Laborator 4 - Utilitare pentru managementul și dezvoltarea proiectelor software

Aplicații web pentru MPS - GitHub

Aplicațiile web pentru managementul proiectelor software integrează mai multe aplicații folosite pentru a asigura o bună comunicare între membrii echipei și monitorizare a evoluției proiectului: wiki, source-code browser, tichete (buguri și solicitări de implementare), calendar (roadmap) etc. Trac, Redmine, Google Code și GitHub reprezintă cele mai cunoscute platforme ale acestei categorii de aplicații.
GitHub oferă următoarele funcționalități:

- crearea de organizații: de pe Select button-ul cu numele utilizatorului se selectează opțiunea "Create Organization"
- crearea de echipe: din interfața pentru organizație se accesează meniul "Teams" și se crează o echipă folosind butonul "New Team"
- crearea de proiecte publice sau private: din meniul din dreapta "Your repositories" se accesează butonul "New Repository" si se selectează tipul acestuia
- crearea unei pagini pe wiki: de pe pagina unui proiect se selectează din meniu "Wiki" și se apasă butonul "New Page"
- crearea de milestone-uri: de pe pagina unui proiect se selectează din meniu "Issues", se comută pe tab-ul "Milestones" și se apasă butonul "Create a new milestone"
- vizualizarea repository-ului și a commit-urilor
- realizarea de fork-uri: se apasă butonul "Fork"
- realizarea și acceptarea de pull request-uri
- crearea de branch-uri
- operația de "merge" între 2 branch-uri
- realizarea de code review
- bugtracking
- crearea și vizualizarea rapoartelor (issues)
- realizarea unor grafice despre evoluția proiectelor
- vizualizarea activităților din cadrul proiectului (creare de rapoarte (issues), știri, upload de document, commit-uri în repository, editări de wiki etc.)
- posibilitatea de a urmări evoluția altor proiecte: se apasă butonul "Watch" de pe pagina proiectului respectiv

Pentru a vă familiariza cu funcționalitățile sale de bază, accesați pagina de ajutor a platformei GitHub.

În caz de hazarde naturale, instrucțiunile sunt disponibile aici.

Modelare - Netbeans, Eclipse

Netbeans și Eclipse reprezintă un set de IDE-uri cu funcționalități complexe ce îi ajută, în special, pe dezvoltatorii de Java. Una din funcționalitățile importante ale IDE-urilor o reprezintă posibilitea creării de diagrame UML. În urma creării acestor diagrame se poate genera cod automat, reducând astfel efortul de dezvoltare a aplicației. Pentru C++/C# cel mai adesea se folosește Visual Studio.

Diagrame - MS Visio, Dia

În activitatea de proiectare se realizează numeroase grafice pentru a descrie componentele și modulele aplicației, legăturile între acestea și cazuri de utilizare.

Una dintre cele mai cunoscute aplicații pentru realizarea de diagrame este Microsoft Visio. Alternativa lightweight a acestuia este Dia. Alternativa din familia OpenOffice este OpenOffice Draw. Toate aceste aplicații conțin un set de imagini/module predefinite (utilizatori, sisteme de calcul, elemente de rețelistică, forme predefinite de imagini de componente). Recomandăm folosirea acestor aplicații pentru realizarea diagramelor și graficelor în realizarea documentației pentru toate proiectele unde este necesar. Aplicațiile de mai sus oferă o interfață grafică user-friendly și permit exportarea într-un număr vast de formate.
Pentru modelarea arhitecturii (în special pentru SDD) amintim StarUML (cu suport pentruMacOSX, Linux, Windows) și DrawIO, soluție online care poate exporta rezultatul atât ca imagine cât și ca pdf. Schema este salvată sub forma unui XML, ceea ce face posibilă salvarea și încărcarea cu ușurință a fișierelor de pe device-ul local direct în aplicația online si invers, pentru editare ulterioară. Alte aplicatii: Visual Paradigm sau Astah

Limbaje de programare, biblioteci, framework-uri, coding style

În cadrul procesului de proiectare și dezvoltare a unui proiect software, trebuie să se aleagă limbajul/limbajele de programare și bibliotecile folosite, framework-urile utilizate și alte aplicații care vor înlesni procesul de dezvoltare, testare și mentenanță.

Una din ideile importante în alegerea unui limbaj sau biblioteci este că nu există un super-limbaj sau un super-framework care să rezolve toate problemele, indiferent de natura acestora. Domeniul de software engineering s-a dezvoltat tocmai pentru a găsi cele mai bune opțiuni și proceduri pentru o situație dată.

Alegerea unui limbaj, a unui framework sau a unor utilitare pentru dezvoltare/testare se realizează, în general, ținând cont de două aspecte:

- cât de potrivit este acel limbaj sau framework pentru proiectul dat;
- cât de acomodați sunt viitorii dezvoltatori cu acel limbaj/framework.

Folosirea unui limbaj/framework adecvat dar pe care utilizatorii nu-l cunosc și pentru care partea de training durează mult, poate produce frustrări în rândul viitorilor dezvoltatori și poate genera ineficiență și, în consecință, un produs necompetitiv.

Ca recomandare generală, limbajele de programare se pretează la anumite situații:

- aplicații low-level, care necesită eficiență, se vor realiza, de obicei în C
- aplicații care au nevoie de dezvoltare rapidă vor fi realizate, de obicei, într-un limbaj cu facilități object-oriented de forma Java sau Python
- aplicații middleware vor folosi biblioteci și API-uri specializate (de forma OpenMP, MPI, RMI, Corba) și limbaje specializate
- aplicațiile web vor folosi framework-uri și limbaje specializate (Ruby on Rails, PHP, JSP/Servlets)

Un alt aspect important în proiectarea și dezvoltarea unei aplicații îl constituie formatul fișierelor de configurare. Acestea pot fi:

- fișiere binare - ocupă puțin spațiu, sunt relativ ușor de parsat, dar nu sunt portabile
- fișiere text format propriu - pot fi editate de utilizator, nu reprezinta un format standard
- fișiere .ini - reprezintă un fișier text în format standard
- fișiere .xml - format standard răspândit cu dezavantajul că ocupă mult spațiu și parsarea fișierului consumă memorie

Indiferent de limbajul, bibliotecile și framework-urile folosite, pentru facilitarea parcurgerii, înțelegerii și îmbunătățirii codului se recomandă folosirea unui set de reguli, încadrate de obicei, într-un document (de obicei fișier text) de tip "Coding Style" sau "Coding Standards". Exemple de astfel de documente sunt:

- Linux Coding Style
- GNU Coding Standards
- Kernel Normal Form
- PEP 8 -- Style Guide for Python Code
- pear Coding Standards
- Code Conventions for the Java Programming Language

În general, convențiile de codare diferă dar urmăresc aspecte comune:

- indentarea codului
- cod lizibil prin folosirea de spații și linii libere
- cod comentat și comentarii relevante
- nume relevante pentru variabile
- funcții cu dimensiune "umană" (nu kilometrice)
- consecvența stilului de codare (dacă se indentează codul cu TAB, se va indenta cu TAB peste tot; dacă se indentează cu 4 spații, se vor folosi 4 spații peste tot)

Documente de raportare

Folosirea unor documente de raportare periodice reprezintă o acțiune de bază în coordonarea și evaluarea activităților din cadrul unui proiect. Astfel de documente de raportare se vor referi la sarcinile planificate și realizate în perioada anterioară. Vor descrie, de asemenea, probleme apărute și soluții la acestea.

Documentele de raportare pot fi furnizate la nivel de săptămâna, lună, semestru sau an, funcție de care variază dimensiunea documentului și detaliile cuprinse. Exemple de șabloane de documente de raportare săptămânală se găsesc aici, aici și aici. Documentul de raportare săptămânală este realizat de project manager prin analiza activităților planificate/efecuate în cadrul săptămânii anterioare.
Documentele de raportare a activității au o structură tabelară în care sunt descrise principalele activități care s-au desfășurat pe parcursul perioadei raportate. Fiecare activitate este descrisă printr-un nume, data planificată pentru încheiere, responsabilul/responsabilii activității, starea curentă (în lucru, suspendată, dependentă de un rezultat, încheiată) și procentul realizat. Exemple de șabloane de documente de raportare a activității se găsesc aici și aici. Documentul de raportare a activității este completat de project manager pe baza informațiilor furnizate de membrii echipei și responsabilii fiecărei activități.
Evaluarea membrilor echipei este importantă în cadrul unui proiect. Evaluarea are rolul de a determina gradul de implicare și competență a membrilor echipei și de a lua decizii de actualizare a distribuției sarcinilor pentru eficientizarea realizării acestora. Un exemplu de formular de implicare a membrilor echipei este acesta. Evaluarea este realizată de project manager.