**Software prototyping**

Marin Andrei grupa 443A

Chiroiu Emilia grupa 442A

**Cuprins:**

**Marin Andrei:**

1. **Introducere**
2. **Dimensiuni de prototip** 
   1. Prototipul orizontal
   2. Prototipul vertical
3. **Tipuri de prototip**
   1. Prototip rapid
   2. Prototip evolutiv
   3. Prototip incremental
   4. Prototip extrem
4. **Prototip rapid (Rapid Prototyping)**

**Chiroiu Emilia:**

1. **Proiecte si metodologii de prototyping**
   1. Dezvoltarea sistemelor evolutive
   2. Dezvoltarea evolutiv rapida (ERD)
   3. Prototipul operational
   4. Metoda de dezvoltare a sistemelor dinamice (DSDM)
2. **Bibliografie**

**Marin Andrei:**

1. **Introducere**

Software prototypingul consta in creearea de prototipuri software, proces asemanator creeari prototipurilor din alte domenii.

Aceste prototipuri sunt versiuni incomplete ale unui program software care urmeaza sa fie dezvoltat, si au menirea de a oferi utilizatorului un mod de a interactiona cu aplicatia in fazele de inceput ale ei si de a primi feedback de la acesta, asa incat sansa ca la finalul procesului de dezvoltare software ca fie nevoie ca produsul final sa fie schimbat radical devine mica. De asemenea ofera inginerului software informatii referitoare la durata proiectului si daca deadline-urile pot fi atinse.

**2. Dimensiuni de prototipuri**

**2.1 Prototipul orizontal:**

* ofera o privire de ansamblu asupra procesului de dezvoltare si este concentrat pe interactiunea cu utilizatorul.
* acest tip de prototip este intalnit in special la echipe mari, unde designeri cu diferite abilitati au grija de layere diferite din arhitectura softwareului.

**2.2 Prototipul vertical:**

- asigura designerul ca poate implementa intreg sistemul dorit de utilizator, acest tip de prototip vine dupa ce s-a facut prototipul orizontal.

- este un tip de prototip de precizie inalta, si este un proces foarte tehnic.

**3. Tipuri de prototipuri**

**3.1 Prototip rapid**

Se refera la creearea unui model initial folosit pentru a prezenta utilizatorului modul in care va arata si functiona in mare aplicatia in momentul in care aceasta va fi finalizata.

**3.2 Prototip evolutiv**

Scopul evolutionary prototypingului este de a construi un prototip robust intr-o maniera structurata, si in mod constant aceasta va fi schimbata, in scopul de a fi imbunatatita.

Aceasta tehnica permite echipei de dezvoltare sa adauge functionalitati, pe care probabil nu ar fi putut sa le faca intr-un sistem mai strict de development.

**3.3 Prototip incremental**

Produsul final este construit ca prototipuri diferite. La sfarsit toate prototipurile sunt imbinate. Folosind incremental prototyping se scade timpul de a interactiona intre utilizator si cel care dezvolta softwareul.

**3.4 Prototip extrem**

Este folosit in special in dezvoltarea de aplicatii web, acesta sparge etapa de development in 3 faze.

Faza 1: se construiesc in general paginile HTML

Faza 2: se simuleaza partea de servicii a aplicatiei. In acestea faza se construieste interfata, simulandu-se serviciile din spatele ei.

Faza 3: se implementeaza partea de servicii a aplicatiei.

**4. Prototip rapid (Rapid Prototyping)**

Avantajele rapid prototyping-ului este acela ca ca poate fi facut foarte rapid, si sa se poata primi feedback de la utilizator inainte ca aplicatia sa intre in stagii mai avansate ale productiei.

Alt avantaj al rapid prototypingului este de a construi interfete pe care utilizatorul le poate vedea, si astfel interactiona cu sistemul.

A face schimbari in faza de inceput a productiei este foarte eficient din punct de vedere al timpului si costurilor, deoarece sunt putine lucruri facute inainte care sa fie pierdute in urma schimbarilor.

Viteza cu care se face prototipul este esentiala, deoarece daca acestea vor fi schimbate, nu se merita investitia de timp si bani.

Rezultatul final, are sansa mai mare de a fi conform cu specificatiile utilizatorului.

Unul din dezvantajele rapid prototypingului apare in momentul in care developerii se concentreaza prea mult pe functionalitati specifice, si pierd din vedere esenta proiectului, asa incat anumite optimizari si imbunatatiri pot fi trecute cu vederea, de asemenea utilizatorul poate sa creada ca prototipul este o reprezentare fidela a produsului final, de asemenea ei se pot atasa de functionalitati care vor disparea din produs in urma feedbackului de la alti utilizatori, de asemenea si developerii se pot atasa de prototip, care poate duce la incercarea de a transforma un prototip in produsul final.

Inca cazul in care prototipul nu este gandit cum trebuie acesta poate consuma prea mult timp si resurse asa incat sa nu mai fie profitabil de a face un asemenea prototip.

**Chiroiu Emilia:**

**5. Proiecte si metodologii de prototyping**

**5.1 Dezvoltarea sistemelor evolutive**

Evolutionary Systems Development este o clasa a metodologiilor ce au ca scop implemetarea formala a prototipului evolutiv (Evolutionary Prototyping).

Principiul metodologiei de prototip, Systemcraft, consta in crearea unui sistem functional pornind de la cerintele initiale si construit pe baza lor, trecand printr-o serie de revizii. Accentul metodologiei cade pe relarea analizei traditionale pe tot parcursul dezvoltarii produsului.

**5.2 Dezvoltarea evolutiva rapida (ERD)**

Consortiul SPC (Software Productivity Consotium) implementeaza Dezvoltarea evolutiva rapida (Evolutionary Rapid Development ), un concept al integrarii sistemelor software bazat pe refolosirea componentelor, a sabloanelor arhitecturale si software. Schimbarea rapida a nevoilor utilizatorului de sisteme software produce o continua evolutie a capabilitatilor produselor. Astfel s-a creat nevoia de o abordare flexibila cu reactie rapida la diferitele dezvoltari tehnologice. Proiectele bazate pe ERD contin o analiza de explorare paralela si dezvoltare de caracteristici, infrastructuri ce au ca scop principal adaptarea la tehnologiile de varf.

Procesul ERD este bazat pe demonstratii de functionalitate, ce creaza un flux mare de informatii intre dezvoltator si beneficiar. Metoda timebox este implementata in acest scenariu. Timebox-urile sunt perioade de timp prestabilite in care diferite sarcini trebuie finalizate. Astfel, rigiditatea si detalierea planulului orar consolideaza viziunea sistemul in ansamblu si se pot seta limite clare pentru proiect.

Stabilirea arhitecturii este urmata de integrarea si testarea zilnica a produsului software. Aceste activitati permit evaluarea obiectiva a progresului si identificarea potentialelor probleme in timp util.

**5.3 Prototipul operational**

Operational Protyping este o metoda de integrare a prototipurilor throwaway si evolutionary in dezvoltarea de sistem convetionala.

Propusa de Alan Davis, aceasta abordare contine urmatorii pasi:

* Un prototip evolutiv este constuit si folosit ca referinta prin strategii de dezvoltare conventional, specificand si implementand numai cerintele bine definite.
* Copii ale referintei sunt trimise diferitelor site-uri de consumatori impreuna cu un expert in prototipuri.
* In cadrul fiecarui site, expertul analizeaza fiecare utilizator al sistemului.
* De fiecare data cand utilizatorul intalneste o problema, dezvolta o noua cerinta sau functie, expertul o pune in aplicare.
* Utilizatorul evalueaza noul sistem. Daca schimbarile nu sunt eficiente, expertul le indeparteaza.
* Daca schimbarile sunt acceptate de catre utilizator, expertul scrie cerinte pentru functia respectiva si le prezinta echipei de dezvoltare.
* Echipa de dezvoltare, avand toate cerintele diferitelor site-uri, produce un nou prototip evolutiv folosind metode conventionale.

Metodologia prototipului operational este recomandata in sisteme complexe cu putine cerinte cunoscute inainte de etapa de dezvoltare.

**5.4 Metoda de dezvoltare a sistemelor dinamice**

Dynamic systems development method (DSDM) este un standard bazat pe metoda agila (Agile Metod) si dezvoltarea rapida a aplicatiei (Rapid Application Development). Acesta este lansat in 1995 de catre consortiul cu acelasi nume ca raspuns la metodele existente de dezvoltare software costisitoare, rigide, nedemne de incredere.

DSDM sustine folosirea mai multor tehnici ce includ: seminarii de facilitare, modelare si dezvoltare iterativa, prioritizare MoSCoW si metoda timebox.

Cele 4 categorii de prototipuri recomandate de DSDM sunt:

* Prototip de business – folosit pentru a proiecta si demonstra automatizarea proceselor de business
* Prototip de utilizare – folosit pentru a define, rafina si demonstra utilizarea, accesibilitatea si aspectul interfetei utilizatorului.
* Prototip de performanta si capacitate – folosit pentru a defini, demonstra si prezice care va fi performanta sistemelor in cazul supraincarcarii, cat si evaluarea aspectelor nefunctionale ale sistemului
* Prototip de tehnica sau capabilitate – folosit pentru a dezvolta, demonstra si evalua o abordare de proiect sau un concept.

Ciclul de viata DSDM al unui prototip este:

* Identificarea prototipului
* Acceptarea unui plan
* Crearea prototipului
* Revizuirea prototipului

1. **Bibliografie**

1. <https://www.kth.se/social/upload/52ef5ee4f2765445a466a28a/mackay-lafon-prototypes-52-HCI.pdf>

2. <http://www.efunda.com/processes/rapid_prototyping/intro.cfm>

3. <http://www.tutorialspoint.com/sdlc/sdlc_software_prototyping.htm>

4. http://www.dsdm.org/

5. http://agilemethodology.org/

6. http://www.methodsandtools.com/archive/dsdmatern.php