**Universitatea Naţională de Ştiinţă şi Tehnologie Politehnica Bucureşti**
**Facultatea de Electronică, Telecomunicaţii şi**
**Tehnologia Informaţiei**

# FIŞA DISCIPLINEI

## 1. Date despre program

| | |
|---|---|
| 1.1 Instituţia de învăţământ superior | **Universitatea Naţională de Ştiinţă şi Tehnologie Politehnica Bucureşti** |
| 1.2 Facultatea | **Electronică, Telecomunicaţii şi Tehnologia Informaţiei** |
| 1.3 Departamentul | **Dispozitive, Circuite şi Arhitecturi Electronice** |
| 1.4 Domeniul de studii | Inginerie Electronică, Telecomunicaţii şi Tehnologii Informaţionale |
| 1.5 Ciclul de studii | Masterat |
| 1.6 Specializarea | Calcul Avansat în Sisteme Embedded |

## 2. Date despre disciplină

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2.1 Denumirea disciplinei (ro)<br>(en) | | | | Dezvoltare software şi testare<br>Software Development Process and Testing | | | |
| 2.2 Titularul activităţilor de curs | | | | Conf. Dr. Radu HOBINCU | | | |
| 2.3 Titularul activităţilor de seminar / laborator | | | | Conf. Dr. Radu HOBINCU | | | |
| 2.4 Anul de studiu | 1 | 2.5 Semestrul | II | 2.6. Tipul de evaluare | V | 2.7 Regimul disciplinei | Ob |
| 2.8 Tipul disciplinei | | DA | 2.9 Codul disciplinei | UPB.04.M1.O.22-06 | | 2.10 Tipul de notare | Nota |

## 3. Timpul total estimat (ore pe semestru al activităţilor didactice)

| | | | | | |
|---|---|---|---|---|---|
| 3.1 Număr de ore pe săptămână | 4 | Din care: 3.2 curs | 2.00 | 3.3 seminar/laborator | 2 |
| 3.4 Total ore din planul de învăţământ | 56.00 | Din care: 3.5 curs | 28 | 3.6 seminar/laborator | 28 |
| Distribuţia fondului de timp: | | | | | ore |
| Studiul după manual, suport de curs, bibliografie şi notiţe<br>Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate<br>Pregătire seminarii/ laboratoare/proiecte, teme, referate, portofolii şi eseuri | | | | | 75 |
| Tutorat | | | | | 5 |
| Examinări | | | | | 0 |
| Alte activităţi (dacă există): | | | | | 14 |

| | |
|---|---|
| 3.7 Total ore studiu individual | 94.00 |
| 3.8 Total ore pe semestru | 150 |
| 3.9 Numărul de credite | 6 |

## 4. Precondiţii (acolo unde este cazul)

| | |
|---|---|
| 4.1 de curriculum | Cunoştinţe de programare în limbajele C/C++, Structuri de date şi algoritmi |
| 4.2 de rezultate ale învăţării | Abilităţi de utilizare a calculatorului, competenţe tehnice. |

## 5. Condiţii necesare pentru desfăşurarea optimă a activităţilor didactice (acolo unde este cazul)

| | |
|---|---|
| 5.1 Curs | Proiector şi acces la Internet. |
| 5.2 Seminar/ Laborator/Proiect | Laborator cu acces la Internet pentru studenţi şi proiector. |

**6. Obiectiv general** *(Se referă la intențiile profesorilor pentru studenți, la ceea ce studenții vor fi învățați în timpul cursului. Oferă o orientare cu privire la locul cursului în cadrul domeniului științific abordat, precum și la rolul pe care acesta îl are în cadrul specializării studiate. Vor fi descrise de o manieră generală tematicile abordate, justificarea includerii cursului în planul de învățământ al specializării studiate etc.)*

Obiectivul general este acela de a obișnui studentul cu procesul complet de dezvoltare software, care nu se rezumă doar la scrierea codului, cât și la toate procese conexe: compilare, testare, documentare și implementare. Astfel, studenții vor studia cum se utilizează eficient sistemele de control al versiunii, ce opțiuni au în privința instrumentelor eficiente de compilare, cum se testează aplicațiile folosind teste unitare, cum se generează documentație pentru o aplicație, și cum se folosesc sistemele de containere pentru integrare continuă, cros-compilare și implementare.

**7. Competențe** *(Capacitatea dovedită de a utiliza cunoștințe, aptitudini și abilități personale, sociale și/sau metodologice în situații de muncă sau de studiu și pentru dezvoltarea profesională și personală. Reflectă cerințele angajatorilor.)*

| | |
|---|---|
| **Specifice** | C3. Aplicarea cunoștințelor, conceptelor și metodelor de bază privitoare la arhitectura sistemelor de calcul, microprocesoare, microcontrolere, limbaje și tehnici de programare (3/4) |
| **Transversale (generale)** | - |

**8. Rezultatele învățării** *(Sunt enunțuri sintetice referitoare la ceea ce un student va fi capabil să facă sau să demonstreze la finalizarea unui curs. Rezultatele învățării reflectă realizările studentului și mai puțin intențiile profesorului. Rezultatele învățării informează studenții despre ceea ce se așteaptă de la ei din punct de vedere al performanței, pentru a obține notele și creditele dorite. Sunt definite în termeni concreți, folosind verbe similare exemplelor de mai jos și indică ceea ce se va urmări prin evaluare. Rezultatele învățării vor fi astfel redactate încât să fie evidențiată clar relația față de competențele definite la punctul 7.)*

| | |
|---|---|
| **Cunoștințe** | *Rezultatul asimilării de informații prin învățare. Cunoștințele reprezintă ansamblul de fapte, principii, teorii și practici legate de un anumit domeniu de muncă sau de studiu. Pot fi teoretice și/sau faptice.*<br>În urma acestui curs, studenții vor înțelege tot lanțul de instrumente necesare pentru aducerea unui software de la concepție la implementare și vor fi capabili să discearnă unde acesta nu funcționează și trebuie îmbunătățit. |
| **Aptitudini** | *Capacitatea de a aplica cunoștințe și de a utiliza know-how pentru a duce la îndeplinire sarcini și a rezolva probleme. Aptitudinile sunt descrise ca fiind cognitive (implicând utilizarea gândirii logice, intuitive și creative) sau practice (implicând dexteritate manuală și utilizarea de metode, materiale, unelte și instrumente).*<br>Concret, studenții vor ști să folosească sistemul de control al versiunii Git cât și platforma Gitlab în scenarii complexe, vor putea să scrie propriile teste unitare folosind biblioteca Google Test, vor putea crea medii complexe de compilare eficiente folosind GNU Make sau CMake, vor putea genera documentație pentru codul scris folosind Doxygen și vor putea integra toate aceste instrumente în sisteme de integrare continuă folosind sistemul de containere Docker. |

| | |
|---|---|
| **Responsabilitate și autonomie** | *Capacitatea cursantului de a aplica în mod autonom și responsabil cunoștințele și aptitudinile sale.* În urma acestui curs, studenții vor putea dezvolta în mod autonom, tot lanțul de dezvoltare software, plecând de la scrierea codului, până la implementare în producție. |

**9. Metode de predare** *(Se vor avea în vedere metode care să asigure predarea centrată pe student. Se va descrie modul în care se asigură participarea studenților la stabilirea propriului parcurs de învățare, cum se identifică eventualele rămâneri în urmă și ce măsuri remediale se adoptă în astfel de cazuri.)*

Predarea se va face într-un mod interactiv, prin exemple practice, prezentate în timp real, legate de proiecte software complexe.

**10. Conținuturi**

**CURS**

| Capitolul | Conținutul | Nr. ore |
|---|---|---|
| 1 | Sisteme de control al versiunii - Git | 8 |
| 2 | Utilizarea platformei Gitlab | 2 |
| 3 | Sisteme de compilare - GNU Make și CMake | 4 |
| 4 | Teste unitare - Google Test & Google Mock | 2 |
| 5 | Calculul ratei de acoperire a testelor - GCOVR | 2 |
| 6 | Generarea de documentație - Doxygen | 2 |
| 7 | Sisteme de containere - Docker | 2 |
| 8 | Cros-compilare folosind GCC/G++ | 2 |
| 9 | Integrare continuă (CI) în Gitlab | 4 |
| | **Total:** | 28 |

**Bibliografie:**

1. Documentația de referință Git - https://git-scm.com/docs
2. Scott Chacon și Ben Straub, Pro Git, ediția a II-a, editura Apress, 2024 - https://github.com/progit/progit2/releases/download/2.1.422/progit.pdf
3. Documentația Gitlab - https://docs.gitlab.com/
4. Documentația GNU Make - https://www.gnu.org/software/make/manual/make.pdf
5. Documentația CMake - https://cmake.org/cmake/help/latest/
6. Documentația Google Test - https://google.github.io/googletest/
7. gcovr, ghidul utilizatorului - https://gcovr.com/en/stable/guide.html
8. Manualul Doxygen - https://www.doxygen.nl/manual/index.html
9. Documentația Docker - https://docs.docker.com/get-started/overview/
10. Ruvinda Dhambarage, A master guide to Linux cross compiling - https://ruvi-d.medium.com/a-master-guide-to-linux-cross-compiling-b894bf909386
11. Documentație Gitlab CI - https://docs.gitlab.com/ee/ci/

**LABORATOR**

| Nr. crt. | Conținutul | Nr. ore |
|---|---|---|

| | | |
|---|---|---|
| 1 | Implementarea sistemului de compilare | 1 |
| 2 | Dezvoltare proiect software | 14 |
| 3 | Scriere de teste unitare | 7 |
| 4 | Adnotare și generare documentație | 2 |
| 5 | Crearea imaginii de Docker pentru compilare și cros-compilare | 2 |
| 6 | Configurarea sistemului de integrare continuă în Gitlab | 2 |
| | **Total:** | 28 |
| **Bibliografie:** | | |

**11. Evaluare**

| Tip activitate | 11.1 Criterii de evaluare | 11.2 Metode de evaluare | 11.3 Pondere din nota finală |
|---|---|---|---|
| 11.4 Curs | | | |
| 11.5 Seminar/laborator/proiect | Studentul a înțeles și a utilizat tehnicile în dezvoltarea proiectului. | Acumularea de puncte asociate sarcinilor de pe Gitlab, puncte acordate odată cu finalizarea cu succes a sarcinilor. | 100% |
| 11.6 Condiții de promovare | | | |
| Studentul va obține minim 10/20 de puncte asociate sarcinilor de pe Gitlab, din care:<br>• cel puțin un punct asociat cu o sarcină de tip ”dezvoltare”;<br>• cel puțin un punct asociat cu o sarcină de tip ”testare”;<br>• cel puțin un punct asociat cu o sarcină de tip ”proces”;<br>• cel puțin un punct asociat cu o sarcină de tip ”documentație”; | | | |

**12. Coroborarea conținutului disciplinei cu așteptările reprezentanților angajatorilor și asociațiilor profesionale reprezentative din domeniul aferent programului, precum și cu stadiul actual al cunoașterii în domeniul științific abordat și practicile în instituții de învățământ superior din Spațiul European al Învățământului Superior (SEÎS)**

În contextul ingineriei software, companiile dezvoltă aplicații din ce în ce mai complexe, acestea având nevoie de un mediu de dezvoltare din ce în ce mai bine automatizat. Lipsa unui astfel de mediu duce la probleme grave de performanță care măresc semnificativ timpul necesar implmentării complete a soluției. Astfel, disciplina este critică pentru orice inginer care lucrează în domeniu, pregătindu-l pe acesta cu un bagaj de cunoștințe care să îi permită să fie eficient pe tot parcursul procesului de dezvoltare.

**Universitatea Națională de Știință și Tehnologie Politehnica București**

**Facultatea de Electronică, Telecomunicații și**

**Tehnologia Informației**

| Data completării | Titular de curs | Titular(i) de aplicații |
|---|---|---|
| 21.02.2025 | Conf. Dr. Radu HOBINCU | Conf. Dr. Radu HOBINCU |

| Data avizării în departament | Director de departament |
|---|---|
| | Prof. Dr. Claudius DAN |

| Data aprobării în Consiliul Facultății | Decan |
|---|---|
| | Prof. Dr. Radu-Mihnea UDREA |