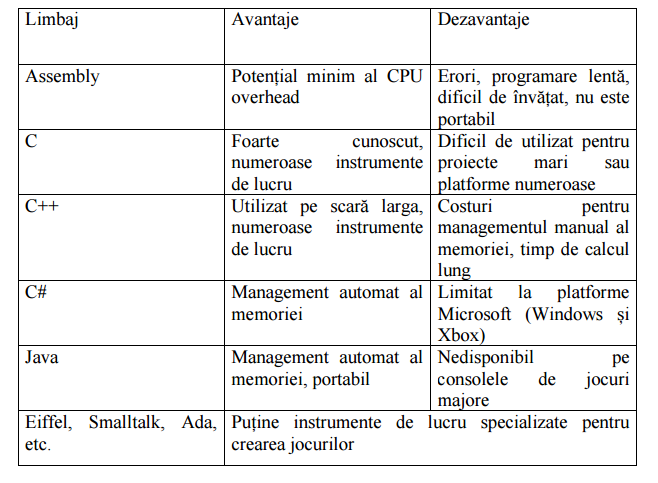
### 

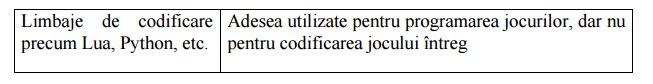
Limbajele de programare diferite cer stiuluri de programare diferite(numite paradigm de programare). [2]

Allen Downey, în cartea sa “How To Think Like A Computer Scientist”, spune: Detaliile arată diferit în limbajele de programare, dar câteva instrucțiuni de bază apar în aproape toate limbajele.

Astăzi, cel mai popular limbaj de programare este C++. Totuși, Java si C sunt de asemenea populare, dar nepotrivite pentru aceste proiecte.[2]

Limbajele de codificare de nivel înalt sunt din ce în ce mai folosite ca extensii integrate în jocul de bază scris într-un limbaj de programare de nivel mediu precum C++. Mulți producători au creat limbaje personalizate pentru jocurile lor, precum QuakeC(id Software) și UnrealScript(Epic Games). În următoarea figură vă prezint cele mai folosite limbajele si avantajele și dezavantajele fiecărui limbaj.[2]





#### Structura unui joc

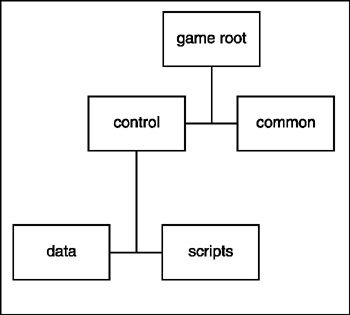
Componenta central a oricărui joc, din punct de vedere al programării, este ciclul de rulare a jocului. Acesta permite jocului să ruleze fără probleme, indiferent de datele de intrare ale unui utilizator sau lipsa acestora.[1]

Majoritatea programelor software tradiționale răspund datelor introduse de utilizator și nu fac nimic fără ele. De exemplu, un procesor de text formatează cuvintele pe măsură ce utilizatorul le testează. Dacă utilizatorul nu tastează nimic, procesorul de text nu va face nimic. Unele funcții pot lua mai mult timp pentru a fi finalizate, dar sun inițiate de un utilizator care îi dă programului o comandă.

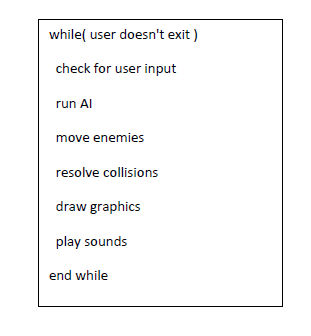
La crearea unui joc, putem utiliza destul de bine orice structura organizationala care ne este de folos. Jocul v-a cuprinde module de program script, imagini grafice, modele 3D, fisiere audio și diverse alte module de definire a datelor.

Singura limitare reală în modul în care structura folderelor unui joc este faptul că modulul principal root, trebuie să se afle în același folder ca și executabil, iar acest dosar va fi folderul root al jocului. Cel puțin ar trebuii sa facem pentru a organiza în mod rațional dosarele de joc, este de a avea un subarbore care conține cod comun, cod care ar fi in esență aceleași tipuri de joc și variații, și un alt subarbore care ar conține codul de control si resursele specifice care se referă la un joc special, tipul de joc, sau variație de joc.

Structura unui joaca poate arata ca si in urmatoarea imagine:



Jocurile, pe de altă parte, trebuie să continue să funcționeze independent de datele de intrare ale unui utilizator. Ciclul de rulare permite acest lucru. Un ciclu foarte simplificat, în pseudocod, poate arăta astfel:



Astfel, chiar dacă utilizatorul nu tastează nicio comandă, jocul va rula, alături de toate caracteristicile sale(realitaea virtuală, spațiul, inamicii).



Acest joc din figura de mai sus se numeste Assassin”s Creed II si a fost creat în C++, C# și Python.

1. **Concluzii**

In concluzie cele mai utilizate limbaje pentru crearea jocurilor sunt C# împreuna cu Unity pentru realizarea jocurilor pentru Android si IOS, dar pentru browser se folosește cel mai des Javascript și HTML5, dar și Flash și ActionScript3. Cel din urma limbaj, cum ar fii ActionScript3 incă se mai foloșeste dar mai rar deoarece in curând browser-ele nu o sa mai suporte swf-urile create de Adobe Flash Player.

**Referinte**

[1] http://www.gamedev.net/page/resources/\_/technical/game-programming/

[2] https://regashi.wordpress.com/2010/09/23/game-programming-structure-basics

[3] https://www.amazon.com/Game-Programming-Patterns-Robert-Nystrom/dp/0990582906?ie=UTF8&\*Version\*=1&\*entries\*=0

[4]https://www.reddit.com/r/gamedev/comments/i7eom/game\_programming\_architecture\_and\_patterns\_how\_to/