Ghid de elaborare a lucrării de licenţă la programul de studii Informatică Economică

Autori:

Lect. dr. Cristian Bologa

Prof. dr. Robert Andrei Buchmann

Conf. dr. Dan-Andrei Sitar-Tăut

Cuprins

Preambul

Prezentul ghid vine în sprijinul studenților care își pregătesc lucrarea de licență la programul de studii Informatică Economică.

Acesta trebuie tratat ca un **supliment** la ghidul și șablonul de redactare oferit de facultate, în consecință prezentul ghid nu oferă instrucțiuni de redactare.

Pentru o informare cât mai detaliată a viitorilor absolvenți, prezentul ghid vine să clarifice natura unei lucrări de licență la Informatică Economică sub diverse aspecte: categorii de lucrări, structura așteptată, natura informației de prezentat.

Urăm tuturor absolvenților succes și îi încurajăm să vadă în realizarea lucrării de licență o oportunitate de învățare într-un ritm propriu, cu sprijinul coordonatorului, mai degrabă decât un obstacol de depășit.

1. Competențe de coordonare

Orice cadru didactic din cadrul Departamentului de Informatică Economică poate coordona lucrări de licență cu condiția să dețină titlul de doctor.

În cazul în care dezvoltarea lucrării necesită competențe de specialitate din alte departamente, se poate alege un coordonator din alt departament în co-tutelă cu un coordonator din Departamentul Informatică Economică.

Tot în cotutelă, pot fi implicați și coordonatori de la companii cu care departamentul/facultatea noastră are parteneriate, sau cercetători afiliaţi facultăţii.

Domeniile de preocupare ale membrilor departamentului pot fi consultate în paginile/CV-urile acestora de pe site-ul FSEGA[1] sau profilurile științifice publice ale acestora (Google Scholar, Publons etc.)

Tematica lucrării de licenţă poate fi dată de către coordonatorul lucrării dar nu e obligatoriu; alternativ, studenţii pot veni la rândul lor cu propuneri pe care le adresează potenţialului coordonator. Temele ce se pot alege nu sunt limitate la cursurile predate de coordonatori, ci se vor stabili de comun acord cu coordonatorul.

[1] https://econ.ubbcluj.ro/departamente/cadre-didactice.php?c=4&s=1

2. Tipuri de lucrări de licență la Informatică Economică

Principii generale:

1. Lucrarea de licență este oglinda competenței unui absolvent și poate deveni o valoroasă piesă de portofoliu personal, de inclus în CV-ul unui student proaspăt absolvent de facultate. Dezvoltarea acesteia trebuie văzută în primul rând ca o oportunitate de învățare.

2. Autorul lucrării de licență trebuie să demonstreze abilități/competențe pentru care s-a acreditat programul de studii urmat[2]. Drept consecință, se vor evita lucrările de tip *prezentare de produs, conspect, sondaj de opinie, prezentare didactică, blog, testimoniale de la locul de muncă* – lucrarea trebuie subordonată unei metodologii consacrate de inginerie software sau de cercetare științifică.

3. Lucrarea de licență reprezintă o probă de evaluare finală complexă în vederea absolvirii unui program de studiu. Așadar, aceasta trebuie să depășească amploarea unui proiect de laborator axat pe o singură disciplină de specialitate. Ea trebuie să integreze atât aspecte teoretice/conceptuale (analiză de business, proiectare/modelare conceptuală), cât și deprinderi tehnice/practice (implementare software, procesare de date, exploatare de tehnologii).

4. Lucrarea de licență este un document cu autor individual, demonstrând abilități și competențe individuale. Nu se pot realiza lucrări de licență cu autori multipli, sau o aceeași lucrare depusă pentru mai mulți absolvenți, chiar dacă lucrarea face parte dintr-un proiect de largă anvergură în care au colaborat mai mulți studenți. Fiecare dintre aceștia trebuie să demonstreze propria abilitate de a redacta un document, de construi o bibliografie, de a elabora documentația tehnică, aspectele analitice, sintetice și practice ale unei astfel de lucrări – nu se pot delega părți din aceeași lucrare spre diferiți autori, chiar dacă lucrările acestora sunt subordonate aceluiași context de business sau unui același proiect de mai mare anvergură.

Pentru atingerea acestor deziderate se vor realiza următoarele tipuri de lucrări de licență:

****

Tip A. Proiectare și implementare de aplicații (Web, desktop, mobile etc.) **sau de componente software** (servicii și interfețe Web, module de automatizare pentru testare, instalare, RPA[3], procese ETL[4] pentru translatare de date/documente, module de inteligență artificială, agenți software, chatbots etc.)

Contribuția acestui tip de lucrare trebuie să acopere:

[2] vezi suplimentul de diplomă RNCIS:
http://intern.anc.edu.ro/virtualanc/crud/rncis/rncis_programe/brain/upload/14785980681523.pdf
[3] Robotic Process Automation
[4] Extract-Transform-Load

- *analiza unui context de business care ar putea beneficia de implementarea propusă* (modelul de business deservit, obiectivele și procesele de afaceri pentru care se realizează software-ul propus, analiză Before/After pentru a evidenția beneficiile așteptate de la implementare);
- *proiectarea aplicației/componentei implementate* folosind standarde de modelare consacrate (UML[5], BPMN[6], DFD[7], ERD[8]);
- *detalierea implementării* (discutarea unor mostre de cod sursă, descrierea bazelor/surselor de date accesate și a structurilor de date manipulate);
- *evaluarea calității implementării* din perspective relevante; exemple de perspective: strategia de asigurare a calității software (cu exemplificare de erori și abordări de depanare folosite în realizarea lucrării), măsurători (asupra utilizabilității, performanței, scalabilității), validare față de cerințe concrete (dacă aplicația are deja utilizatori reali).

Pentru a putea evidenția competențele de programare se va evita implementarea „prin configurare" în care studentul se limitează la a configura componente pe o platformă pre-programată (Wordpress, Joomla, ADOxx, Unity, UIPath, Mendix, Outsystems, sisteme ERP/CRM/CMS etc.) fără a intra în contact cu elemente de programare. Folosirea de astfel de platforme nu e interzisă, dar pentru obținerea unui calificativ bun se va ține cont de următoarele:

- rămân valabile cerințele de a acoperi analiza de context de business, proiectarea și evaluarea calității implementării;
- se vor evidenția abilitățile de programare pe una din următoarele căi:
    - fie se va folosi limbajul intern de programare oferit de aceste platforme (exemple: C# în Unity, AdoScript în ADOxx, .NET în UiPath, ABAP în SAP, PHP în Wordpress etc.) pentru a implementa funcționalități care nu sunt oferite implicit de acestea;
    - fie se va dezvolta o componentă externă care să interacționeze cu astfel de platforme printr-un mecanism de interoperabilitate (exemplu: un serviciu Web care să fie accesat de un robot UiPath, de o entitate Unity, de o diagramă ADOxx etc.).

Alternativ, aceste platforme se pot folosi și pentru a realiza lucrări din celelalte categorii descrise în continuare (cercetare științifică sau studiu de caz privind utilizarea/adopția a astfel de platforme).

****

Tip B. Cercetare științifică orientată empiric (formularea de ipoteze de cercetare și validarea/testarea lor prin analiză non-trivială de date colectate sau experimente)

Pentru acest tip de lucrare se va merge dincolo de statistică descriptivă și agregare/consolidare de date (sume, medii aritmetice, numărare etc., cu grafice Pie/Bar bazate pe sondaje de opinie) pentru a demonstra deprinderi de analiză statistică sau învățare automată subordonate științei datelor sau inteligenței artificiale. Contribuția acestui tip de lucrare trebuie să vizeze:

- formularea de întrebări științifice non-triviale (la care nu se cunoaște răspunsul a priori sau nu se obține prin statistică descriptivă sau agregări de date);

[5] Unified Modeling Language
[6] Business Process Model and Notation
[7] Data Flow Diagrams
[8] Entity-Relationship Diagrams

- aplicarea unei metodologii științifice consacrate sau îmbunătățite de autor;
- caracterizarea datelor folosite și a modului de colectare a lor (proiectarea de experimente dacă e cazul);
- sintetizarea literaturii în domeniu, fundamentată pe bibliografie științifică;
- interpretarea rezultatelor;
- discutarea de limitări și amenințări la adresa validității concluziilor.

****

Tip C. Studii de caz privind administrarea și configurarea de medii/platforme IT (exemple: medii Internet of Things, configurări de rețele, soluții de securitate, medii de virtualizare și deployment, precum și sisteme ERP, CMS, CRM).

Acest tip de lucrare nu se va limita la prezentarea de produs. Se va evita perspectiva de agent de vânzări/utilizator care promovează funcționalitățile unui produs sau prezintă documentația produsului.

Contribuția acestui tip de lucrare trebuie să vizeze:

- să fie un studiu de caz (posibil anonimizat), structurat conform metodei „studiului de caz" („case study");
- să ofere
    - fie o *analiză comparativă* - comparare fundamentată empiric (bazată pe date culese în cadrul studiului de caz) între soluții/produse comparabile, între arhitecturi diferite, între situații Before vs. After, între companii diferite care au adoptat o aceeași soluție cu rezultate diferite;
    - fie o *componentă dezvoltată* (automatizarea de instalări/configurări, scripting, implementare cu ajutorul unui SDK[9] sau serviciu Web oferit de soluția/platforma studiată);
    - sau o combinație între cele două.

****

D. Se acceptă și îmbinări între aceste tipuri de lucrări, ceea ce de obicei va crește complexitatea lucrării, de exemplu (dar nu limitat la):

- implementarea unei aplicații împreună cu un studiu de caz privind adopția aplicației într-un caz real;
- implementarea unei aplicații urmată de analiză asupra datelor obținute prin acea aplicație;
- crearea unui mediu de deployment urmat de implementarea unei aplicații în cadrul acelui mediu;
- implementarea aceleiași aplicații în două framework-uri asemănătoare în vederea unei analize comparative;
- etc.

[9] Software Development Kit

3. Cerințe de urmărit de către coordonator în vederea acceptării depunerii lucrării

1. Lucrarea să se încadreze într-una dintre categoriile menționate și să reflecte competențele pentru care este acreditat programul de studii Informatică Economică;
2. Existența componentei practice a lucrării (aplicația implementată, instrumentele de procesare a datelor, platformele analizate în cadrul studiului de caz) să fie demonstrabilă;
3. Lucrarea să conțină minim 30 de pagini de conținut original (conform formatului șablonului oficial la nivel de facultate), în urma scăderii din dimensiunea totală a lucrării a procentajului de similitudine detectat de aplicația facultății[10]; conținutul original să fie mult peste conținutul asimilat similitudinilor;
4. Lucrarea să aibă un titlu specific, nu doar cel generic al șablonului („Lucrare de licență”). Titlul lucrării să ofere indicii privind domeniul de aplicare și tehnologiile-cheie folosite:
    - exemple pozitive: „Aplicație de tip rețea socială implementată în Laravel cu bază de date MySQL”; „Sistem de recomandare de produse cu elemente de inteligență artificială implementate în Python”;
    - exemple negative: „Site făcut în PHP”, „Aplicație pentru Lavanda S.R.L.”, „Proiectarea și implementarea unui magazin virtual”;
5. Partea practică (implementarea demonstrată la susținere) și cea descriptivă (analiză/sinteză) a lucrării să fie în concordanță una cu cealaltă. Se vor evita situațiile în care:
    - diagramele sunt despre *o altă aplicație decât cea implementată* (exemple: diagrama use case indică funcționalități care nu există, diagrama de arhitectură arată componente inexistente, diagrama BD indică alte tabele decât cele prezente în aplicație etc.);
    - tehnologiile prezentate în lucrarea scrisă sunt *altele decât cele utilizate în implementare* (exemplu: se pretinde că s-a folosit un framework MVC și o bază de date Mongo, dar aplicația prezentată nu e MVC și/sau baza de date nu e Mongo);
    - în lucrare s-a folosit un anumit set de date/metodă/instrument, dar la susținere se folosește alt set de date/metodă/instrument.
6. Lucrarea să folosească un sistem de citare consistent și consacrat (Harvard, IEEE etc.) pentru respectarea drepturilor de autor, eventual cu ajutorul unui instrument de citare accesibil (de exemplu cel prezent în Word). Pentru evitarea unor situații de plagiat accidental, atragem atenția cu privire la următoarele scenarii ce trebuie evitate:
    - publicarea proiectului într-un depozit public (ex. Github) de unde poate fi preluat și prezentat de altcineva, uneori chiar înaintea autorului real (sau poate fi detectat ca o similitudine de aplicația de verificare);
    - folosirea de capitole care au fost utilizate la comun ca suport de învățare pentru diverse examene (deci riscă să fie preluate simultan în multiple lucrări);
    - folosirea de capitole preluate din ghiduri, modele din lucrări vechi sau de la alte centre universitare (adesea și acestea conduc la apariția de conținut comun în multiple lucrări dacă nu sunt filtrate și adaptate corespunzător).

[10] Turnitin la momentul redactării ghidului

4. Limbajul și stilul

1. Lucrarea se poate scrie în română sau engleză, dar nu o combinație între cele două. Excepție fac termenii netraductibili precum „mișcarea mouse-ului”, „baze de date în cloud” etc. Se vor evita exprimări de tipul „Sistem de Reporting al Taskurilor și Bugurilor pentru Developerii Remote”, având în vedere că toți termenii sunt traductibili;
2. Conținutul trebuie să fie alcătuit din fraze complete, cu subiect și predicat. Se va evita preluarea de text direct de pe prezentări Powerpoint. Powerpoint e menit să prezinte text schematic, lipsit de context narativ și conectori lexicali (deci necesită completări pentru a le transpune în contextul narativ al unor paragrafe Word);
3. Nu se va reproduce codul sursă integral în lucrare. În schimb, diverse mostre de cod-sursă se pot discuta în capitolul dedicat implementării (exemple: mostre de interogări, mostre de cereri HTTP, o mostră de controller PHP, un algoritm aparte care e contribuție proprie);
4. Descrierea aplicației implementate nu se va realiza exclusiv prin capturi de ecran – scopul lucrării nu e oferirea unui manual de utilizare, ci evidențierea modului în care s-a realizat implementarea. În consecință, eventualele capturi de ecran vor fi însoțite (în capitolul de implementare) de mostre de cod-sursă relevante pentru ce se vede în captura de ecran. Excepție fac eventualele schițe *mock-up*/*wireframe* (ecrane false realizate cu instrumente precum Figma), care nu sunt propriu-zis capturi de ecran, ci țin de proiectarea UX[11] și prin urmare își au locul în capitolul de proiectare;
5. Lucrarea trebuie să descrie contribuția efectivă a autorului, fără a avea stilul unui curs sau a unei prezentări de produs. De exemplu, lucrarea nu va dedica pagini întregi explicând ce e testarea, ce înseamnă Scrum, ce e un serviciu Web, ce e Web-ul, ce e o bază de date. Pentru a introduce astfel de noțiuni se vor folosi două secțiuni dedicate:
    - un **Glosar** de termeni de specialitate (pentru a sistematiza diverse „definiții” cu privire la tehnologii sau la domeniul de aplicare);
    - o secțiune **Tehnologii specifice**, în care se va dedica maxim un paragraf fiecărei tehnologii/framework/librării folosite, indicând și adresa URL oficială; această secțiune are rolul de a oferi o informare rapidă comisiei cu privire la tehnologiile folosite în lucrare, fără intenția replicării unui curs adresat comisiei despre acele tehnologii.

[11] User Experience

5. Structura

Sub aspect structural se va asigura o distincție clară între capitole:

- **Introducerea** va prezenta motivația lucrării (încadrată în paradigma tehnologică sau economică de care aparține lucrarea), va formula obiectivele, va menționa metodologia și tehnologiile folosite și va trece în revistă structura întregii lucrării. Șablonul facultății impune și includerea unui **Rezumat** separat, care să surprindă esența contribuției întregii lucrări într-o formă foarte concisă, indicând: natura lucrării (dezvoltare software, cercetare științifică, studiu de caz IT), obiectivul, tehnologia folosită și rezultatul obținut;
- Un capitol va descrie **Metodologia** de lucru (pașii în care s-a realizat lucrarea); aceasta va prezenta, în funcție de tipul lucrării, metodologia de inginerie software, cea de cercetare și analiză a datelor, respectiv de realizare a studiului de caz. Nu se va folosi acest capitol pentru a prezenta toate metodologiile cunoscute, dar se poate include o justificare a preferinței autorului pentru metodologia aleasă (față de altele). În acest capitol se va include și subsecțiunea **Tehnologii specifice** menționată anterior, ca o trecere în revistă a instrumentarului folosit (vezi și secțiunea 4) - tehnologiile folosite în implementare sau pentru prelucrarea datelor. Tot aici, lucrările de tip B vor oferi o privire de ansamblu (preferabil ca „workflow", diagrame de proces) asupra metodologiei de cercetare științifică (modul de colectare și prelucrare a datelor), iar lucrările de tip C asupra modului de desfășurare a studiului de caz, urmând ca acestea să fie detaliate în capitolele următoare;
- Lucrările de tip B și C vor include un capitol scurt de **Analiză a Literaturii**, care va invoca lucrări științifice ce au investigat probleme similare, sau studii de caz comparabile identificate în literatură[12]. Rolul acestui capitol este să rezume ce au realizat alți autori cu instrumente similare și/sau în scopuri similare, nu să prezinte didactic tehnologiile folosite și nici să ofere definiții de termeni;
- În capitolele de analiză/proiectare nu se vor include cod sursă sau capturi de ecran, ci diagrame și aspecte teoretice-descriptive; se va asigura separarea între un capitol de **Analiză de business** (concentrat pe a descrie contextul de business, motivația, cerințele și beneficiile contribuției) și unul de **Proiectare** (concentrat pe a descrie software și aspecte ale implementării). Pentru lucrările de tip B și C, capitolul de proiectare va începe cu proiectarea cercetării („research design"), respectiv proiectarea studiului de caz (o detaliere a privirii de ansamblu din capitolul Metodologie) înainte de a intra în detalii de proiectare a artefactelor IT (arhitectură, componente software, structuri de date) implicate;
- În capitolul de **Implementare** nu se vor include diagrame, ci fragmente de cod sursă însoțite de capturi de ecran (se va evita descrierea implementării exclusiv prin capturi de ecran, fără a oferi niciun detaliu privind modul de implementare). Pentru lucrările de tip B aici se va detalia demersul științific propriu-zis, de la colectarea la prelucrarea datelor, și implementarea eventualelor componente software ocazionate de acesta. Pentru lucrările de tip C se va detalia soluția IT analizată și modul în care a fost folosită în cadrul studiului de caz;
- Un capitol de **Evaluare** va evidenția atât calitățile măsurabile (bazate pe date) cât și limitările contribuției. În cazul lucrărilor de tip A se va evidenția și modul în care s-a asigurat testarea

[12] Pentru studiul literaturii se vor consulta și biblioteci digitale cu literatură științifică (AIS, ACM, IEEE, Springer, ScienceDirect etc.) sau platforme de indexare a acestora (Google Scholar)

software pe parcursul implementării (cu exemple de erori și modul de surprindere a lor). În cazul lucrărilor de tip B și C se vor interpreta și evalua rezultate (cantitative sau calitative), li se vor evidenția limitările și se vor formula recomandări derivate din rezultate;

- **Concluziile** vor reflecta asupra a ce s-a prezentat în lucrare și vor propune unele direcții de dezvoltare viitoare. Paragraful final al Concluziilor va descrie pe scurt ce consideră autorul că a învățat pe parcursul realizării lucrării (învățarea unei tehnologii, aprofundarea unui fenomen economic, a unei metodologii etc.) și care consideră că e contribuția cea mai importantă a lucrării;
- Un **Glosar** de termeni va oferi definiții succinte privind termeni tehnici sau terminologie de specialitate din domeniul de aplicare. Nu se vor include aici abrevieri, pentru acelea există o secțiune separată în șablonul facultății.

6. Formatarea

Principii generale:

- Formatarea lucrării trebuie să denote stăpânirea regulilor elementare de redactare a unui document Word;
- Se va folosi șablonul de formatare al facultății, acela conținând și ghidul de redactare/formatare.

Precizăm doar câteva cerințe punctuale:

1. Formatările să fie uniforme, fără variații de la un capitol la altul (toate paragrafele și spațierile, tabelele etc. să arate la fel structural și estetic);
2. Paginile să fie numerotate;
3. Orice tabel/figură să fie numerotată, cu o etichetă (*caption*) iar eticheta să aibă funcție informativă (se vor evita etichete ce conțin un nume de fișier, precum *add.html*, folosind în schimb etichete care informează cititorul cu privire la ce vede, de exemplu *Componenta client pentru adăugarea de produse*);
4. Orice tabel/figură să fie menționată în text prin trimitere la numărul acesteia (Fig. 2, 3 etc.) evitând referirea la poziție relativă („figura de mai jos”, „de pe pagina precedentă” etc.) căci nu există garanția că acele poziții vor rămâne nemodificate pe parcursul elaborării lucrării;
5. Figurile să se poată citi la imprimare alb-negru (font rezonabil, contrast adecvat).

7. Bibliografia

Principiu general: se vor include referințe bibliografice nu doar la citatele exacte, ci și la surse care au fost conspectate/rezumate.

Suplimentar indicațiilor oferite prin șablonul facultății, se vor include în bibliografie:

- adrese URL pentru orice instrument/librărie/framework folosit (Bootstrap, Laravel, JQuery etc.):
  - link de unde se poate obține;
  - link cu documentația oficială și eventualele tutoriale după care s-a deprins folosirea instrumentului;
  - acolo unde se poate identifica, link la specificațiile originale (exemple: la standardul UML, la specificația REST, JSON, HTTP etc.);
  - link la instrumente folosite în alte scopuri decât implementarea (instrumente pentru testare, mock-up design, diagrame etc.).
- linkurile de descărcare ale instrumentelor software se pot include și ca note de subsol, în secțiunea *Tehnologii specifice* menționată anterior (menită să rezume specificul tehnologic al lucrării);
- orice item din bibliografie, inclusiv adresele URL, trebuie să aibă titlu (să se indice ce se găsește la acel link) și, acolo unde e identificabil, autorul/organizația și anul publicării:
  - la adresele URL unde nu e identificabil autorul, obligatoriu se va preciza titlul;
  - dacă un link duce la o carte/articol se va trece în bibliografie cartea/articolul cu toate detaliile sale (editură, an etc.); se poate adăuga la final și linkul cu o mențiune de forma „disponibil/accesat la ...” dar nu se va folosi linkul ca substitut pentru detaliile publicației;
  - se vor evita paginile Wikipedia și depozitele colective de documente (ScriTube, Scribd, Referate etc.); în cazul Wikipedia se pot folosi sursele indicate pe orice pagină Wikipedia în secțiunea *External Links* (de unde sunt preluate informațiile sintetizate acolo);
- lucrările de tip B și C vor avea o bibliografie mai bogată în literatură științifică (în special tipul B, dar și pentru tipul C se pot invoca studii de caz pentru tehnologii sau provocări similare celor abordate de lucrare).

8. Diagramele

Principii generale:

- Diagramele sunt cel mai concis mijloc de raportare/descriere în analiza de business, în proiectarea software sau în descrierea unui software deja implementat, pentru o comunicare eficientă cu potențialii evaluatori. Acestea permit lucrării de licență să prezinte nu doar ce s-a realizat, ci și cum (partea de proiectare) și de ce (partea de analiză de business);
- Diagramele trebuie să fie în concordanță cu ce s-a implementat, să nu conțină elemente care nu se regăsesc în implementare;
- Diagramele trebuie să fie specifice pentru lucrarea realizată, și nu generaliste (să nu fie valabile pentru orice lucrare, ci distinctive și informative pentru fiecare lucrare în parte);
- Diagramele trebuie să respecte standardele de modelare adoptate (UML, BPMN, ER etc.).

Consecințe:

- Diagramele nu pot avea sursă bibliografică. Dacă au, înseamnă că ori sunt prea generaliste, ori sunt despre o altă aplicație;
- Se vor evita diagramele generaliste din literatură (MVC, client-server). Ele trebuie instanțiate și defalcate pe componentele concrete care s-au implementat, în următoarele scopuri:
    - de a oferi cititorului răspunsuri rapide la întrebări precum: *Ce fel de client s-a implementat?* (React? Angular? Android?) *Ce fel de back-end?* (PHP? Laravel?) *Ce fel de interfață?* (REST? GraphQL? Mongoose?);
    - de a permite defalcarea implementării în diagrama de componente: *Ce controllere s-au creat?* (produse, utilizatori etc.) *Ce componente React?* (afișare produse, editare produse etc.).

Așadar, elementele diagramelor trebuie să fie specifice proiectului și identificabile în cadrul proiectului:

- Exemple negative (generice): *Client, Server, Database, Entity, Table, Interface;*
- Exemple pozitive (specifice): *Controller produse, Tabel clienți, Tabel produse, serviciu de plăți Paypal, interfață HTTP de tip REST, Componentă afișare produse*[13]*.*

Tipurile de diagrame relevante pot varia în funcție de tipul lucrării. Diferențiem în continuare tipurile de diagrame după relevanță:

Pentru lucrările de tip A:

- *diagrame utile analizei de business*, pentru a evidenția potențialele beneficii ale componentei/aplicației dezvoltate într-un context de business (fictiv sau real)*:* Business Model Canvas, diagramă Fishbone, grafic Pareto, diagramă de obiective (cu accent pe obiectivele de business, și nu ale programatorului), diagramă use case, diagrame BPMN pentru procese de afaceri As-Is („before”) vs. To-Be (”after”);

[13] diagrama de arhitectură/deployment va indica și în ce tehnologii s-a implementat fiecare

- *diagrame utile în proiectare sau în descrierea implementării realizate*: diagrame arhitecturale (deployment și de componente), schițe UX[14] („wireframes”), procese de utilizare (diagramă BPMN sau de activități, diagramă de stări), diagrame ale bazei de date (conceptuală și fizică), diagrama fluxurilor de date (DFD), iar acolo unde e relevant și diagrama de clase;
- pentru proiectele care implementează componente ce nu pot funcționa independent (exemple: suite de testare, roboți RPA) se vor documenta prin diagrame:
    - *atât sistemul software pentru care au fost realizate* (aplicația testată, aplicația automatizată);
    - *cât și componenta dezvoltată, prin acele diagrame care au sens* (exemple: structurile și fluxurile de date manipulate în suita de testare sau robotul de automatizare, procesele de management al testării sau al proiectului RPA etc.).

Pentru lucrările de tip B:

- aceleași diagrame de analiză de business, concentrate pe beneficiile și contextul de business ale cercetării empirice prezentate;
- diagramă de proces pentru a descrie succint metodologia de prelucrare a datelor;
- diagrame de descriere a structurilor de date manipulate (se pot adopta diagrame specifice bazelor de date din care să reiasă entitățile și atributele relevante);
- soluții de vizualizare a datelor adecvate analizei realizate și eventuale diagrame specifice metodologiei/instrumentelor adoptate în scopul prelucrării datelor.

Pentru lucrările de tip C:

- aceleași diagrame de analiză, concentrate pe beneficiile și contextul de business al organizației supuse studiului de caz (care a adoptat soluția/platforma în discuție);
- diagrame de proiectare: diagrame arhitecturale (în special deployment), diagrame pentru procesele de configurare/instalare, procesele de utilizare, iar acolo unde se manipulează date și diagrame ce surprind structurile și fluxurile de date manipulate;
- soluții de vizualizare specifice platformei analizate;
- dacă se programează anumite componente (exemplu: pentru automatizare, integrare), devin relevante și celelalte diagrame de la lucrările de tip A; dacă se pune accent pe analiza de date, devin relevante cele indicate la lucrări de tip B.

În toate cele trei tipuri de lucrări, eventualii algoritmi implementați pot fi descriși:

- fie ca pseudocod,
- fie ca diagrame flowchart (scheme logice) sau diagrame de proces/activități.

Notă: Descrierea de algoritmi direct prin exemplificarea directă a codului-sursă nu ține de capitolele de analiză/proiectare, ci de capitolul de implementare.

[14] User Experience