Model procjene količine posla za razvoj programskih rješenja zasnovan na višestrukom korištenju slučajeva uporabe

Rak, Katija

Sveučilište u Zagrebu

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Katija Rak

MODEL PROCJENE KOLIČINE POSLA ZA RAZVOJ PROGRAMSKIH RJEŠENJA ZASNOVAN NA VIŠESTRUKOM KORIŠTENJU SLUČAJEVA UPORABE

DOKTORSKI RAD

Zagreb, 2019.

Sveučilište u Zagrebu

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

KATIJA RAK

MODEL PROCJENE KOLIČINE POSLA ZA RAZVOJ PROGRAMSKIH RJEŠENJA ZASNOVAN NA VIŠESTRUKOM KORIŠTENJU SLUČAJEVA UPORABE

DOKTORSKI RAD

Mentor: prof. dr.sc. Željka Car

Zagreb, 2019.

University of Zagreb

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Katija Rak

EFFORT ESTIMATION MODEL FOR SOFTWARE DEVELOPMENT PROJECTS BASED ON USE CASE REUSABILITY

DOCTORAL THESIS

Supervisor: Professor Željka Car, PhD

Zagreb, 2019

O mentoru

Prof. dr. sc. Željka Car

Željka Car rođena je u Zagrebu. Diplomirala je, magistrirala i doktorirala u polju elektrotehnike na Fakultetu elektrotehnike i računarstva (FER) Sveučilišta u Zagrebu 1993., 1999. odnosno 2001. godine.

Od ožujka 1993. godine radi na Zavodu za telekomunikacije FER-a. U svibnju 2018. godine izabrana je u zvanje redovitog profesora. Sudjeluje kao istraživač na znanstvenim i tehnološkim projektima pod pokroviteljstvom Ministarstva znanosti, obrazovanja i sporta Republike Hrvatske i Hrvatske zaklade za znanost, projektima financirana iz fondova Europske unije i UNICEF-a, kao voditeljica u multidisciplinarnim projektima uz potporu Hrvatske akademije znanosti i umjetnosti i Sveučilišta u Zagrebu i u projektima suradnje s industrijskim partnerima. Njezini istraživački interesi su programsko inženjerstvo s fokusom na inženjerstvo programskih procesa, razvoj inovativnih ICT rješenja za ostvarenje pristupačnosti i e-inkluziju te upravljanje projektima. Koautorica je većeg broja znanstvenih i stručnih radova objavljenih u međunarodnim časopisima i zbornicima radova s međunarodnih konferencija. Voditeljica je istraživačkog Laboratorija za asistivne tehnologije i potpomognutu komunikaciju. Sudjeluje u nastavi na preddiplomskom, diplomskom, doktorskom i specijalističkom studiju na FER-u, doktorskom studiju na Edukacijsko-rehabilitacijskom fakultetu, preddiplomskom studiju Hrvatskog vojnog učilišta "Dr. Franjo Tuđman" te na diplomskom studiju Sveučilišta u Dubrovniku, studij Poslovno računarstvo.

Prof. Car dopredsjednica je IEEE interesne skupine Žene u inženjerstvu u okviru Hrvatske sekcije IEEE - Institute of Electrical and Electronic Engineers te je članica IEEE Odjela za menadžment u inženjerstvu i Odjela za komunikacije. Članica je stručnih udruga MIPRO, ELMAR i Hrvatskog društva za biomedicinsko inženjerstvo i medicinsku fiziku.

About the Supervisor

Professor Željka Car, PhD

Željka Car was born in Zagreb in 1969. She received B.Sc., M.Sc. and Ph.D. degrees in electrical engineering from the University of Zagreb, Faculty of Electrical Engineering and Computing (FER), Zagreb, Croatia, in 1993, 1999 and 2001, respectively.

From March 1993 she is working at the Department of Telecommunications at FER. In May 20118 she was promoted to Full Professor. She has participated in the projects funded by Croatian Ministry of Science, Education and Sports and Croatian science foundation, in three projects funded by EU Funds and was leader of the multidisciplinary projects funded by Croatian Academy of Science and Arts and University of Zagreb as well as several projects with industry partners. Her research interest includes software engineering with focus on software process engineering, development of innovative ICT solutions in the area of accessibility and e-inclusion and project management. She co-authored a number of scientific and professional papers, book chapters and articles in journals and conference proceedings. She is a Head of the Research Laboratory for Assistive Technology and Assisted Communication. She teaches at undergraduate, graduate, doctoral and specialist studies at FER, doctoral studies at the Faculty of Education and Rehabilitation, undergraduate study of the Croatian Military Academy "Dr. Franjo Tudjman" and graduate study at University of Dubrovnik, Business Computer Engineering.

Prof. Car is a vice-chair of Women in Engineering Affinity Group within IEEE Croatia Section Society, and member of IEEE Engineering Management and Communication chapters, ELMAR and Croatian Biomedical Engineering and Medical Physics Society.

Zahvale

Zahvaljujem prof. dr. sc. Željki Car na mentorstvu i stručnom vodstvu tijekom studija i izrade doktorske disertacije. Njezini vrijedni savjeti uvelike su doprinijeli postignutim rezultatima u okviru istraživačkog rada.

Zahvaljujem prof. dr. sc. Ignacu Lovreku na konstruktivnim sugestijama pri pisanju znanstvenog rada, kao i na nesebičnom doprinosu u objavi rada.

Hvala prof. dr. sc. Draganu Jevtiću na poučnim razgovorima iz područja strojnog učenja.

Posebnu zahvalu upućujem svojim roditeljima, sestri i bratu na moralnoj podršci i vjeri u mene. Bila je posebna privilegija odrastati uz roditelje koji su me uvijek poticali na ulaganje u obrazovanje. Mom tati, a sada napokon i kolegi, bit ću uvijek zahvalna što je prepoznao i podržao istraživački duh u meni i naučio me kontinuirano podizati ljestvicu i želju za napretkom. Mojoj mami najveće hvala na pravim i toplim riječima ohrabrenja na temelju kojih nisam posustala. I, naravno, na druženju s unukama.

Najveću zahvalu upućujem Mariu, mom ravnopravnom partneru kroz život, kao i našim curama Sofiji i Karli – na podršci, strpljenju i razumijevanju. Uz vas su moja dostignuća potpuna.

Sažetak

Analiza široko rasprostranjenih tehnika procjene količine posla za projekte razvoja programskih rješenja pokazuje da su postojeće tehnike prvenstveno namijenjene razvoju novih rješenja. Primjenom tih tehnika na projektima koji koriste prethodno razvijene artefakte iz projekata sa sličnim obimom posla dolazi do velikih odstupanja između procijenjene i stvarne količine posla. Ovim doktorskim istraživanjem predložen je novi model procjene količine posla za razvoj programskih rješenja zasnovan na višestrukom korištenju slučajeva uporabe i značajkama projektnog tima, nazvan model UCR (*Use Case Reusability model*). Model UCR nastao je modifikacijom postojećeg Karnerovog modela za procjenu količine posla UCP (*Use case Points*) te dodavanjem elemenata koji opisuju aspekt višestruke iskoristivosti. U sklopu validacijskog procesa, model UCR se primijenio na inicijalnim i slijednim projektima iz tri različita i nepovezana programa iz industrije i akademske zajednice. Rezultati analize pokazuju da se model UCR može primijeniti u različitim projektnim okruženjima. Prema promatranoj srednjoj veličini relativne greške, primjena modela UCR pokazuje poboljšane rezultate procijenjene količine posla pri usporedbi s modelom UCP za promatrane projekte unutar studijskog primjera.

Opisan je proces oblikovanja modela procjene količine posla zasnovanog na metodama strojnog učenja. Primjenom regresijskih modela (metode linearne regresije, umjetne neuronske mreže i stablo odlučivanja) oblikovan je model UCR_ML. U sklopu evaluacije rezultata, za dane povijesne podatke najpreciznije rezultate procijenjene količine posla dala je metoda radijalnih neuronskih mreža.

Ograničenje pri oblikovanju modela predstavlja relativno mali skup povijesnih podataka jer su posljedično mali skupovi podataka za učenje i testiranje. Na temelju validacije modela procjene UCR_ML, odnosno usporedbom s rezultatima procjene modela UCR, može se zaključiti da su za organizacije s ograničenim brojem projekata sa sličnim obimom posla primjereniji algoritamski modeli.

Organizacije mogu lako primijeniti opisani proces razvoja modela UCR i UCR_ML i po potrebi ih prilagoditi svojim potrebama modifikacijom ulaznih faktora, težina ili kalibracijom veličine UCR izražene u čovjek – sat.

Ključne riječi: procjena količine posla, razvoj programskog proizvoda, slučajevi uporabe, višestruka iskoristivost.

Summary

Effort estimation model for software development projects based on use case reusability

Effort estimation required for a software development project is extremely important for the success of the overall solution delivery. Despite this fact, studies show that there a significant progress in improving the performance estimation techniques has not been reported, which represents one of the major challenges within the software industry. Incorrect effort evaluation often causes budget overrun, delay in delivery, failure to fulfil contractual obligations and indirectly affects the quality of the product itself. It is therefore not surprising that a very common cause of failure of software development projects in the field of information technology (IT) is incorrect effort estimation.

The estimated amount of work has a direct impact on several aspects of the software development process life cycle: it may challenge or support a decision on the development of software product depending on the investment justification, it is used as an input parameter for determining the budget of the project and the market price, affects the project plans, schedules delivery of project artifacts, etc.

Present trends impose goals such as shorter duration of the software development cycle and cheaper product prices. This directly targets lower project effort, but there is still a parallel demand to maintain the agreed quality of the product. To meet set requirements, software development models introduce the practice of defining software modules and their implementation. Those serve as the core solution with a possibility to reuse certain modules in other (separate) software solutions. Core solution beside program code contains a vast number of development process artifacts, such as requirement descriptions, architecture, use cases, test specifications, etc. Reusability practice of software artifacts improves the productivity and quality of new software products, while reducing the resources, cost and time of the future software development projects.

Conducted studies include an analysis of the most commonly used effort estimation techniques and those can be categorized into two groups: algorithmic models based on parameters (Constructive Cost Model - COCOMO, Lines of Code – LOC, Functional Points – FP, Use

Case points, etc.) and heuristic approach (expert estimation, neural networks, a rule of thumb, techniques, Delphi, etc.).

Analysis of the widely spread effort estimation techniques for software development projects show that these techniques were primarily intended for the development of new software solutions. By applying these techniques to the projects that are reusing artifacts previously developed in past projects with a similar scope of work, there is great discrepancy between the estimated and actual project effort. Such results clearly show there is a strong need for defining a new effort estimate model that would serve the projects which are reusing artifacts developed in previous projects within the same program. The term "initial project" refers to the project where all project artifacts are developed from scratch.

This thesis describes new effort estimation model based on use case reuse and project team characteristics, called the Use Case Reusability (UCR). UCR model is modification of the existing Use Case Points (UCP) effort estimation model developed by Karner with the elements that are describing the reusability aspect. Input factors of UCR model appertained to three main aspects: functional scope, technical complexity and environmental factors. The UCR model is distinctive by inclusion of additional classification of use cases for the subsequent projects based on their reusable elements (new, similar or identical use case). In the subsequent project artifacts developed in previous projects are being reused. In the initial project where all project artifacts are built from scratch, all the use cases are defined as new use cases. Each of the subsequent projects has a dedicated functional scope and every new requirement must be analyzed to determine if it fits the elements of an already existing use case.

Each of the technical and environmental factors has been assigned a descriptive scale of attributes (low, medium, high) with their set of criteria for each attribute. By setting the clear criteria for each possible attribute of individual factor, input parameters for the estimation model are objective and transparent. Historical project data from three different programs were collected to calibrate the size of UCR expressed in man hours for initial and subsequent projects.

This paper also presents a study which validates the usage of UCR model. Validation of UCR model is performed as an experiment that evaluates the accuracy of estimated project effort when different estimation models are used. The study is conducted within industry and academic environments using industry project teams and postgraduate students as subjects. The analysis results show that UCR model can be applied in different project environments and that

according to the observed mean magnitude relative error, it produced very promising effort estimates.

Projects whose historic data was used for model calibration and validation are dominantly small sized in terms of project scope (small or moderate number of deliverables to be produced) and team size (up to five team members), so certain deviations in effort estimation might be expected for medium or large projects. As part of future work, there is a possibility for medium or large sized projects to enlarge the scale of factor attributes in the UCR model. Project size and project complexity are increasing from small to large sized projects: medium and large sized projects have a moderate to high number of deliverables which are usually technically more complex, number of team members rises and timeframe for delivery expands. Additional granularity of factor attributes would allow more precise differentiation between the projects in terms of technical complexity solution and project team characteristics.

Apart from parametric UCR model, this thesis also presents effort estimation model based on machine learning techniques. UCR_ML model was formed based on regression techniques: linear regression method, artificial neural network and decision tree. Evaluation results of all three techniques showed that the most precise estimate of project effort was produced by radial neural network. The design constraint of UCR_ML model is a relatively small set of historical data for learning and testing. Based on the validation of the UCR_ML estimation model, i.e. by comparing the results of the UCR model estimation, it can be concluded that for organizations with a limited number of projects of similar scope more appropriate are algorithmic effort estimation models. However, as experts in effort estimation advise, the application of several different estimation models contributes to a more accurate estimation of effort. Consequently, organizations with a limited amount of historical data should not rule out the use of a model based on machine learning methods, but after designing them, they should be enlarged with new historical data.

Organizations can easily apply the described both UCR and UCR_ML models, and if necessary, tailor them by modifying the input factors and weights or calibrating UCR size expressed in the man hour.

The sections of this thesis are organized as follows. The introduction gives an overview of current research along with the analysis of effort estimation models with the focus on expert estimation and use case points. This section describes mathematical model of UCP and algorithm of estimated project effort.

The second chapter describes an experience and challenges of applying UCP model in projects that are reusing previously developed artifacts. Great discrepancy between estimated and actual project effort clearly show there is a strong need for defining a new effort estimate model that would serve the projects which are reusing artifacts developed in previous projects with the similar scope.

The third section describes software reusability elements along with the process of comparing a new software requirement with the existing use cases. In addition to the existing classification of cases by complexity, a new classification of use cases is established based on their reusable elements.

The fourth section proposes the definition of a new effort estimation model based on use case reuse (UCR). The UCR model introduces new classification of use cases based on their reusability and it includes only those technical and environmental factors of UCP model that according to the effort estimation experts have significant impact on effort for the target projects. This section also describes mathematical model along with all algorithms for producing effort estimation followed by model parameterization.

Furthermore, the fifth section presents the validation process of the UCR model. Validation is performed as an experiment that evaluates the accuracy of estimated project effort when different estimation models are used. The factor in the experiment is the estimation model and the treatments are UCP and UCR models. Subjects are different project teams from the industry and academic community (Ph.D. students).

Effort estimation model based on machine learning techniques UCR_ML is defined within the sixth section. Case study describes development of the UCR_ML model using regression techniques: linear regression, neural networks and decision trees. Effort estimation results of UCR_ML were validated against the results of UCR model. As part of the case study, the UCR_ML model based on machine learning shows slightly inferior to the UCR algorithm model. Under the threat of the validity of the results, a relatively small set of historical data has been recorded, so there are small sets of data for learning and testing. It can be concluded that for organizations with a limited number of projects with a similar scope of work, who want to develop a suitable model for estimating the effort, more appropriate are the algorithmic models. However, organizations with a greater number of historical data can easily apply the described estimation model and, if necessary, adjust input factors.

The last chapter outlines the conclusions of the dissertation research and suggests the directions for further research in this area.

Key words: effort estimation, use case points, software development, use cases, reusability.

Sadržaj

1. Uvod

Procjena količine posla (*effort estimation*) za razvoj programskog proizvoda od iznimne je važnosti za uspjeh cjelokupne isporuke programskog rješenja. Unatoč toj činjenici, istraživanja [1] pokazuju kako je mali napredak postignut u poboljšanju performansi tehnika procjene (estimacije), što predstavlja jedan od glavnih izazova unutar softverske industrije [2]. Netočna procjena često uzrokuje prekoračenje proračuna, kašnjenja u isporuci, neispunjavanje ugovornih obveza te uz ostale posljedice, indirektno utječe i na samu kvalitetu proizvoda. Stoga i ne čudi što je kao vrlo česti uzrok neuspjeha projekata razvoja softvera u području informacijske i komunikacijske tehnologije (IKT) navedena upravo netočna procjena količine posla [3].

Procijenjena količina posla izravno utječe na više aspekata životnog ciklusa razvojnog procesa: može osporiti ili podržati odluku o razvoju softverskog proizvoda ovisno o investicijskoj opravdanosti, koristi se kao jedan od ulaznih parametra za određivanje proračuna projekta i tržišne cijene, utječe na projektne planove, raspored isporuke projektnih artefakata, itd. U praksi je često procijenjena količina posla manja od stvarne količine posla na kraju projekta [4]. Takve optimistične procjene idu na ruku budućem korisniku (kupcu) jer nerijetko „optimistična“ procjena opravdava nižu, a time i kompetitivnu cijenu proizvoda. Međutim, isti se projektni tim pri razvoju proizvoda susreće s mnogim izazovima – od kojih su najčešći, već spomenuti, prekoračenje zadanog proračuna i kašnjenje isporuke [1] [5]. Prosječno odstupanje odnosno prekoračenje od planirane količine posla u provedenim istraživanjima iznosi oko 30% [5]; razlozi prekoračenja su kompleksni te nisu adekvatno elaborirani u dosadašnjim istraživanjima o procjeni količine posla [1].

S druge strane, precizna procjena količine posla projekta utječe na efikasno korištenje postojećih resursa; u slučajevima kad je iznos procijenjene količine posle veći od iznosa stvarne količine posla (planirani iznos je precijenjen), postoji mogućnost da su ljudski resursi unaprijed bili angažirani za projekt u većoj mjeri nego što je bilo potrebno za samo izvođenje projekta.

Današnji trendovi nameću ciljeve poput što kraćeg trajanja razvojnog ciklusa softvera i što jeftinije cijene proizvoda. Time se izravno cilja i na manju količinu posla, međutim ne smije se negativno

utjecati na kvalitetu proizvoda. Kao jedan od mogućih odgovora na postavljene zahtjeve, u programskom inženjerstvu i različitim modelima razvoja softvera uvodi se praksa definiranja i implementacije programskih modula koji služe kao jezgreno rješenje (*core asset*) s ciljem višestrukog korištenja (*reusability*) pojedinih modula u drugim (odvojenim) softverskim rješenjima. Jezgreno rješenje uz programski kod može sadržavati veliki broj artefakta razvojnog procesa kao što su opisi zahtjeva, arhitekture, slučajeve uporabe (*use cases*), specifikacije testiranja, itd. Takvim praksama poboljšava se produktivnost i kvaliteta programskog proizvoda, a smanjuju se resursi, trošak te vrijeme razvoja programskog proizvoda u budućim projektima [6] [7] [8].

U pomno planiranim projektima razvoja softvera moguće je unaprijed predvidjeti da će se pojedina programska rješenja moći implementirati u više okolina (npr. zbog potrebe za rješenjem istih ili vrlo sličnih funkcionalnosti u različitim okolinama) te je takvu pretpostavku nužno uvažiti prilikom procjene količine posla za svaku od planiranih implementacija. Prije početka razvoja jezgrenog modula programskog rješenja treba uzeti u obzir nužnu karakteristiku višestruke iskoristivosti programskih komponenti i utjecaj te karakteristike na količinu posla razvoja u prvom projektu implementacije. S druge strane, pri procjeni količine posla za daljnje projekte implementacija programskog proizvoda treba analizirati utjecaj iskoristivosti već postojećih modula.

Dosadašnja istraživanja obuhvatila su analizu najčešće korištenih tehnika za procjenu koje se mogu kategorizirati u dvije skupine [9] [10] [11]: algoritamski modeli zasnovani na parametrima (*Constructive Cost Model* - COCOMO, *Lines of Code* - LOC, *Functional Points* - FP, Use Case modeli, itd.) te heuristički pristup [12] (ekspertna procjena, neuronske mreže, pravilo palca, tehnika Delphi, itd.). Sve tehnike zasnovane na heuristici smatraju se „mekanim“ (*soft*) tehnikama jer se procjena ne izračunava prema specificiranom modelu.

Analiza gore navedenih tehnika za procjenu [10] [11] [13] pokazala je da su navedeni modeli primarno namijenjeni za razvoj novog programskog rješenja. Radi nedostatka primjerene tehnike za procjenu količine posla koja obuhvaća aspekt iskoristivosti, u nekim od promatranih projekata razvoja alternativa je korištenje ekspertne procjene, također s odstupanjem procijenjene količine posla u odnosu na stvarnu.

Opisani rezultati jasno pokazuju potrebu za definicijom novog modela za procjenu količine posla koji bi se primjenjivao u projektima koji koriste artefakte razvijene u prethodnim projektima. Preporuka je da bi organizacije prvenstveno trebale automatizirati procedure procjene, prilagoditi alate i pristupe svojim potrebama [9]. Alati koji u sebi imaju ugrađene algoritamske modele temeljene na parametrima kojima se opisuju zahtjevi programskog rješenja pružaju standardizirani proces procjene te je stoga fokus istraživanja u sklopu ovog rada usmjeren upravo prema algoritamskim modelima.

Prema trenutnom istraživanju literature, do sada niti jedan od modela procjene koji su opisani u literaturi nije se bavio aspektima iskoristivosti. Doprinos u sklopu ovog doktorskog istraživanja je matematički model procjene količine posla koji se može prilagoditi projektima koji koriste artefakte prethodno razvijene u prošlim projektima sa sličnim obimom posla. Pojam "projekti sa sličnim obimom posla" odnosi se na one projekte koji imaju slične funkcijske zahtjeve. Termin "inicijalni projekt" upotrijebit će se za označavanje projekta u kojem su svi artefakti projekta razvijeni od nule, dok se pojam „slijedni projekt“ odnosi se na projekte koji se odvijaju nakon inicijalnog projekta te koriste ranije razvijene artefakte.

Novi model za procjenu količine posla mjerit će višestruku iskoristivost na temelju slučajeva uporabe iz prethodnih projekata i značajkama projektnog tima. Pored algoritamskog modela, cilj ovog istraživanja je oblikovati model za procjenu količine posla zasnovanog na metodama strojnog učenja s dinamičkom prilagodbom parametra. Primjenom sustava učenja namjera je poboljšati kroz vrijeme rezultate modela procjene koristeći nove povijesne podatke.

Uvod rada donosi pregled dosadašnjeg istraživanja i analizu postojećih metoda za procjenu količine posla s naglaskom na ekspertnu procjenu i procjenu količine posla pomoću slučajeva uporabe (*Use Case Points* - UCP) [15]. Opisan je matematički model UCP-a i algoritam izračuna procijenjene količine posla.

Drugo poglavlje opisuje iskustvo i izazove primjene postojećih modela procjene količine posla u projektima koji iskorištavaju artefakte iz prethodnih projekata. Postojeći modeli procjene bilježe velika odstupanja u procijenjenoj količini posla za slijedne projekte. Javlja se potreba za definicijom novog modela procjene količine posla koji bi otklonio opisani problem.

Tematika trećeg poglavlja odnosi na višestruku iskoristivost programskog proizvoda. Definira se proces usporedbe novog programskog zahtjeva s postojećim slučajevima uporabe. Uz postojeću klasifikaciju slučajeva uporabe prema složenosti, postavlja se nova klasifikacija slučajeva uporabe prema kriteriju iskoristivosti.

U četvrtom poglavlju predložena je definicija novog modela procjene količine posla zasnovanog na višestrukom korištenju slučajeva uporabe – UCR (*Use Case Reusability*). Ulazni faktori modela UCP analizirani su kvantitativnom analizom stručnjaka za procjenu količine posla te se neki od faktora uključuju u novi model procjene količine posla. Definiraju se novi ulazni faktori te se isti uključuju u novi model procjene količine posla koji bi obuhvatio aspekt iskoristivosti prethodno razvijenih artefakata. Oblikovan je matematički model i svi pripadni algoritmi za određivanje količine posla te su određeni parametri modela UCR.

Slijedno, u petom poglavlju je provedena validacija modela UCR empirijskim ispitivanjem. Cilj postupka validacije je evaluacija procijenjene količine posla iz perspektive projektnog tima u kontekstu industrijskih i akademskih projekata. U sklopu eksperimenta uspoređivani su rezultati procjene količine posla primjenom dvaju modela - UCP i UCR.

Nakon algoritamskog, u šestom poglavlju oblikovan je model procjene količine posla zasnovan na tehnikama nadziranog strojnog učenja UCR_ML. U studijskom primjeru oblikovan je model uporabom regresijskih tehnika: linearne regresije, neuronskih mreža i stabla odlučivanja. Provedena je usporedba rezultata dvaju modela UCR i UCR_ML.

Posljednje sedmo poglavlje opisuje donesene zaključke disertacijskog istraživanja te predlaže pravce nastavka istraživanja u ovom području.

1.1. Analiza postojećih metoda za procjenu količine posla

Često korišteni modeli procjene količine posla pokazuju dobre rezultate u procjeni količine posla za projekte razvoja softvera u kojem se artefakti razvijaju iz temelja. U takvim se projektima prema zadanom obimu posla artefakti (analiza zahtjeva, dizajn arhitekture, programski kod, testne specifikacije i ostala projektna dokumentacija) definiraju od početka do kraja unutar tog projektnog tima bez dodavanja ili prilagođavanja dijelova dokumentacije ili softverskog koda iz nekog drugog projekta.

Međutim, očit je nedostatak dokazane metode procjene koje uzima u obzir aspekt iskoristivosti, odnosno korištenja artefakata koji su ranije razvijeni za potrebe prethodnih projekata. Ovo istraživanje je inicijalno usmjereno na odabir odgovarajućeg postojećeg modela procjene, a potom i na njegovu modifikaciju s novim čimbenicima koji bi opisali aspekt iskoristivosti.

Istraživanje je započeto pretragom prethodno provedenih sustavnih pregleda literature u časopisima i međunarodnih znanstvenih skupova na temelju ključne riječi (pojma): procjena količine posla za razvoj softvera (*software estimation effort*). Rezultati pronađeni u [1] [9] [14] [16] [17] [18] pomogli su procijeniti koja od postojećih procjena količine posla za projekte razvoja softvera bi bila najprikladnija osnova za novi model procjene količine posla.

Na temelju anketa i sustavnih pregleda literature pokazalo se da je model UCP, kojeg je razvio Gustav Karner [14] [15]], najprikladniji model procjene količine posla kao baza za razvoj novog modela. U okviru istraživanja i sustavnih pregleda literature, model UCP je istraživan u smislu upotrebe i procjene točnosti kao (1) izvorna metoda koju je definirao Karner, (2) hibridna tehnika u kombinaciji s drugim tehnikama procjene (npr. COCOMO), ili (3) unutar šire skupine modela procjene količine posla.

UCP je jedan od najčešćih modela za procjenu količine posla za razvoj softvera [16] [17] [19], a s druge strane, ovaj model je pogodan za različite tipove projekata, od kratkog trajanja (do mjesec dana), do onih trajanja preko jedne godine. Postoji široka primjenjivost - nema ovisnosti o softverskoj arhitekturi ili programskom jeziku (Java, Web Logic, MS Visual Studio, C ++ itd.). Razvijen je prvenstveno za metodologiju razvoja RUP (Rational Unified Process), ali je značajno da se UCP koristi i u agilnom razvoju softvera [16].

Istraživanja akademske zajednice i industrije pokazala su zanimanje za primjene modela procjene utemeljene na slučaju uporabe zbog obećavajućih rezultata i rane primjene u životnom ciklusu projekta [19]. Neki od razvijenih modela temeljeni na pristupu slučaja uporabe su: iUCP [20], e-UCP [21] i Re-UCP [22]. Svi ti modeli temeljeni su na modelu UCP uz dodatna proširenja (granulacije) postojećih klasifikacija ulaznih faktora u odnosu na UCP, dok se u modelima e-UCP i i-UCP uvode i neki novi ulazni faktori.

Izazov prilikom prilagodbe postojećeg modela je razumjeti moguće posljedice same prilagodbe. Istraživački tim mora razumjeti uvjete korištenja, matematički model i potencijalna ograničenja. Za UCP model ovi elementi su jasno opisani i razumljivi korisnicima modela, što UCP čini prikladnom bazom za izgradnju novog modela procjene količine posla.

1.1.1. Ekspertna procjena

Unatoč dostupnim metodama za procjenu količine posla i alatima koji omogućuju izračun procijenjene količine posla, najraširenija metoda procjene je i dalje ekspertna procjena [12] [13] [16]. Ekspertna procjena može se definirati kao mišljenje pojedinog stručnjaka ili grupe stručnjaka vezano uz određenu materiju ili nepoznatu mjeru [23]. Ova vrsta procjene primjerena je za projekte za koje empirijski podaci nisu lako dostupni i pri procjeni složenih, slabo definiranih i nerazumljivih problema [24]. Stručnjakom se smatra osoba koja ima iskustvo u promatranom području te je prepoznata od strane svojih kolega (ili onih koji provode procjenu) kao kvalificirani resurs za davanje odgovora na postavljena pitanja [23] [25].

Kao razlog širokog korištenja ekspertne procjene sudionici u istraživanjima [26] navode sljedeće razloge:

- brza procjena,
- ne iziskuje veliki broj resursa (u smislu vremena i troška),
- može biti precizna kao i „skuplje“ metode procjene [27] [28].

S druge strane, ova procjena nosi i svoje nedostatke:

- procjena je subjektivna i sklona greškama,

- veći broj stručnjaka s istim informacijama o proizvodu i projektu mogu donijeti različite procjene,
- proces stručne procjene nije konzistentan niti strukturiran,
- proces odlučivanja je sklon pristranosti naspram osobnog iskustva, dostupnih resursa, sredstava i drugih utjecaja,
- proces odlučivanja o procjeni poznat je samo stručnjaku koji provodi procjenu,
- teško je napraviti procjenu pri iskorištavanju ili modifikaciji pojedinog dijela programskog proizvoda,
- procjena se otežano kvantificira i provjerava (evaluira),
- napusti li stručnjak organizaciju, njegovo znanje i iskustvo „odlazi“ s njim.

Ekspertna procjena koristila se na samom početku projekata razvoja programskih proizvoda koji su u domeni istraživanja ovog rada jer se upravo radi o „nestandardnim“ razvojnim projektima koji iskorištavaju postojeće artefakte iz prethodnih projekata razvoja programskih proizvoda sa sličnim zahtjevima. Unatoč ranije spomenutom nedostatku ove metode procjene koji spominje da je teško napraviti ekspertnu procjenu pri korištenju ili modifikaciji pojedinog dijela programskog proizvoda, u nedostatku postojećeg formalne metode (modela) za takve projekte ekspertna procjena je ostala dostupna metoda za procjenu te je stoga i izabrana za korištenje uz metodu UCP.

Kako bi se izbjegao utjecaj pojedinačnog „subjektivnog mišljenja“ o procijenjenoj količini rada, u praksi se usuglašavaju mišljenja više stručnjaka različitim metodama (npr. tehnika Delphi, *Poker planning*).

1.1.2. Procjena količine posla pomoću slučajeva uporabe (*Use Case Points*)

1.1.1.1. Slučaj uporabe (*Use Case*)

Pojam „slučaj uporabe“ ili „slučaj korištenja“ (*Use Case*) opisuje način na koji korisnik koristi sustav. Podrazumijeva listu mogućih interakcija između sustava (programskog proizvoda) koji se razvija i vanjskih sudionika (aktera) radi postizanja specifičnog cilja. Akteri predstavljaju ljude ili informacijske sustave te je jedan od aktera i sustav koji se razvija, a koji ujedno i surađuje s ostalim akterima [29].

Namjena slučaja uporabe je postizanje cilja za aktera, a kako bi se postigao cilj, mora se izvesti aktivnost [30]. Svi akteri imaju listu odgovornosti; aktivnost povezuje cilj jednog aktera s odgovornošću drugog aktera [31]. Dva su tipa aktera: **primarni** i **sekundarni**. Primarni akter pokreće slučaj korištenja, a sekundarni je sudionik slučaja korištenja nakon što je on pokrenut [31].

Model slučajeva uporabe je skup slučajeva uporabe koji predstavljaju funkcionalnost sustava. Modeli mogu obuhvatiti vanjske entitete kao što su (ljudski) korisnici i drugi sustavi koji koriste te funkcionalnosti. Model opisuje samo poglede na ponašanje sustava od strane korisnikove percepcije i ne opisuje kako je funkcionalnost izvedena unutar sustava.

Scenarij unutar slučajeva uporabe opisuje specifičan slijed aktivnosti (radnji) kojima se ilustrira ponašanje. **Glavni scenarij** (uspješan scenarij) opisuje najčešći (pozitivan) ishod izvođenja aktivnosti kada se sve odvija prema planu (nema neželjenih utjecaja). Glavni scenarij dijeli se na korake slučaja uporabe koji su pisani „prirodnim jezikom (govorom)“ [32]. U praksi se nerijetko glavni scenarij naziva i **osnovni tijek izvođenja** (*basic flow*). Pored osnovnog tijeka izvođenja, slučaj uporabe mogu sačinjavati i **alternativni tijek izvođenja**: čini ga slijed koraka kojima se postiže željeni ishod slučaja uporabe, ali na drugačiji način nego što je naznačeno u osnovnom tijeku izvođenja. S druge strane, mogući neželjeni ishod izvođenja slučaja uporabe opisuje se scenarijima iznimaka. U praksi se ponekad i iznimke opisuju unutar alternativnog scenarija.

Model slučajeva uporabe može uključiti odnose između samih slučajeva uporabe. S obzirom da slučajevi uporabe predstavljaju funkcionalnosti sustava, ti odnosi indiciraju odgovarajuće odnose između funkcionalnosti sustava. Slučaj uporabe može uključivati ponašanje drugog slučaja uporabe (*include*) ili može proširiti funkcionalnost drugog slučaja uporabe (*extend*).

Uključuje li slučaj uporabe A slučaj uporabe B, to ukazuje da će slučaj uporabe A uključivati ponašanje specificirano slučajem uporabe B [30]. Odnos uključivanja izbjegava ponovno navođenje teksta u alternativnom scenariju u više slučajeva uporabe [33]. Sadrži broj koraka unutar glavnog scenarija pri kojem se izvršava uključivanje, uvjet koji mora biti ispunjen pri tom koraku i pobrojeni slijed koraka koji čine uključivanje [32].

Generalizacijom se povezuju dva aktera ili dva slučaja uporabe sličnog ponašanja. U slučaju da se generalizacijom povezuju dva aktera, specifičniji akter preuzima sve uloge apstraktnijeg uz

dodatak novih uloga. Kod korištenja generalizacije između dva slučaja uporabe, specifičniji proširuje odnosno mijenja funkcionalnosti apstraktnijeg.

1.1.1.2. Matematički model procjene količine posla pomoću slučajeva uporabe

Procjenu količine posla zasnovanu na slučajevima uporabe moguće je provesti u ranim fazama projekta ako je dostupan opis domene problema koji se nastoji riješiti razvojem softvera (obim posla) te veličine sustava i arhitekture u trenutku izvođenja procjene [34] [35].

Model procjene količine posla na temelju slučajeva uporabe (model UCP u daljnjem tekstu) razvio je Gustav Karner [36]. Temelje je zasnovao na metodi funkcijskih točaka (*Function Points*) [37], a cilj joj je bio pojednostaviti metodu procjene količine posla za projekte koji se temelje na razvoju objektno-orijentiranog programskog proizvoda.

Klasifikacija aktera i slučajeva uporabe

Temelj modela UCP leži u analizi slučajeva uporabe sustava [36]. Prvi korak je klasifikacija aktera na **jednostavne**, **prosječne** i **složene** aktere. **Jednostavni akter** predstavlja sustav koji komunicira definiranim aplikacijskim programskim sučeljem (*Application Programming Interface* - API). **Prosječni akter** predstavlja sustav koji interakciju ostvaruje protokolom TCP/IP. **Složeni akter** predstavlja sudionika koji interakciju ostvaruje preko grafičkog korisničkog sučelja (*Graphical User Interface* – GUI) ili web aplikacije. Svakom od navedenih tipova klasifikacije dodjeljuje se težinski faktor (težina):

- Jednostavni akter (A_j) – težina: 1
- Prosječni akter (A_p) – težina: 2
- Složeni akter (A_s) – težina: 3

Neprilagođena težina aktera (*Unadjusted actor weights* - UAW) računa se kao suma produkata količine aktera i dodijeljene težine za svaki tip aktera (jednadžba 1.1):

$$UAW = (nA_j + 2 \times nA_p + 3 \times nA_s) \quad (1.1)$$

Pri čemu nA_j označava količinu jednostavnih aktera (A_j), nA_p količinu prosječnih aktera (A_p), a nA_s količinu složenih aktera (A_s).

Slučajevi uporabe također se klasificiraju na **jednostavne**, **prosječne** i **složene**, ovisno o broju transakcija unutar slučaja uporabe (uključivo i broj transakcija u alternativnom scenariju). Transakcija označava događaj (aktivnost) između aktera i sustava. Svakom od navedenih tipova klasifikacije dodjeljuje se težinski faktor (težina):

- **Jednostavni slučaj uporabe**, 3 ili manje transakcija, težina: 5
- **Prosječni slučaj uporabe**, od 4 do uključivo 7 transakcija, težina: 10
- **Složeni slučaj uporabe**, 8 ili više transakcija, težina: 15

Neprilagođena težina slučajeva uporabe (*Unadjusted use case weights* - UUCW) računa se kao suma produkata količine slučajeva uporabe i dodijeljene težine za svaki tip slučaja uporabe (jednadžba 1.2):

$$UUCW = (5 \times nUC_{\text{j}} + 10 \times nUC_{\text{p}} + 15 \times nUC_{\text{s}}) \quad (1.2)$$

Pri čemu nUC_j označava količinu jednostavnih slučajeva uporabe (UC_j), nUC_p količinu prosječnih slučajeva uporabe (UC_p), a nUC_s količinu složenih slučajeva uporabe (UC_s).

Neprilagođena količina slučajeva uporabe (*Unadjusted Use Case Point* – UUCP) računa se kao suma neprilagođenih težina aktera i slučajeva uporabe (jednadžba 1.3)

$$UUCP = UAW + UUCW \quad (1.3)$$

Tehnički faktori i faktori okoline metode UCP

Neprilagođena količina slučajeva uporabe (UUCP) prilagođava se vrijednostima koji su dodijeljeni tehničkim faktorima (tablica 1-1) i faktorima okoline (tablica 1-2). Navedeni faktori utječu na količinu posla nevezano uz veličinu programskog proizvoda; tehnički faktori definirani su pod utjecajem ranije spomenute metode funkcijskih točaka, dok se faktori okoline oslanjaju na karakteristike projektnog tima i nefunkcijske zahtjeve.

Svakom od faktora dodjeljuje se vrijednost između 0 i 5 ovisno o pretpostavljenom utjecaju na projekt. Vrijednost 0 označava da je taj faktor irelevantan za taj projekt, dok vrijednost 5 znači da je od esencijalne važnosti.

Tablica 1-1 Tehnički faktori modela UCP

Faktor	Opis (HR)	Opis (EN)	Težina	Vrijednost
T1	Distribuirani sustavi	Distributed System	2	0 – 5 Za svaki od faktora
T2	Vrijeme odaziva	Response adjectives	2	
T3	Efikasnost krajnjeg korisnika	End-user efficiency	1	
T4	Složenost procesiranja	Complex processing	1	
T5	Iskoristivost koda	Reusable code	1	
T6	Jednostavna instalacija	Easy to install	0.5	
T7	Jednostavno korištenje	Easy to use	0.5	
T8	Prenosivost (podržavanje više platformi)	Portable	2	
T9	Lako mijenjanje	Easy to change	1	
T10	Konkurentni procesi	Concurrent	1	
T11	Sigurnosni zahtjevi	Security features	1	
T12	Omogućen pristup trećim stranama	Access for third parties	1	
T13	Posebna obuka	Special training required	1	

Tablica 1-2 Faktori okoline modela UCP

	Opis (HR)	Opis (EN)	Težina	Vrijednost
F1	Poznavanje procesa RUP	Familiar with RUP	1.5	0 – 5
F2	Iskustvo u poznavanju aplikacije	Application experience	0.5	
F3	Iskustvo u radu s objektno-orijentiranom arhitekturom	Object-Oriented Experience	1	

F4	Iskustvo tima	Lead Analyst Capability	0.5	Za svaki od faktora
F5	Motivacija	Motivation	1	
F6	Stabilnost zahtjeva	Requirements stability	2	
F7	Članovi tima s djelomičnim radnim vremenom	Part – time workers	-1	
F8	Složenost programskog jezika	Difficult programming language	-1	

Faktor tehničke složenosti (*Technical Complexity Factor* – TCF) računa se na način da se prvo pomnože vrijednosti svakog od faktora T1 – T13 s dodijeljenom težinom te se sve vrijednosti zbrajaju u sumu nazvanu T_{faktor}; naposljetku se T_{faktor} množi i zbraja s određenim koeficijentima (jednadžba 1.4):

$$TCF = 0.6 + (0.01 * T_{faktor}) \quad (1.4)$$

Faktor okoline (*Environmental Factor* – EF) računa se na način da se prvo pomnože vrijednosti svakog od faktora F1 – F8 s dodijeljenom težinom te se sve vrijednosti zbrajaju u sumu nazvanu E_{faktor} te se naposljetku E_{faktor} množi i zbraja s određenim koeficijentima, jednadžba 1.5:

$$EF = 1.4 + (-0.03 \times E_{faktor}) \quad (1.5)$$

Prilagođena količina UCP računa se na sljedeći način (jednadžba 1.6):

$$UCP = UUCP * TCF * EF \quad (1.6)$$

1.1.3. Izračun procijenjene količine posla temeljem prilagođene količine UCP-a

U konačnici pri procjeni količine posla za razvoj programskog proizvoda traži se procijenjena količina posla izražena u mjeri potrebnih ljudskih resursa u vremenu, npr. čovjek – sat, čovjek – mjesec, itd. Slijedno tome, potrebno je odrediti koliko je npr. čovjek – sati potrebno za jedan UCP (jednadžba 1.6).

$$UCP = UUCP * TCF * EF \quad (1.6)$$

Karner u svom radu predlaže vrijednost od 20 čovjek – sati za jedan UCP (*person hour per UCP* – PH/UCP) za procjenu količine posla [36]. U primjerima korištenja metode iz industrijskih slučajeva nailaze se mjere od 15 do 30 čovjek – sati za jedan UCP [17] [34].

Drugačiji pristup koji se temelji na razini iskustva i stabilnosti projektnog tima neovisno o tehničkim karakteristikama proizvoda predlažu Schneider i Winters [38] [39]: analizom produkta težine i vrijednosti faktora okoline, zbraja se količina faktora okoline kojima je vrijednost spomenutog produkta ispod 3 za faktore F1 do F6 te za faktore F7 i F8 kojima je vrijednost produkta iznad 3. Za ukupnu količinu 2 ili manje, preporuča se mjera od 20 čovjek – sati za jedan UCP. Za ukupnu količinu od 3 do 4 preporuča se mjera od 28 čovjek – sati za UCP. Za ukupnu količinu iznad 4 preporuka je poboljšati performanse tima ili pak povećati mjeru na 36 čovjek – sati za UCP.

2. Opis problema i motivacija

Mogućnost korištenja ranije razvijenih artefakata kroz više projekata javlja se kod programskih proizvoda s istim ili vrlo sličnim zahtjevima. Kao primjeri mogu se navesti sljedeći slučajevi:

- veliki poslovni subjekt uvodi isti programski proizvod u više poslovnica koji mogu, a i ne moraju, biti međusobno integrirani,
- zakonodavni okvir države nalaže uvođenje fiskalnih blagajni kod određenih tipova trgovaca,
- prema direktivi EU, svaka država članica mora uvesti programski proizvod za praćenje pojedinog segmenta javne uprave, itd.

Fokusiranje organizacija koje se bave razvojem programskog proizvoda na višestruki razvoj i implementaciju programskih proizvoda s istim ili vrlo sličnim zahtjevima u više okolina omogućuje tim organizacijama da se profiliraju u dotičnom industrijskom segmentu za proizvode s ciljanom namjenom. Proizvodi koje razvijaju unutar iste industrije imaju iste ili slične poslovne procese čime članovi razvojnog tima stječu znanje o specifičnim procesima industrije.

Današnji trendovi nameću ciljeve poput što kraćeg trajanja razvojnog ciklusa softvera i što jeftinije cijene proizvoda. Time se izravno cilja i na manju količinu posla, međutim ne smije se negativno utjecati na kvalitetu proizvoda. Kao jedan od mogućih odgovora na postavljene zahtjeve, u programskom inženjerstvu i različitim modelima razvoja uvodi se praksa definiranja i implementacije programskih modula koji služe kao osnovno rješenje (*core asset*) s ciljem višestrukog korištenja (*reusability*) pojedinih modula u drugim (odvojenim) softverskim rješenjima. Osnovno rješenje može sadržavati veliki broj artefakta razvojnog procesa kao što su opisi zahtjeva, arhitekture, slučajeve uporabe (*use cases*), specifikacije testiranja, itd. Takvim praksama poboljšava se produktivnost i kvaliteta programskog proizvoda, a smanjuju se resursi, trošak te vrijeme razvoja [6] [7] [8]. Ostali moduli definirani su kao dodatni zahtjevi koji nisu obavezni za implementaciju u svim okolinama da bi osnovno rješenje moglo biti pušteno u rad (*deployed*).

Stupanj iskoristivosti artefakata u više projekata ovisi o nizu čimbenika koje bi mogli sažeti u dvije skupine:

1) vezano uz funkcijske i nefunkcijske zahtjeve proizvoda – što su zahtjevi sličniji ili čak identični, stupanj iskoristivosti je veći,

2) ugovornim stavkama između klijenta (korisnika) i dobavljača (organizacije koja razvija programski proizvod); ugovorom se definira mogu li se projektni artefakti koristiti u daljnjim projektima (npr. mogu se koristiti isključivo na projektima za tog istog klijenta, ili se uopće ne smiju koristiti niti na jednom daljnjem projektu). Za slučajeve u kojima klijent zahtijeva da su u njegovom vlasništvu svi projektni artefakti, razvojni tim i tada može iskoristiti poznavanje poslovnih procesa u narednim projektima.

U pomno planiranim razvojnim projektima moguće je unaprijed predvidjeti da će se pojedina programska rješenja moći implementirati u više okolina (npr. zbog potrebe za rješenjem istih ili vrlo sličnih funkcionalnosti u različitim okolinama) te je takvu pretpostavku nužno uvažiti prilikom procjene količine posla za svaku od planiranih implementacija. Prije početka razvoja osnovnog modula programskog rješenja treba uzeti u obzir nužnu karakteristiku višestruke iskoristivosti programskih komponenti i utjecaj te karakteristike na količinu posla tijekom razvoja u prvom ciklusu implementacije. S druge strane, pri procjeni količine posla za daljnje cikluse implementacija treba analizirati utjecaj iskoristivosti već postojećih modula.

Postojeće metode procjene količine posla temeljene na algoritamskim modelima, uključujući i metodu UCP, doživjele su široku primjenu u softverskoj industriji. Pored osnovnih modela, postoje i mnoge inačice tih metoda jer su korisnici uvidjeli da su osnovni modeli generički te da se njihovom modifikacijom omogućuje preciznija procjena količine posla za proizvode (ili projekte) posebnih karakteristika. Takav pristup modifikaciji postojećih metoda pozitivan je odgovor na preporuke organizacijama koje provode procjenu: one bi prvenstveno trebale prilagoditi procedure (modele) procjene i prateće alate prema svojim potrebama te naposljetku, isti proces procjene automatizirati [9].

2.1. Upravljanje programom i projektima za razvoj programskog proizvoda

Proces razvoja programskog proizvoda odvija se unutar projekta, a na njemu radi projektni tim. Prema sljedećim definicijama PMBOK-a [40], projekt se smatra privremenim u smislu da ima

definirani početak i kraj te u tom vremenu definirane obim posla (*scope*) i resurse. Projekt je jedinstven prema tome što se ne odvija u rutinskim operacijama, već prema specifičnim aktivnostima osmišljenim radi postizanja jedinstvenog cilja. Stoga projekt često okuplja ljude koji obično ne surađuju – ponekad iz različitih organizacija i različitih geografskih lokacija.

Projektima se mora stručno upravljati kako bi se isporučili na vrijeme, u okviru proračuna te da bi isti u konačnici doprinijeli podizanju znanja i integracije unutar organizacije.

Upravljanje projektima je primjena znanja, vještina, alata i tehnika u projektnim aktivnostima kako bi se ispunili zahtjevi projekta [40].

Program se definira kao grupa povezanih projekata; aktivnostima programa upravlja se na koordinirani način kako bi se postigla dostignuća koja nisu moguća ako se projektima upravlja individualno i neovisno.

Programi se vode na način da se optimizira upravljanje međuzavisnostima projekata. Takve aktivnosti mogu uključivati:

- Raspodjeljivanje zajedničkih i ograničenih resursa između više projekata unutar programa.
- Usmjeravanje organizacijskih ili strateških odluka koje mogu utjecati na ciljeve projekata i programa.
- Rješavanje problema i upravljanje promjenama unutar zajedničke upravljačke strukture.

2.1.1. Formiranje promatranih projekata u program

Osnovne značajke promatranih projekata unutar kojih se razvijaju programski proizvodi s identičnim ili sličnim zahtjevima ukazuju da bi se tim projektima moglo efikasnije upravljati udruže li se u zajednički program:

- ista organizacija određuje strategiju, odnosno daje smjernice za proces razvoja programskog proizvoda koji se odvija unutar više projekata,
- proces razvoja programskog proizvoda moguće je standardizirati (formalizirati) na razini programa,
- proces upravljanja projektima moguće je standardizirati na razini programa,

- resursi koji su dio organizacije mogu sudjelovati u više projekata (moguće istovremeno ili slijedno po projektima), stoga se resursima može upravljati na razini programa.

Treba uzeti u obzir i ostale čimbenike koji su prisutni pri upravljanju programom, a utječu na upravljanje pojedinačnim projektima:

- na razini programa određuje se vremenski slijed izvođenja projekata te se postavlja vremenski okvir (planirani početak i kraj) izvođenja svakog od projekata,
- upravljanje promjenama može se izvoditi na razini programa (pritom se može sagledati eventualni utjecaj promjene unutar jednog projekta na ostale projekte unutar programa),
- identifikacija problema i rizika za vrijeme izvođenja jednog projekta, doprinosi eventualno ranijem otkrivanju istih u drugim projektima.

Jedna od glavnih prednosti udruživanja projekata u zajednički program jest dijeljenje znanja i naučenih lekcija o procesu razvoja programskog proizvoda i upravljanju takvim projektima između svih projekata u tom programu. Gledano s aspekta količine posla potrebnog za razvoj programskog proizvoda, za postizanje maksimalne iskoristivosti svih resursa u programu izrazito je važno njima kvalitetno i efikasno upravljati. Pri tome treba uzeti u obzir da projektni tim čine stručnjaci s adekvatnim znanjem i vještinama za izvođenje svog posla (za dodijeljenu rolu u projektu) te sukladno potrebama, potrebno je donijeti pravovremene odluke o njihovom angažmanu u programu.

2.2. Studijski primjer – primjene postojećih metoda procjene količine posla u industrijskim projektima

Studijski primjer analiziran je u okviru programa softverske industrije koji se sastojao od više od dvadeset projekata koji su imali vrlo sličan obim posla: razvoj rješenja koja su omogućila integraciju različitih aplikacija za upravljanje informacijskim sustavom. Svaki projekt obuhvatio je opseg aktivnosti integracije uključujući specifične zahtjeve koje je nametnuo pojedini klijent. Aplikacije za upravljanje sustavom koje koristi veliki broj različitih klijenata imale su iste ili vrlo slične funkcionalnosti.

Projekti su bili ujedinjeni u program kako bi se njima učinkovitije upravljalo poštujući sljedeće smjernice: ljudski resursi su bili dijeljeni između projekata u programu, projektni problemi i rizici

koji su bili primjenjivi za dva ili više projekta upravljani su na razini programa te metodologija i proces razvoja softvera bili su usklađeni za sve projekte.

Program je usvojio metodologiju razvoja softvera RUP. Upotreba modela slučaja uporabe pokazala je da se u smislu veličine projekata svi projekti mogu karakterizirati kao mali prema njihovom opsegu. Broj aktera varirao je od tri do četiri, a broj slučajeva uporabe varirao je od deset do dvanaest.

Jedna od glavnih prednosti ujedinjenja projekata u zajednički program bila je razmjena znanja i naučenih lekcija o procesu razvoja softvera. Za postizanje maksimalne iskoristivosti svih resursa u programu, bilo je izuzetno važno pravilno i učinkovito upravljati njima. Treba uzeti u obzir da se projektni tim sastojao od stručnjaka s odgovarajućim znanjima i vještinama za obavljanje zadane uloge u projektu. Osim toga, bilo je potrebno donijeti pravovremene odluke o njihovom angažmanu u projektima. Precizna procjena količine posla za svaki od projekata bila je vrlo važna za nekoliko aspekata upravljanja programom; na početku programa utjecala je na pojedinačne planove projekata, troškove projekata i upravljanje resursima.

Na početku programa odlučeno je da bi softversko rješenje (uključujući cjelokupnu isporuku) trebalo biti primjenjivo u više projekata (poštujući ugovorne obveze za svaki projekt). Artefakti za višestruku uporabu obuhvaćali su ne samo izvorni kod (ili dijelove izvornog koda) nego i projektne dokumente: obim posla, arhitektura rješenja, kontekst sustava, model poslovnih procesa, model uporabe slučaja i testne specifikacije. Višestruka uporaba pojedinih artefakata razmatrana je tijekom analize funkcijskih zahtjeva, dok je tijekom faze izgradnje odlučeno koji elementi postojećih artefakata će se ponovno koristiti u promatranom projektu. S druge strane, količina posla je trebala biti procijenjena tijekom faze planiranja na temelju „grubog“ (*high level*) opisa funkcionalnosti i karakteristika projektnog tima. Projektni tim trebao je pažljivo pratiti dobro definirana pravila o dokumentiranju svakog artefakta za isporuku kako bi se osiguralo da drugi članovi projekta mogu lako procijeniti može li se određeni artefakt isporuke primijeniti u drugim projektima. Artefakti isporuke bili su dostupni projektnom timu u zajedničkom repozitoriju dokumenata (s određenim ograničenjima pristupa na temelju uloge u projektu). Paralelno, projektni tim je uspostavio mehanizam praćenja ponovne uporabe artefakata u drugim projektima.

Količina posla za prvih pet projekata programa procijenjena je koristeći dvije metode: ekspertna procjena i model UCP. Nakon završetka svakog projekta tim je uspoređivao procijenjenu količinu posla u odnosu na stvarnu količinu posla.

Prema modelu UCP, procijenjena količina posla za inicijalni projekt u kojem su svi artefakti projekta razvijeni ispočetka (bez korištenja ranije razvijenih artefakata) bio je 9% veći od stvarne količine posla. Međutim, za slijedne projekte, gdje su artefakti iz prethodnih projekata ponovno korišteni, procijenjena količina posla prema UCP modelu bila je znatno veća od stvarne količine posla.

Projektni tim je pored modela UCP za procjenu količine posla koristio i ekspertnu procjenu za inicijalni i slijedne projekte. U procesu procjene sudjelovali su arhitekti, programeri i voditelj projekta te su svi imali više od 5 godina iskustva u projektima razvoja programskog proizvoda. Arhitekti i voditelj projekta su ranije sudjelovali na projektima u sklopu kojih su se koristili artefakti prethodno razvijenih softverskih rješenja. Ekspertna procjena količine posla za inicijalni projekt u kojem se razvija novi programski proizvod usuglašena je konsenzusom iskusnih članova tima i temeljena je na procjeni složenosti funkcijskih i nefunkcijskih zahtjeva. Nakon završetka inicijalnog projekta uočeno je da nema velikog odstupanja između procijenjene i stvarne količine posla. Količina posla za ostale slijedne projekte također je usuglašena konsenzusom, a izračun je temeljen na neformalnoj usporedbi obima posla inicijalnog projekta s obimom posla slijednih projekata. Pri tome je važno napomenuti da se obim posla između projekata uspoređivao na visokoj razini poznavanja zahtjeva (*high level requirements)*, odnosno na temelju glavnih funkcionalnosti sustava i poslovnih procesa. Članovi ekspertnog tima pokušali su procijeniti koliko će se ukupna količina posla za slijedne projekte smanjiti u odnosu na količinu posla za inicijalni projekt te se smanjenje posla uglavnom mjerilo u postocima. Problem uočen pri ovom izračunu je bio u tome što se taj postotak smanjenja nije mogao opravdati egzaktnim razlozima (poput povijesnih podataka, istraživanja, međuovisnosti i sl.), već se isključivo radilo o subjektivnoj procjeni pojedinca koliko bi moglo biti manje posla za svaku ulogu u slijednim projektima.

Primijećena su znatna odstupanja stvarne od procijenjene količine posla i u metodi ekspertne procjene, međutim zbog spomenute nesigurnosti pri izračunu tih procijenjenih vrijednosti i neformalnog procesa procjene, članovi tima su zaključili da navedene vrijednosti nemaju osobiti

značaj te da se nisu koristile kao ulazni parametar pri izračunu troška projekata (ili programa) niti u ostalim procesima pri upravljanju projektima i programom.

Nemogućnost precizne procjene količine posla uočen je kao važan rizik s velikim utjecajem na procese upravljanje projektima, ali i programom koji sve projekte objedinjuje. Spomenuti rizik ima izravne posljedice na sljedeće procese:

- planiranje projektnih aktivnosti i upravljanje vremenskim tokom izvođenja projekta,
- upravljanje resursima (definiciju potrebnog broja članova tima sukladno potrebnim ulogama, efikasna raspodjela posla među članovima tima),
- upravljanje troškovima – troškovi projekata računaju se izravno prema količini posla za pojedinu ulogu i trošku pojedinog resursa.

Neizravno, rizik neprecizne procjene količine posla utječe na gotovo sve aspekte upravljanja projektom. Radi eliminacije spomenutog rizika, od iznimne je važnosti definirati standardizirani proces procjene količine posla za projekte koji višestruko koriste postojeće programske komponente već razvijenog rješenja.

Raširena mjera preciznosti procjene količine posla za razvoj softverskog proizvoda jest relativna greška (*Magnitude of Relative Error* – MRE). MRE je apsolutna greška pri procjeni projekta, a što je bliža nuli, točniji je model procjene. Izražava se u postocima (jednadžba 2.1):

$$MRE = \frac{|(Stvarna\ količina\ posla - Procijenjena\ količina\ posla)|}{Stvarna\ količina\ posla} \quad (2.1)$$

Tablica 2-1 prikazuje MRE količine posla prvih pet projekata (P1 - P5) gdje je procijenjena količina posla izračunata na temelju modela UCP.

Tablica 2-1 Procijenjena i stvarna količina posla modelom UCP i MRE

Projekt	Procijenjena količina posla (UCP) u čovjek – sat	Stvarna količina posla u čovjek – sat	MRE
P1	1129	1032	9%
P2	740	283	161%
P3	649	207	214%
P4	649	197	229%

P5	649	195	233%

Model procjene UCP omogućuje procjenu količine posla za svaki pojedini projekt, temeljenu na obimu posla, funkcijskim i nefunkcijskim zahtjevima proizvoda. Takav pristup pokazao se prikladan samo za procjenu napora inicijalnog projekta unutar programa gdje su svi artefakti projekta izgrađeni otpočetka. Za slijedne projekte, tim nije mogao postaviti bilo kakav ulazni podatak u model procjene koji bi opisao da su određeni artefakti rješenja ponovno korišteni iz prethodnih projekata. Posljedično, količina posla procijenjena modelom UCP za slijedne projekte nije bila dovoljno precizna jer je znatno odstupala od stvarne količine posla. S druge strane, procijenjena količina posla prema modelu UCP istog je iznosa za slijedne projekte P3, P4 i P5 jer ulazni faktori modela ne odražavaju razlike između spomenutih projekata.

3. Postupak analize procjene višestruke iskoristivosti programskih zahtjeva i slučajeva uporabe

3.1. Višestruka iskoristivost programskog proizvoda

Pojam višestruke iskoristivosti (višestruke uporabe) programskog proizvoda (*software reuse*) podrazumijeva korištenje postojećih artefakta koji su nastali tijekom procesa razvoja programskog proizvoda. Artefakti objedinjuju programski kod, obim posla, arhitektura rješenja, kontekst sustava, model poslovnih procesa, model uporabe slučaja i testne specifikacije i ostalu prateću projektnu dokumentaciju.

Frake i Kang [41] višestruku iskoristivost programskog proizvoda definiraju kao korištenje postojećeg proizvoda ili znanja o proizvodu radi razvoja novog proizvoda. Pojedine teze zastupaju da efikasna višestruka iskoristivost proizvoda mora biti sustavna, a ne oportunistička; sustavna višestruka iskoristivost podrazumijeva pomno planirani proces s dodijeljenim resursima, dok je s druge strane oportunističko iskorištavanje *ad hoc* i prema potrebi [42].

Sustavna višestruka iskoristivost proizvoda nalaže uporabu standardiziranog procesa razvoja koji se odvija u četiri faze [43] (Slika 3-1):

1) Upravljanje infrastrukturom iskorištavanja (*Manage Reuse Infrastructure* - MRI)
2) Kreiranje iskoristivih artefakata (*Produce Reusable Assets* – PRA)
3) Održavanje iskoristivih artefakata (*Broker Reusable Assets* - BRA)
4) Uporaba iskoristivih artefakata (*Consume Reusable Assets* – CRA)

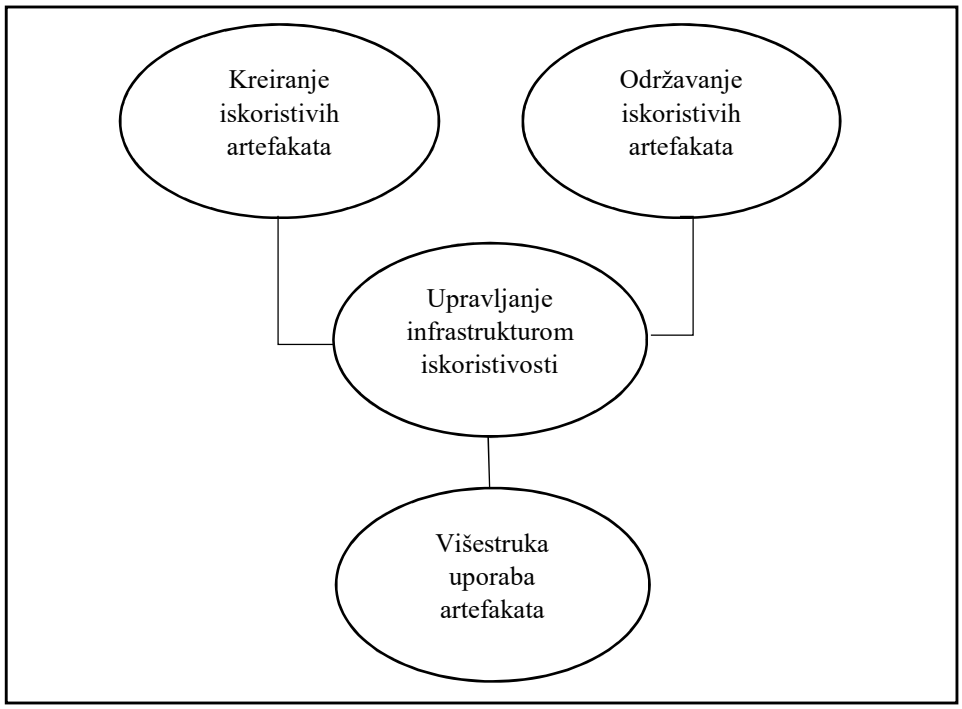
Upravljanje infrastrukturom iskorištavanja (MRI) obuhvaća definiciju pravila višestruke iskoristivosti, uloga koje sudjeluju u procesu i ciljeva koje se želi postići iskoristivošću artefakata. Određuju se konvencije i standardi za upravljanje artefaktima (dozvoljene nadopune, brisanja i promjene). U ovoj fazi planira se količina posla za iskorištavanje artefakta.

Proizvodnja artefakata za iskorištavanje (PRA) obuhvaća razvoj, proizvodnju ili reinženjering artefakata s jasnim ciljem ponovne uporabe. PRA uključuje analizu domene i reinženjering.

Analiza domene proces je identifikacije, prikupljanja, organiziranja, analiziranja sličnosti i različitosti između sustava u domeni aplikacije i arhitekture proizvoda. Reinženjering predstavlja izgradnju dijelova, metoda, alata i njihove prateće dokumentacije kako bi se riješili problemi razvoja sustava ili podsustava primjenom znanja u modelu domene i softverske arhitekture.

Održavanje iskoristivih artefakata (BRA) uključuje kvalificiranje, certificiranje i konfiguraciju artefakata. Aktivnosti također uključuju razvrstavanje i pretraživanje u katalogu artefakta za ponovnu uporabu.

Uporaba iskoristivih artefakata (CRA) podrazumijeva izgradnju sustava pomoću iskoristivih artefakata. Katalog i povezani alati pomažu korisnicima u razumijevanju raspoloživih artefakata, identifikaciji i pretraživanju potrebnih komponenti i integraciji komponenata u novi sustav. Također, katalog se ažurira podacima o korištenim komponentama.



Slika 3-1 Proces višestrukog iskorištavanja programskog proizvoda [43]

S druge strane, istraživanja ukazuju i na pojedine negativne posljedice pretjeranog planiranja iskoristivosti u sklopu gore opisanog sustavnog pristupa [44] [45]:

- razvoj artefakata na način da se omogući višestruko iskorištavanje značajno poskupljuje trošak razvoja te je ekonomski neizvedivo razviti sve artefakte da se jednostavno iskorištavaju [46],
- s obzirom da se ne može svaki artefakt razviti s ciljem višestrukog iskorištavanja, često se moraju donositi teške odluke koji artefakti će se razviti na takav način [47],
- za artefakte razvijene s ciljem višestrukog iskorištavanja često se postavljaju određene pretpostavke i uvjeti o mogućnostima iskorištavanja, čime se smanjuje kontekst uporabe [48].

Unutar organizacija koje razvijaju slične proizvode za specifičnu domenu problema pojavljuju se izvrsne prilike za iskorištavanje proizvoda jer se eksploatiraju sličnosti tih aplikacija [49]. Većina

istraživanja vezana za tu temu fokusirana su na iskoristivost koda i arhitekture proizvoda, međutim potrebno je uzeti u obzir i znanja o razvoju proizvoda koja su također iskoristiva [50].

3.2. Specifikacija novih programskih zahtjeva pomoću slučajeva uporabe

U sferi programskog inženjerstva primarni preduvjet za višestruku iskoristivost ranije definiranog artefakta jest definiranje srodnog zahtjeva i uporabljivost (kompatibilnost) istog u razvoju drugog proizvoda.

U okviru inženjerstva zahtjeva, slučajem uporabe opisuje se zahtjev za razvoj programskog proizvoda. U inicijalnom projektu, model slučajeva uporabe opisuje zahtjeve postavljene u prvom projektu u sklopu kojeg se razvija novi programski proizvod. S obzirom da se slučaj uporabe opisuje slobodnim tekstom, da bi se izbjegle različite forme slučajeva uporabe preporuka je koristiti univerzalni obrazac s nomenklaturom naziva i podacima koje jedan slučaj uporabe mora sadržavati. Također, postavljena su dodatna pravila o informacijama koje treba sadržavati kako bi se slučajevi uporabe mogli lako pregledati i identificirati u slijednim projektima sa sličnim zahtjevima:

- naslov slučaja uporabe – kratki opis funkcionalnosti s naglaskom na primarni cilj izvođenja slučaja uporabe, bez suvišnih detalja,
- opis slučaja uporabe (*Use Case Overview or Scope*) – opis radnje koje pojedini akteri izvode,
- akteri (*Actors*) – lista aktera koji sudjeluju u tom slučaju uporabe,
- osnovni tijek događaja *(basic flow)* – lista detaljnih aktivnosti navedenih prema redoslijedu izvođenja; opis uspješnog scenarija,
- alternativni tijek događaja (*alternative flow*) – lista detaljnih aktivnosti koje se mogu dogoditi, a nisu dio uspješnog scenarija (uključuje i iznimke, odnosno neželjeni ishod),
- preduvjeti za ispunjenje događaja (*pre-conditions*) – opis preduvjeta koji moraju biti ispunjeni kako bi se slučaj uporabe izveo,
- rezultat izvođenja događaja (*post-conditions*) – opis daljnjih događaja koji će se dogoditi ako se predmetni slučaj uporabe uspješno izvede,

- dio obima posla - naziv programskog proizvoda (projekta) u kojem se predmetni slučaj uporabe koristi (uključen u obim posla),
- nasljeđivanje (*inheritance*) - opis nasljeđivanja drugog slučaja uporabe.

Svi slučajevi uporabe svih proizvoda unutar programa navode se u katalogu slučajeva uporabe programa; kod razvoja proizvoda većeg obima posla, preporuka je slučajeve uporabe grupirati prema modulima, odnosno grupama funkcionalnosti.

Po sličnom principu definirani su obrasci za opis aktera s podacima koje je potrebno navesti:

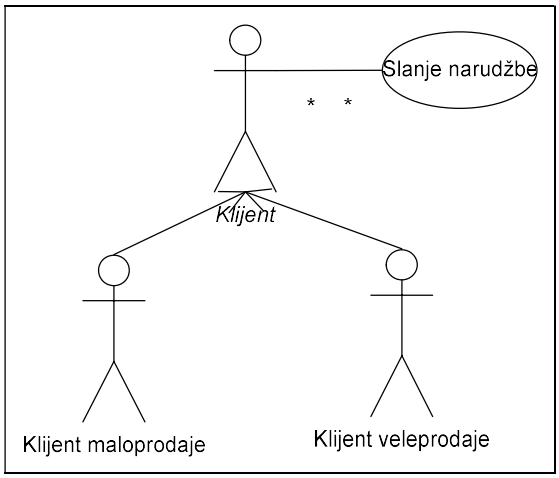
- opis aktera – kratki opis subjekta (osobe ili sustava) kojeg predstavlja,
- odnos – opis eventualnog odnos s drugim akterima,
- nasljeđivanje – opis nasljeđivanja drugog aktera.

Uz definiciju slučajeva uporabe i aktera obvezno je održavati rječnik pojmova vezano uz specifične termine koji se javljaju u poslovnim procesima kako bi se unutar projekata koristili isti pojmovi i time zadržala sljedivost (*traceability*).

Nakon analize svih zahtjeva novog programskog proizvoda, kreirani su katalozi slučajeva uporabe (ranije opisanom metodom) i katalog aktera.

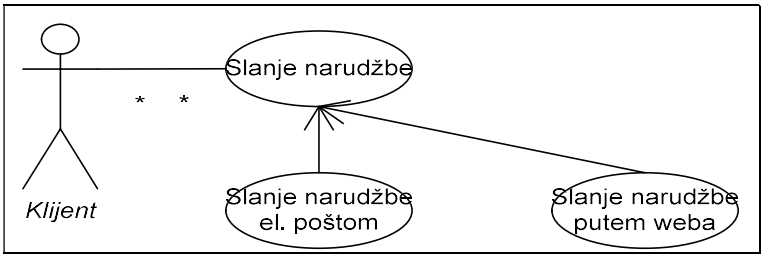
Generalizacija aktera i slučajeva uporabe

Generalizacija aktera označava da jedan akter može naslijediti rolu drugog aktera. Pri tome, akter **nasljednik** nasljeđuje sve slučajeve uporabe svog **pretka**. Nasljednik može imati jedan ili više slučajeva uporabe koji se vežu samo uz tog aktera. Primjer generalizacije aktera je prikazan na slici (Slika 3-2).



Slika 3-2 Primjer generalizacije aktera

Generalizacija slučaja uporabe istim principom označava da jedan slučaj uporabe može naslijediti opis drugog slučaja uporabe; ponašanje slučaja uporabe **pretka** nasljeđuje **nasljednik** (primjer Slika 3-3).



Slika 3-3 Primjer generalizacije slučaja uporabe

Opisana asocijacija generalizacije može znatno doprinijeti efikasnoj iskoristivosti slučaja uporabe; već sam čin generalizacije uvjetovan je sličnošću slučajeva uporabe. Pri pretraživanju sličnih slučajeva uporabe i aktera u pripadnim katalozima, pregledom samo onih koji predstavljaju pretka bitno se sužava pretraga, a već se u opisu može naslutiti radi li se o sličnom slučaju uporabe ili akteru. Stoga se projektnim timovima preporuča da se u katalog aktera i slučajeva uporabe unesu i oni koji predstavljaju pretka, a u pododjeljku oni koji predstavljaju nasljednike (primjer kataloga slučajeva uporabe programa Slika 3-4). Za svaki slučaj uporabe u katalogu se definira je li dio obima posla promatranih projekata unutar programa (stupac „dio projekta“).

Katalog slučajeva uporabe programa					
#	Slučaj uporabe		Predak/ nasljednik	Dio Projekta 1	Dio Projekta 2
1.0	Slanje narudžbe		Predak	DA	DA
1.1		Slanje narudžbe elektronskom poštom	Nasljednik	NE	DA
1.2		Slanje narudžbe putem weba	Nasljednik	DA	DA
1.3		Slanje narudžbe putem telefona	Nasljednik	NE	DA
2.0	Zaprimanje narudžbe		Predak	DA	DA
2.1		Zaprimanje narudžbe putem weba	Nasljednik	DA	DA
3.0	Ručni unos narudžbe		Predak	NE	DA

Slika 3-4 Primjer kataloga slučajeva uporabe programa

Proces generalizacije aktera i slučajeva uporabe moguće je provesti već kod razvoja novog programskog proizvoda ukoliko identificirani slučajevi uporabe zadovoljavaju tražene preduvjete generalizacije (nasljeđivanje ponašanja).

3.3. Proces usporedbe novog programskog zahtjeva s postojećim slučajevima uporabe

Pri pokretanju projekta razvoja programskog proizvoda unutar programa potrebno je prikupiti zahtjeve za taj programski proizvod. Unatoč tome što je već u fazi planiranja programa postavljena pretpostavka da će programski proizvodi razvijani u sklopu projekata imati slične poslovne zahtjeve, svaki od tih programskih proizvoda ima svoj zasebni skup zahtjeva. Zahtjeve je potrebno prikupiti, opisati, analizirati te s obzirom na specifičnosti programa, utvrditi koji zahtjevi su već bili definirani u sklopu ranijih projekata.

U inicijalnom projektu gdje su svi artefakti projekta izgrađeni otpočetka, svi slučajevi uporabe definirani su kao novi slučajevi uporabe. Svaki od sljedećih projekata ima svoj obim posla, a svaki novi zahtjev mora se analizirati kako bi se utvrdilo odgovara li elementima već postojećeg slučaja upotrebe iz prethodnog projekta. Analiza zahtjeva provodi se na sljedeći način: svi elementi postojećih slučajeva uporabe (naslov, opis, osnovni tijek događaja, akter, alternativni tijek događaja, preduvjeti za ispunjenje događaja, rezultat izvođenja događaja) moraju se pročitati kako bi se razumjelo da li prema sadržaju zadovoljavaju nove zahtjeve. Grupiranje slučajeva upotrebe u funkcijska područja pomaže projektnom timu da pregleda manji broj postojećih slučajeva upotrebe.

Postojeći slučajevi uporabe mogu biti grupirani u slučajeve uporabe prema generalizaciji; za one koje imaju slučaj uporabe koji označava pretka, novi zahtjev prvo se uspoređuje s postojećim slučajem uporabe predak. Oni slučajevi uporabe koji su „samostalni“, odnosno bez nasljednika, također se u kontekstu usporedbe tretiraju kao slučaj uporabe predak, iako bez nasljednika.

Novi zahtjev se sadržajno (analizom teksta) uspoređuje s naslovom i opisom slučaja uporabe predak. Odgovaraju li sadržajno naslov i opis postojećeg slučaja uporabe (predak) novom zahtjevu, dalje se provjerava ima li postojeći slučaj uporabe nasljednika. Postoji li nasljednik, analizira se odgovaraju li ostali elementi slučaja uporabe (osnovni tijek događaja, akteri, alternativni tijek događaja, preduvjeti i rezultati izvođenja). U slučaju da svi elementi odgovaraju, promatrani slučaj uporabe u potpunosti opisuje i novi zahtjev i time se postojeći slučaj uporabe dodaje u zasebni skup slučajeva uporabe za novi programski proizvod i nosi oznaku R (*reusable*). Ne odgovaraju li ostali elementi slučaja uporabe novom zahtjevu, isti se postupak ponavlja za preostale nasljednike istog pretka. U slučaju da ostali elementi niti jednog nasljednika ne odgovaraju, definira se novi slučaj uporabe u skupu slučajeva uporabe za novi programski proizvod gdje nosi oznaku S (*similar*) te se isti slučaj uporabe dodaje u katalog slučajeva uporabe programa kao nasljednik. Posljednja aktivnost se ponavlja i u slučaju da slučaj uporabe nije imao nasljednika, a ostali elementi ne odgovaraju novom zahtjevu.

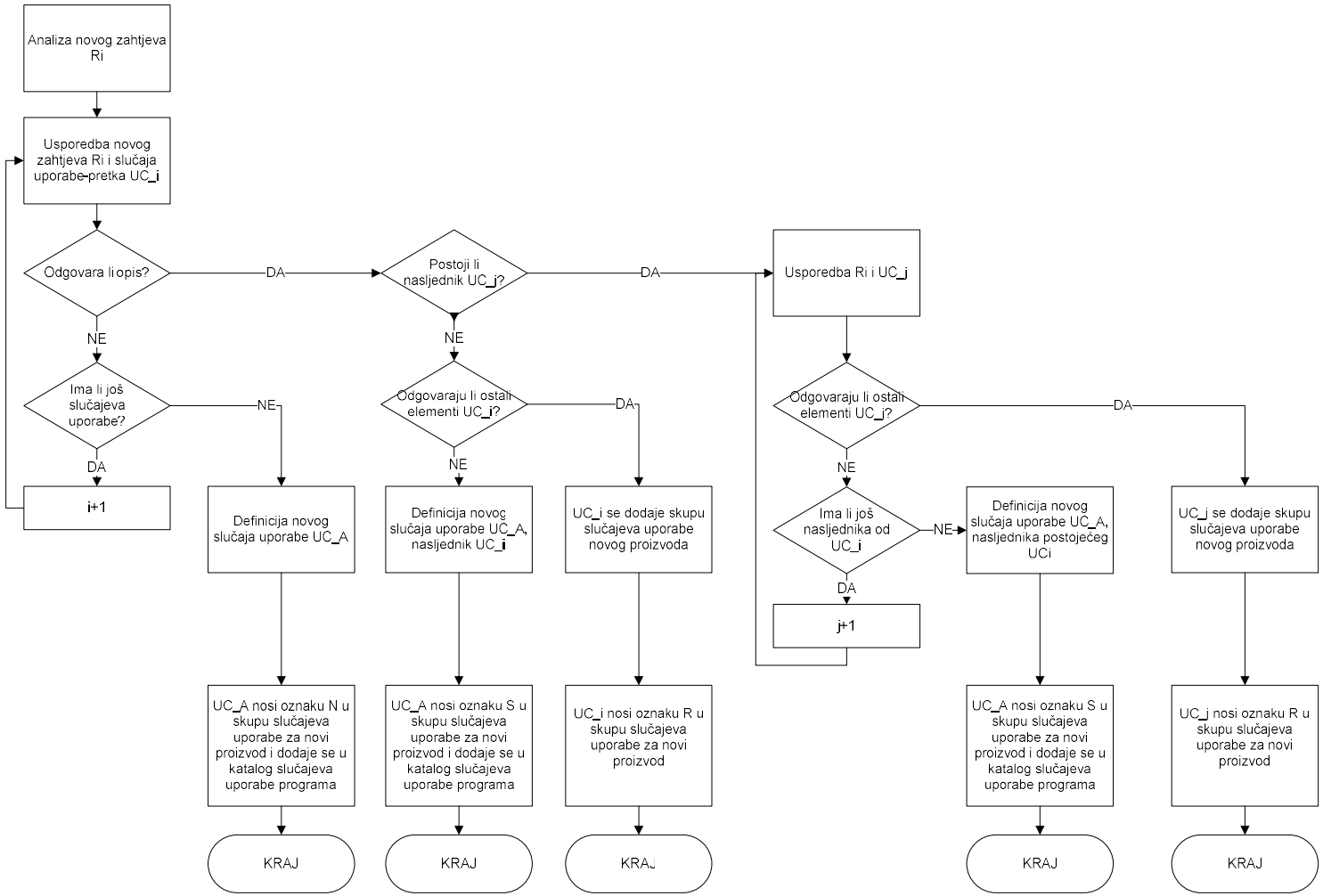
Ne odgovara li novi zahtjev opisu niti jednog slučaja uporabe, tada se definira novi slučaj uporabe koji se dodaje u skup slučajeva uporabe za novi programski proizvod i nosi oznaku N (*new*) te se također dodaje u katalog slučajeva uporabe programa.

Važnost kvalitetnog naslova i jednoznačnog opisa slučaja uporabe odražava se i u prethodno navedenim aktivnostima; slijede li se prethodno spomenute smjernice o tekstualnom opisu (informacije koje slučaj uporabe mora sadržavati, poglavlje 3.2), vrijede sljedeće pretpostavke za dijagram toka usporedbe novog zahtjeva s postojećim slučajima uporabe:

- ne odgovara li novi zahtjev opisu slučaja uporabe pretka, preskače se usporedba s nasljednicima tog slučaja uporabe,
- novi zahtjev može odgovarati opisu samo jednog slučaja uporabe pretka.

Dijagram toka opisanih aktivnosti prikazuje proces usporedbe novog zahtjeva s postojećim slučajevima uporabe, pri čemu su korištene sljedeće oznake:

- *R* predstavlja novi zahtjev,
- *UC_i* predstavlja slučaj uporabe predak definiran u prethodnom projektu,
- *UC_j* predstavlja slučaj uporabe nasljednik definiran u prethodnom projektu,
- *UC_N* predstavlja sasvim novi slučaj uporabe koji ne predstavlja bilo koji dio prethodno definiranih slučajeva uporabe,
- *UC_S* predstavlja slučaj uporabe sličan već postojećem slučaju uporabe iz prethodnog projekta,
- *UC_R* predstavlja identičan (jednak) slučaj uporabe već ranije definiran u prethodnom projektu.



Slika 3-5 Dijagram toka usporedbe novog zahtjeva (Ri) s postojećim slučajima uporabe

3.4. Nova klasifikacija slučajeva uporabe prema kriteriju iskoristivosti

Proces usporedbe novog programskog zahtjeva s postojećim slučajevima uporabe doveo je do nove klasifikacije slučajeva uporabe prema kriteriju njihove iskoristivosti; slučajevi uporabe nose oznake N, S ili R sa sljedećim obilježjima:

1) N – **novi slučaj uporabe** koji nije bio dio zahtjeva prethodnog (sličnog) programskog proizvoda,
2) S – **sličan slučaj uporabe** (ali nisu identični),
3) R – **identičan zahtjev** i slučaj uporabe je definiran u prethodnom sličnom projektu (ili više njih) unutar tog programa.

Slučaj uporabe identičan (UC_R) je postojećem slučaju uporabe samo ako su zahtjevi definirani istim elementima postojećeg slučaja uporabe: oba imaju isti naslov, opis, osnovni tijek događaja, aktere, alternativni tijek događaja, preduvjete i rezultate izvođenja.
Slučaj uporabe je sličan (UC_S) postojećem slučaju uporabe samo ako su zahtjevi definirani istim naslovom i opisom postojećeg slučaja uporabe. Ostali elementi kao što su akteri i tijek događaja nisu identični.
U svim ostalim slučajevima, slučaj uporabe je novi (UC_N).

3.5. Studijski primjer primjene procesa usporedbe novog zahtjeva s postojećim slučajevima uporabe u industrijskim projektima

U studijskom primjeru organizacije opisanom u Poglavlju 2 unaprijed je planirana višestruka iskoristivost artefakata u više projekata što je jedan od preduvjeta sustavnog iskorištavanja proizvoda. S obzirom da se radilo o komercijalnom programu razvoja sličnih programskih proizvoda, trebao se uskladiti proces njihovog razvoja. Kao što je ranije spomenuto, nisu se svi artefakti mogli razviti na način da ih je moguće višestruko iskorištavati jer bi takav pristup značajno povećao trošak razvoja prvog projekta u sklopu kojeg se razvijao novi programski proizvod. S druge strane, u fazi planiranja programa nemoguće je bilo točno odrediti broj projekata koji će se provesti, a samim time i podnošljivo povećanje troška u prvom projektu. Stoga je u promatranom programu donesena odluka da će se artefakti iskorištavati na oportunistički način

(doslovno prema potrebi), međutim kontrolirano i praćeno u sklopu projekta i programa. Takav pristup bi se mogao okarakterizirati kao hibridni pristup, a zanimljivo je da su istraživanja pokazala da se takav pristup već koristi u industriji te da ima svoje prednosti [44]:

- neiskorišten potencijal postojećeg koda stavlja se u uporabu, a time se izravno povećava produktivnost programera jer nisu ograničeni pravilima koja bi uvjetovala višestruko iskorištavanje,
- smanjuje se ukupni trošak razvoja novog programskog proizvoda.

Način kontrole razvoja artefakata prije izvođenja prvog projekta postavljen je na sljedeći način:

- u fazi planiranja inicijalnog projekta u sklopu kojeg se razvijao novi programski proizvod (i artefakti koji se namjeravaju višestruko koristiti) definirane su norme za oblikovanje projektne dokumentacije poput projektnog plana, opisa obima posla i arhitekture, testne specifikacije, itd.,
- definirane su uloge projektnih članova odgovorne za spomenute dokumente isporuke,
- definirana su pravila o obaveznom detaljnom ažuriranju svih dokumenata isporuke (uključujući i razloge za promjenama unutar dokumenata, eventualnim posljedicama i sl.).

Projektni tim je u ranoj fazi razvoja htio spriječiti da stečena znanja projektnog tima ostanu „u glavama“ pojedinaca koji su dio projekta. Bilo je bitno ažurno i kvalitetno održavati projektnu dokumentaciju koja će ostati dostupna organizaciji koja upravlja projektnim timom kako bi se znanje moglo podijeliti i s novim članovima organizacije koji pristupaju projektu (npr. u kasnijoj fazi) ili za slične projekte u budućnosti.
Primjenom nove klasifikacije slučajeva uporabe prema kriteriju iskoristivosti, unutar promatranog programa praksa je pokazala da je kod projekata koji imaju veliki broj slučajeva uporabe koji su slični (S) ili identični (R) postojećim slučajevima uporabe, moguće dijelom iskoristiti mnoge postojeće projektne artefakte iz prethodnih projekata i u novom projektu:

- tehničku dokumentaciju (specifikacija poslovnih procesa, katalog poslovnih pravila, komponentni model, testne specifikacije, itd.),
- programski kod.

4. Model procjene količine posla za razvoj programskih rješenja zasnovan na višestrukom korištenju slučajeva uporabe i značajkama projektnog tima

U studijskom primjeru primjene postojećih metoda procjene količine posla (Poglavlje 2), uočena su znatna odstupanja stvarne od procijenjene količine posla primjenom modela UCP i ekspertne procjene u projektima koji višestruko koriste ranije razvijene artefakte.

Model procjene UCP omogućuje procjenu količine posla za pojedini projekt, temeljenu na obimu posla, funkcijskim i nefunkcijskim zahtjevima proizvoda. Takav pristup pokazao se prikladan samo za procjenu napora inicijalnog projekta unutar programa gdje su svi artefakti projekta izgrađeni otpočetka. Za slijedne projekte, tim nije mogao postaviti bilo kakav ulazni podatak u model procjene koji bi opisao da su određeni artefakti rješenja ponovno korišteni iz prethodnih projekata.

Radi pružanja preciznije procjene količine posla za projekte u kojima se artefakti ponovno koriste, predlaže se definiranje novog modela procjene količine posla koji se temelji na iskoristivosti slučajeva uporabe – nazvan *Use Case Reusability* (UCR). Model UCP pokazao se učinkovitim za procjenu količine posla inicijalnih projekata te će se odrediti kao polazišna točka za definiciju novog modela UCR. Ulazni faktori koje je Karner odredio u modelu UCP će se ponovno preispitati i potencijalno uključiti u model UCR. U sklopu modela će se također definirati novi ulazni faktori koji opisuju aspekt višestruke iskoristivosti. Nakon utvrđivanja ulaznih faktora i njihovih povezanih atributa (ljestvica svakog ulaznog faktora), postavit će se matematički model koji uključuje algoritam procjene količine posla. Posljednji korak u procesu definiranja modela UCR bit će postavljanje težina svakog atributa ulaznih faktora.

U predloženom modelu UCR slučajevi uporabe će poslužiti kao ulazni parametri za usporedbu sličnosti i kompatibilnosti opsega posla između inicijalnog i slijednih projekata. Usporedba slučajeva uporabe među projektima karakterizira mjeru iskoristivosti artefakata, što ne podrazumijeva samo uporabu programskog koda, već potencijalno i druge projektne artefakte kao što su model dizajna, model podataka, odluke o arhitekturi sustava i testne specifikacije. Razina

iskoristivosti drugih artefakata povećava se s brojem jednakih ili sličnih slučajeva uporabe. Slučajevi uporabe se definiraju u ranoj fazi razvojnog projekta i stoga se ovaj model procjene može koristiti već u fazi planiranja projekta.

Nakon završetka nekoliko slijednih projekata u spomenutoj studiji slučaja, analizom stvarne količine posla po projektnim fazama nametnula su se tri glavna aspekta koja su utjecala na manju stvarnu količinu posla u slijednim projektima u odnosu na stvarnu količinu posla inicijalnog projekta: i) funkcionalni obim posla projekta, ii) tehnička složenost razvijenog rješenja i iii) faktori okoline. Karner je temeljio model UCP na vrlo sličnim aspektima: obim posla projekta izražen neprilagođenom količinom slučajeva uporabe (UUCP), tehnička složenost definirana tehničkim faktorom (TCF) i karakteristike projektnog tima izražene faktorom okoline (EF). Stoga su u sljedećem koraku procijenjeni svih faktori koji čine izračun UUCP, TCF i EF prema Karneru radi donošenja odluke hoće li isti faktori biti uključeni u novi model UCR. Radi objektivnosti procjene, u analizu faktora je uključen ekspertni tim koji se sastojao od tri voditelja projekta i pet arhitekata. Svi oni su imali ranije iskustvo u procesu procjene količine posla razvojnog projekta. Voditelji projekata bili su iskusniji u svojim ulogama i svi su imali certifikat *Project Management Professional* (PMP). Svaki arhitekt je imao više od pet godina iskustva u trenutnoj ulozi, a trojica od njih su certificirani IBM arhitekti. Dvojica projektnih menadžera i dva arhitekata u prošlosti su radili na projektima razvoja softvera koji su ponovno koristili artefakte iz prethodnih projekata koji uključuju ponovnu uporabu projektnih dokumenata i izvornog koda.

Svi članovi stručnog tima imali su mogućnost navesti dodatne čimbenike koji po njihovom mišljenju utječu na količinu posla povrh faktora koje je odredio Karner. Svi novo predloženi faktori koji su utjecali na količinu posla slijednih projekata (ili povećali ili smanjili količinu posla) raspravljeni su između stručnjaka i svi oni pripadali su među tri već poznata aspekta koje je i uveo Karner: funkcionalni obim posla, tehnička složenost i faktori okoline.

Funkcionalni obim posla

Funkcionalni obim posla odnosi se na usluge koje nudi određeni sustav [29]. Karner ga je odlučio opisati u svom modelu UCP pomoću slučajeva uporabe, a isti pristup slijedi se u modelu procjene količine posla UCR.

U novom modelu procjene zadržala bi se postojeća klasifikacija aktera iz modela UCP. Slučajevi uporabe bi se klasificirali prema kompleksnosti i iskoristivosti.

Kompleksnost slučajeva uporabe obilježava se na sličan način kao i u metodi UCP; ovisno o broju transakcija unutar slučaja uporabe dijele se na jednostavne, prosječne, složene i kritične. Transakcija označava jedan događaj (aktivnost) između aktera i sustava. Kriteriji za određivanje kompleksnosti slučaja uporabe s obzirom na broj transakcija:

- **jednostavni slučaj uporabe** (UC_L): 3 ili manje transakcija,
- **prosječni slučaj uporabe** (UC_M): od 4 do uključivo 7 transakcija,
- **složeni slučaj uporabe** (UC_H): od 8 do uključivo 15 transakcija,
- **kritični slučaj uporabe** (UC_C): 16 ili više transakcija.

Dodatan tip slučaja uporabe, **kritični slučaj uporabe** za one s više od 15 transakcija – uveden je zajedničkim dogovorom stručnog tima; takva klasifikacija uvedena je i u nekim drugim metodama procjene temeljenima na slučajevima uporabe [19] te evaluacije pokazuju precizniju procjenu količine posla proširenjem klasifikacije dodatnim tipom.

UCR se razlikuje uključivanjem dodatne klasifikacije slučajeva upotrebe za slijedne projekte temeljene na njihovoj iskoristivosti:

- **novi slučaj uporabe** UC_N – predstavlja potpuno novi slučaj uporabe koji ne predstavlja bilo koji dio prethodno definiranih slučajeva uporabe,
- **sličan slučaj uporabe** UC_S – predstavlja slučaj uporabe sličan već postojećem slučaju uporabe iz prethodnog projekta,
- **identičan slučaj uporabe** UC_R – predstavlja identičan slučaj uporabe već ranije definiran u prethodnom projektu.

U inicijalnom projektu gdje su svi artefakti projekta izgrađeni otpočetka, svi slučajevi uporabe definirani su kao novi slučajevi uporabe. Svaki od sljedećih projekata ima svoj obim posla, a svaki novi zahtjev mora se analizirati kako bi se utvrdilo odgovara li elementima već postojećeg slučaja upotrebe iz prethodnog projekta. Proces analize zahtjeva opisan je u Poglavlju 3 (3.3. Proces usporedbe novog zahtjeva s postojećim slučajima uporabe).

Tehnička složenost

Karner je definirao trinaest tehničkih faktora za opisivanje ukupne tehničke složenosti softverskog proizvoda (tablica 1-1). Znanstvene studije ukazuju da je utjecaj tehničkih faktora (TCF) u modelu UCP manji i da ti faktori ne pokrivaju sve nefunkcijske zahtjeve proizvoda [34]. Neke studije čak preporučuju da se TCF ne računa u modelu UCP (postavljanje vrijednosti na 1 u jednadžbi 1.4) [17] [51] [52]. Stoga su u sljedećem koraku definirani odgovarajući tehnički faktori za novi model procjene količine posla UCR koji bi imali utjecaja na izračun količine posla za inicijalni i slijedne projekte.

Zadatak stručnog tima je bio procijeniti koji od postojećih tehničkih faktora koje je definirao Karner ima značajan utjecaj na količinu posla. Svaki od eksperata je ocijenio važnost Karnerovih tehničkih faktora dodijelivši tri boda za iznimnu važnost, dva boda za umjerenu važnosti ili jedan bod za neznatnu važnosti. Prije ocjenjivanja, usuglašeno je među stručnjacima da novi model UCR sadrži samo one faktore čiji je rezultat najmanje 80% maksimalnog mogućeg ukupnog rezultata.

S obzirom na broj članova ekspertne skupine, maksimalni broj ocjena za svaki od faktora iznosio je 24. Ukupne ocjene dodijeljene za svaki od faktora prikazane su u Tablica 4-1.

Tablica 4-1 Procjena važnosti tehničkih faktora modela UCP za novi model procjene UCR

Faktor	Opis	Ukupna ocjena
T1	Distribuirani sustavi	22
T2	Vrijeme odaziva	13
T3	Efikasnost krajnjeg korisnika	12
T4	Složenost procesiranja	16
T5	Iskoristivost koda	24
T6	Jednostavna instalacija	4
T7	Jednostavno korištenje	12
T8	Prenosivost	21
T9	Lako mijenjanje	13
T10	Konkurentni procesi	14
T11	Sigurnosni zahtjevi	11

T12	Pristup trećim stranama	10
T13	Posebna obuka krajnjeg korisnika	11

Promatrajući ukupne ocjene faktora u Tablici 4-1 dva faktora opisuju značajke proizvoda koje imaju bitan utjecaj na količinu posla za razvoj proizvoda te bi isti bili uključeni u novi model procjene:

- distribuirani sustavi (T1, opis: arhitektura rješenja može biti centralizirana ili može biti distribuirana),
- podržavanje više platformi (T8, opis: mora li programsko rješenje podržati više platformi, npr. web, mobilne aplikacije, itd.).

Značajka iskoristivosti koda, koja je pokrivena faktorom T5, obavezna je kod promatranog programskog proizvoda; međutim, ovaj faktor se neće uključiti u novi model jer je ta značajka nužna (nosila bi najveću vrijednost) ne samo zbog eventualnog zahtjeva proizvoda, već zbog namjene (karakteristike) ciljanih projekata. Na isti način, model UCR ne uključuje čimbenike poput fleksibilnosti i modularnosti koji opisuju generalne aspekte iskoristivosti koji se moraju slijediti u projektima koji planiraju ponovno upotrijebiti artefakte.
Ostale tehničke faktore (T2-T7, T9-13), za koje se ne smatra da opisuju aktivnosti koje bi imale značajan utjecaj na količinu posla, novi model procjene neće uključivati.

Stručni tim dogovorio je preimenovanje faktora T1 distribuirani sustavi u faktor imena **arhitektura softvera** jer će se time obuhvatiti i šire karakteristike poput distribuiranosti sustava, slojevitosti arhitekture, složenosti infrastrukture, sigurnosnih zahtjeva i nefunkcijskih zahtjeva.

Prema prethodnom iskustvu, stručni tim ocijenio je da sljedeći čimbenik ima značajan utjecaj na količinu posla slijednih projekata (može se svrstati među tehničke faktore): **integracija s novim sustavima**. Spomenuti faktor opisuje integrira li se programski proizvod slijednog projekta s većim brojem novih sustava (u odnosu na proizvod inicijalnog projekta).

Faktori okoline

Faktori okoline prema Karneru (pobrojani u tablici 1-2) prvenstveno se odnose na kompetenciju članova projektnog tima (F1 – F5). Analiza stvarne količine posla projekata u studijskom primjeru pokazala je da iskustvo članova tima obrnuto proporcionalno utječe na količinu posla za slijedne projekte. Vrijednosti faktora okoline za slijedne projekte rastu (odnosno *EF* pada) jer je za očekivati da će isti tim ljudi koji radi na vrlo sličnim projektima prvenstveno imati veće iskustvo u poznavanju aplikacije, bolje poznavanje objektno-orijentirane arhitekture i općenito veće iskustvo tima.

Stručni tim ocijenio je faktore okoline modela UCP na isti način kao i tehničke faktore istog modela: svaki član tima ocijenio je faktore kao iznimno važnog (dodijelivši 3 boda) ili umjereno važnog (dodijelivši 2 boda) ili malo važnog (dodijelivši 1 bod). Niže u tablici su ukupne ocjene za svakog od faktora. Odlučeno je da će novi model sadržavati faktore čija ocjena je minimalno 80% maksimalno moguće ukupne ocjene; maksimalna moguća ukupna ocjena za svaki faktor iznosi 24, stoga će u novi model biti uključeni faktori s ocjenom većom od 19 bodova.

Tablica 4-2 Procjena važnosti faktora okoline modela UCP za novi model procjene UCR

Faktor	Opis	Ukupna ocjena
F1	Poznavanje procesa RUP	14
F2	Iskustvo u poznavanju aplikacije	24
F3	Poznavanje objektno-orijentirane arhitekture	14
F4	Iskustvo tima (analitičara)	23
F5	Motivacija	16
F6	Stabilnost zahtjeva	20
F7	Skraćeno radno vrijeme tima	10
F8	Kompleksnost programskog jezika	10

Promatrajući ukupne ocjene faktora u tablici 4-2, tri faktora opisuju značajke proizvoda koje imaju bitan utjecaj na količinu posla za razvoj proizvoda te će se isti uključuje u novi model procjene:

- iskustvo u poznavanju aplikacije (F2, opis: mjera iskustva članova tima potrebnog za modifikaciju koda),
- iskustvo članova tima (F4, opis: mjera iskustva prvenstveno glavnog arhitekta i glavnog programera),
- stabilnost zahtjeva (F6, opis: mjera stabilnosti postojećih zahtjeva u sklopu koje nema čestih izmjena postojećih ili uvođenja novih zahtjeva).

Stručni tim dogovorio je preimenovanje faktora F6 iz „stabilnost zahtjeva“ u širi pojam „zrelost zahtjeva“ kojim bi se obuhvatila i kvaliteta postavljenih zahtjeva (potrebni detalji poslovnog procesa i poslovna pravila).

Prema prethodnom iskustvu, stručni tim predložio je uvođenje sljedećih čimbenika koji opisuju karakteristike projektnog tima:

- lociranost tima – nalaze li se svi članovi na istoj lokaciji ili je tim distribuiran (virtualan),
- veličina tima – veći broj članova tima iziskuje više aktivnosti integracije projektnih zadataka (što se može i uvrstiti u dodatan, *overhead* napor),
- održavanje projektne dokumentacije - dostupnost kvalitetne dokumentacije koja opisuje artefakte inicijalnog projekta (artefakte koji će se iskorištavati),
- kohezija tima – opisuje jesu li članovi tima i ostale strane (*stakeholders*) surađivali u ranijim projektima.

Svi ulazni faktori modela UCR model navedeni su u tablicama 4-3 i 4-4. Svakom od tehničkih faktora i faktora okoline dodjeljuje se deskriptivna skala atributa (slabo, umjereno, jako) sa svojim skupom kriterija za svaki atribut. U modelu UCP procjenitelj mora dodijeliti numeričku vrijednost od 0 do 5 koja opisuje značenje svakog tehničkog faktora ili faktora okoline, ali bez jasnih smjernica koja jasno opisuju razliku između susjednih vrijednosti. Postavljanjem jasnih kriterija za odabir svakog atributa, ulazni parametri modela procjene objektivni su i transparentni.

Tablica 4-3 Opis tehničkih faktora i dodijeljenih vrijednosti

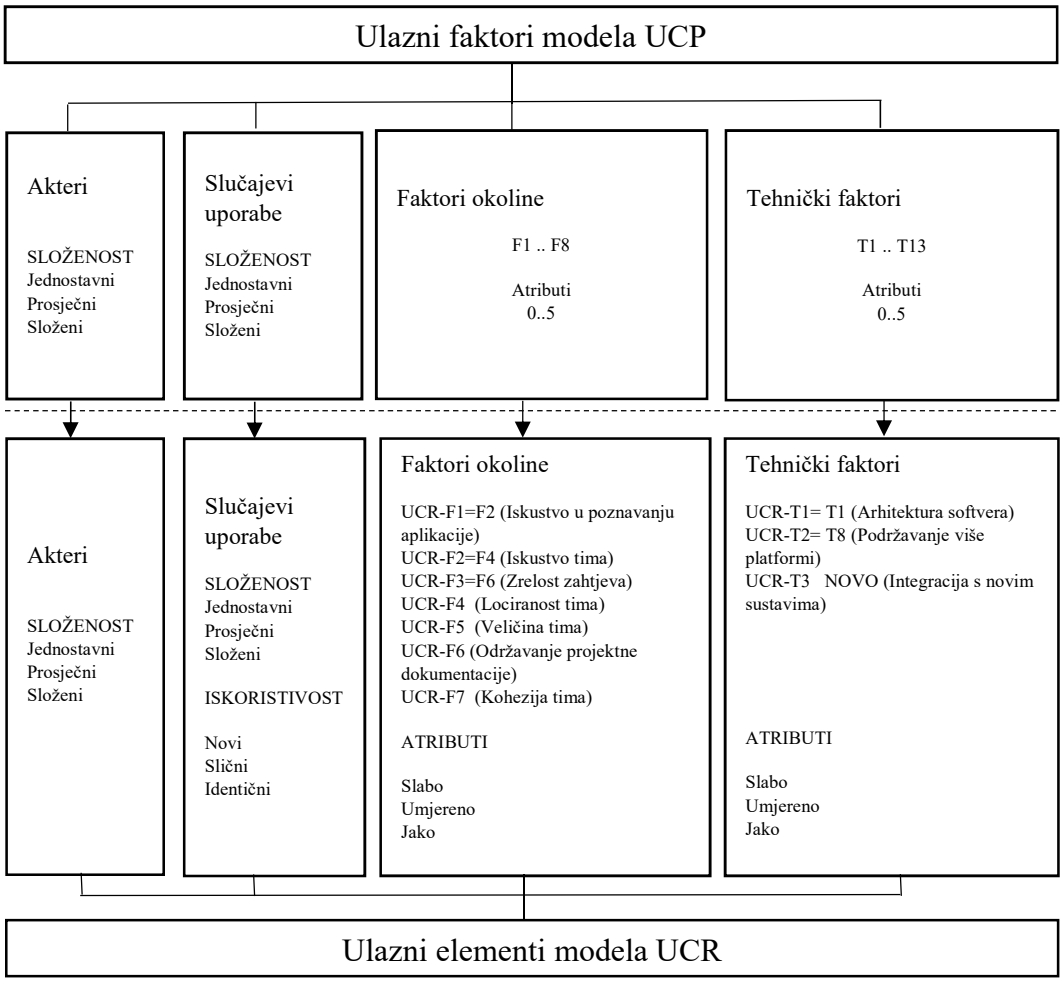
Oznaka	Tehnički faktori	Opis	Ljestvica ocjena		
			Slabo (*Low* - L)	**Umjereno (*Medium* - M)**	**Jako (*High* - H)**
UCR-T1	Arhitektura softvera	Distribuiranost sustava, *multi-tenan*t arhitektura, složenost infrastrukture i nefunkcijskih zahtjeva, sigurnosni zahtjevi (npr. regulativne smjernice)	Centralizirani sustav, jednostavna infrastruktura, mali broj nefunkcijskih zahtjeva	Centralizirani sustav, umjereno složena infrastruktura, umjeren broj nefunkcijskih zahtjeva	Distribuirani sustavi, složena infrastruktura, velik broj nefunkcijskih zahtjeva, poštivanje odrednica regulativnih tijela (npr. sigurnost)
UCR-T2	Podržavanje više platformi	Podržavanje više platformi istovremeno: web, mobilne aplikacije, razni operativni sustavi, itd.	Podržavanje samo jedne platforme	Podržavanje dvije platforme	Podržavanje tri ili više platformi
UCR-T3	Integracija s novim sustavima	Povezivanje s većim brojem drugih programskih rješenja	Nema povezivanja s drugim sustavima	Povezivanje s jednim do dva programska rješenja	Povezivanje s tri ili više programska rješenja

Tablica 4-4 Opis faktora okoline i dodijeljenih vrijednosti

Oznaka	Faktori okoline	Opis	Ljestvica ocjena		
			Slabo (*Low* - L)	**Umjereno (*Medium* - M)**	**Jako (*High* - H)**
UCR-F1	Iskustvo u poznavanju aplikacije	Poznavanje aplikacije od strane glavnog arhitekta i glavnog programera	Glavni arhitekt i glavni programer ne poznaju poslovne procese aplikacije kao ni značajke rješenja (arhitekturu, zahtjeve, itd.)	Glavni arhitekt ili glavni programer poznaje poslovne procese aplikacije i značajke rješenja (arhitekturu, zahtjeve, itd.)	Glavni arhitekt i glavni programer dobro poznaju poslovne procese aplikacije kao i značajke rješenja (arhitekturu, zahtjeve, itd.)
UCR-F2	Iskustvo tima	Prijašnje iskustvo glavnog arhitekta i glavnog programera	Glavni arhitekt ili glavni programer ima manje od 3 godine iskustva u toj roli; slabo poznaju poslovne procese (industriju) klijenta	Glavni arhitekt ili glavni programer ima više od 3 godine iskustva u toj roli; umjereno poznaju poslovne procese (industriju) klijenta	Glavni arhitekt i glavni programer imaju više od 3 godine iskustva u toj roli; poznaju poslovne procese (industriju) klijenta
UCR-F3	Zrelost zahtjeva	Kvaliteta i stabilnost postavljenih zahtjeva (potrebni detalji poslovnog procesa, poslovna pravila, itd.)	Zahtjevi se često mijenjaju, ne opisuju detalje poslovnih procesa, potreban reinženjering poslovnih procesa	Zahtjevi se rijetko mijenjaju, ne uključuju detalje poslovnih procesa, nije potreban reinženjering poslovnih procesa	Zahtjevi se ne mijenjaju, uključuju detalje poslovnih procesa, kvalitetno su komunicirani, nije potreban reinženjering poslovnih procesa
UCR-F4	Lociranost tima	Fizički razmještaj članova tima	Članovi tima su na različitim lokacijama i u različitim vremenskim zonama	Članovi tima su na različitim lokacijama i u istoj vremenskoj zoni	Članovi tima su smješteni na istoj lokaciji
UCR-F5	Veličina tima	Broj članova u projektnom timu	Iznad 30 članova tima	Između 15 i 30 članova tima	Do 15 članova tima

UCR-F6	Održavanje projektne dokumentacije	Definicija procesa dokumentiranja projektnih zadataka i artefakata	Ne postoji standardizirani proces za održavanje projektne dokumentacije	Definiran je standardni proces za održavanje projektne dokumentacije, ali nije sveobuhvatan ili se ne prati u cijelosti	Definiran je standardni proces za održavanje projektne dokumentacije koji obuhvaća sve projektne aktivnosti i artefakti i prati se u cijelosti
UCR-F7	Kohezija tima	Stupanj prethodne suradnje članova tima	Članovi projektnog tima nisu ranije surađivali na projektu sličnog obima posla	Članovi projektnog tima surađivali su na projektu sličnog ili istog obima posla	Članovi projektnog tima surađivali su na više od dva projekta sličnog ili istog obima posla

Slika 4-1 prikazuje ulazne faktore modela UCP i UCR njihovim atributima. Slika prikazuje odnos faktora između dvaju modela procjene i koji od faktora u modelu UCR su naslijeđeni iz modela UCP.



Slika 4-1 Usporedba modela UCP i UCR

4.1. Matematički model

Neprilagođena težina aktera (*Unadjusted actor weights* - UAW) računa se kao suma dodijeljene težine svakog aktera klasificiranog prema složenosti (jednadžba 4.1):

$$UAW_R = \sum_{i=1}^{n} A_i(kompleksnost) \tag{4.1}$$

Za i = 1, 2, ..., n (n predstavlja ukupnu količinu aktera)

Kompleksnost slučajeva uporabe obilježava se na sličan način kao i u metodi UCP; ovisno o broju transakcija unutar slučaja uporabe dijele se na **jednostavne** (*UC_L*), **prosječne** (*UC_M*), **složene** (*UC_H*) i **kritične** (*UC_C*).

Ključna razlika u klasifikaciji slučajeva uporabe u odnosu na originalnu metodu UCP leži u dodatnoj dimenziji vrste slučajeva uporabe u odnosu na iskoristivost; definiraju se **novi slučaj uporabe** (*UC_N*), **sličan slučaj uporabe** (*UC_S*) i **identičan slučaj uporabe** (*UC_R*).

S obzirom da se svaki slučaj uporabe klasificira u odnosu na kompleksnost i iskoristivost, **neprilagođena težina slučajeva uporabe** (*Unadjusted use case weights – $UUCW_R$*) računa se kao suma produkata dodijeljenih težina slučaja uporabe za svaki tip slučaja uporabe:

$$UUCW_R = \sum_{i=1}^{n} UC_i\,(kompleksnost) \times UC_i(iskoristivost) \tag{4.2}$$

Za i = 1, 2, ..., n (n predstavlja ukupnu količinu slučajeva uporabe)

Neprilagođena količina slučajeva uporabe (*Unadjusted Use Case Point – $UUCP_R$*) računa se kao suma neprilagođenih težina aktera i slučajeva uporabe:

$$UUCP_{\mathrm{R}} = UAW_R + UUCW_{\mathrm{R}} \tag{4.3}$$

Faktor tehničke kompleksnosti (*Technical Complexity Factor – TCF_R*) računa se kao produkt težina svih tehničkih faktora (od *UCR_T1* do *UCR_T3*):

$$TCF_{\mathrm{R}} = \prod_{i=1}^{3} UCR\_Ti \tag{4.4}$$

Za i = 1, 2, 3

Faktor okoline (*Environmental Factor* – $\mathrm{EF_R}$) računa se kao produkt težina svih faktora okoline (od *UCR_F1* do *UCR_F7*):

$$EF_{\mathrm{R}} = \prod_{i=1}^{7} UCR\_Fi \tag{4.5}$$

Za i = 1, 2, 3, ..., 7.

Prilagođena količina *UCR* računa se na sljedeći način:

$$UCR = UUCP_{\mathrm{R}} * TCF_{\mathrm{R}} * EF_{\mathrm{R}} \tag{4.6}$$

Naredni korak je definicija vrijednosti čovjek – sati po UCR. Karner je u svom opusu predložio 20 čovjek – sati po UCP [36], dok će se vrijednost UCR definirati u narednim poglavljima kalibracijom koristeći stvarnu količinu posla projekata u studiji slučaja.

$$UCR[\check{c}ovjek - sat] = \frac{Stvarna\ količina\ posla\ [\check{c}ovjek - sat]}{UCR} \tag{4.7}$$

4.2. Povijesni podaci studijskog primjera

Povijesni podaci projekata tri različita programa (Program A, Program B i Program C) prikupljeni su radi (1) kalibracije mjere UCR izražene u čovjek – sat za UCR model i (2) evaluacije modela UCR. Povijesni podaci projekata uključivali su ulazne atribute za matematički model UCR, količinu posla procijenjenu modelom UCP i stvarnu količinu posla za svaki od projekata.

Dva najčešća načina prikupljanja podataka su putem upitnika i intervjua [53]. Upitnik se najčešće šalje u papirnatom ili elektroničkom obliku, zajedno s instrukcijama kako ispuniti sam upitnik. Odgovarajuća osoba ispunjava upitnik te ga šalje nazad istraživaču. Intervjuom su prikupljeni povijesni podaci Programa B i C – time su pitanja o obimu posla, slučajevima uporabe, akterima, tehničkim faktorima, faktorima okoline te specifičnostima projekta mogla biti dodatno pojašnjena prema potrebni za vrijeme intervjua. S obzirom da je ista osoba intervjuirala članove oba programa, osigurao se konzistentni pristup u prikupljanju podataka. Upitnici u sklopu intervjua često imaju veću razinu odaziva nego oni poslani putem elektronske pošte [56].

Povijesni podaci o projektima unutar Programa A prikupljeni su iz dostupne dokumentacije programa. Dokumentacija je sadržavala specifikacije slučajeva uporabe i aktera, pripadne transakcije, model i katalog slučajeva uporabe prema kojem se jednostavno mogla odrediti njihova klasifikacija prema složenosti i iskoristivosti. Svi članovi tima unosili su u zajedničku bazu podataka stvarnu količinu posla za svaku od faza unutar svakog projekta. Voditelji projekata osigurali su poštivanje pravila za točan unos stvarne količine posla, kako bi se mogla precizno pratiti i uspoređivati količina posla između projekata. Istraživač je sam sudjelovao u programu A, znao je karakteristike proizvoda i projektnog tima te je mogao odrediti tehničke faktore i faktore

okoline za svaki projekt. Za dodatna pojašnjenja, kontaktirao je ostale voditelje projekata unutar programa (slično formi intervjua) kako bi osigurao točno razumijevanje skupljenih podataka.

Svaki program imao je različit obim posla i različite projektne timove (resursi nisu bili dijeljeni ni u jednom od programa). Projekti unutar programa A bili su industrijski (komercijalni) projekti u kojima su razvijene softverske aplikacije (web / desktop) za integraciju različitih aplikacija. Svaki od projekata imao je 10 do 12 slučajeva uporabe s 4 do 6 članova projekta.

Projekti u Programu B bili su komercijalni projekti u kojima su razvijene softverske aplikacije za klijente unutar javnog sektora. Svaki od projekata imao je do 22 slučaja uporabe s 8 do 12 članova projekta.

Oba programa A i B slijedila su načela metodologije razvoja RUP-a tako da su akteri i slučajevi uporabe bili definirani u projektnim dokumentima.

Program C predstavljao je 4 projekta u kojima su razvijene softverske aplikacije za osobe s kompleksnim komunikacijskim potrebama, pod platformom ICT-AAC (ICT Kompetentna mreža za inovativne usluge za osobe s kompleksnim potrebama komunikacije) sufinancirano od strane EU, UNICEF i Fakulteta elektrotehnike i računarstva [54] [55]. Svaki od projekata imao je od 15 do 22 slučajeva uporabe s 4 do 5 članova projekta. Projektni tim slijedio je agilnu razvojnu metodologiju. Slučajevi uporabe i akteri dokumentirani su i klasificirani na temelju funkcionalnosti aplikacije anketiranjem članova projektnog tima.

Voditelji projekata svih programa pružili su informacije o atributima tehničkih faktora i faktora okoline te stvarne količine posla, iskazane u mjeri čovjek – sat za svaki projekt.

Projekti i njihovi povijesni podaci podijeljeni su u dva odvojena skupa: skup za kalibraciju i skup za validaciju. Ukupno 18 projekata iz programa A čine kalibracijski skup, dok 11 preostalih projekata iz programa A, B i C su dio skupa za validaciju. Projekti su bili podijeljeni u spomenute skupove s obzirom na vremenski tijek njihova završetka. Svi projekti u kalibracijskom skupu su završeni prije projekata u validacijskom skupu. U oba skupa zastupljeni su "inicijalni" i "slijedni" tip projekata. Karakteristike svakog od programa unutar kalibracijskog i validacijskog skupa navedene su u Tablica 4-5.

Tablica 4-5 Opis povijesnih podataka unutar skupova za kalibraciju i validaciju

Skup podataka za kalibraciju						
Program	**Količina inicijalnih projekata**	**Količina slijednih projekata**	**Vrsta projekta**	**Obim posla**	**Veličina obima posla (#UC)**	**Veličina projekta (broj članova projektnog tima)**
Program A	2	16	Industrijski projekti	Integracija aplikacija i automatizacija procesa	Od 10 do 12	Od 4 do 6
Skup podataka za validaciju						
Program	**Količina inicijalnih projekata**	**Količina slijednih projekata**	**Vrsta projekta**	**Obim posla**	**Veličina obima posla (#UC)**	**Veličina projekta (broj članova projektnog tima)**
Program A	0	5	Industrijski projekti	Integracija aplikacija i automatizacija procesa	Od 10 do 12	Od 4 to 6
Program B	1	1	Industrijski projekti	Razvoj aplikacije za javni sektor	Od 20 do 22	Od 8 do 12
Program C	2	2	Akademska zajednica	Razvoj aplikacije za osobe sa složenim komunikacijskim potrebama	Od 15 do 22	Od 4 do 5

4.3. Određivanje parametara modela UCR

Nakon definicije matematičkog modela, slijedi postavljanje težina faktora (parametara) za sve atribute u modelu UCR ovisno o njihovom utjecaju na ukupnu količinu posla. Težine (ponderi) faktora koji su naslijeđeni iz modela UCP kao što su akteri i složenost slučajeva uporabe isti su kao i u modelu UCP. Težine novih faktora u modelu UCR modelu određeni su metodom Delphi koja je provedena među pet stručnjaka za procjenu, a koji su sudjelovali u definiciji tehničkih faktora i faktora okoline.

U metodi Delphi skupina stručnjaka vodi se do usuglašenog mišljenja o promatranom pitanju. Tehnika je primjenjiva kod ispitivanja više osoba i pritom se želi izbjeći dominantan utjecaj jedne osobe koji je često prisutan pri zajedničkom odlučivanju. U prvom (preliminarnom) krugu sudionici daju procjenu o pojedinom problemu individualno (najčešće putem upitnika) i bez konzultacija s ostalim sudionicima. Rezultati procjena sakupljaju se, obrađuju i distribuiraju svim sudionicima nakon čega su sudionici u drugom krugu zamoljeni da ponovo procjene odgovor na isto postavljeno pitanje (ovog puta sa saznanjem o odgovorima drugih sudionika). Drugi krug obično rezultira u sužavanju raspona procijenjenih vrijednosti te se dolazi do zajedničke odluke (koncensusa).

U sklopu ove studije, tehnikom Delphi postavljaju se razumne početne vrijednosti za faktore modela UCR. Dobivene vrijednosti će kasnije poslužiti kao polazište za kalibraciju modela UCR.

Stručnjaci su za faktore okoline, tehničke faktore te nove slučajeve uporabe (UC_N, UC_S, UC_R) procijenili mjeru utjecaja na količinu posla na način da su procijenili vrijednost raspona produktivnosti (PR). Vrijednost raspona produktivnosti označava najveći raspon utjecaja koji bi pojedini faktor mogao imati na količinu posla razvoja programskog proizvoda za cijeli raspon zadanih vrijednosti (za raspon zadanih ljestvica ocjena – od slabo do jako). Konačne vrijednosti raspona produktivnosti postavljene metodom odlučivanja Delphi poslužit će za postavljanje inicijalnog skupa težina za svaku od vrijednosti faktora.

Primjerice, vrijednost raspona produktivnost 100% označava da se za pojedini faktor količina posla može povećati 100% promijeni li se vrijednost faktora od slabo prema jako. Za takav primjer sudionik daje vrijednost 2. S druge strane, za neki drugi faktor, raspon produktivnosti može biti

75% te to označava da se količina posla može povećati 75% promijeni li se vrijednost faktora od slabo prema jako – za takav primjer sudionik daje vrijednost 1.75.

Tehničkim faktorima i faktorima okoline dodijeljena su tri moguća atributa (slab, umjeren, jak). Proces odlučivanja provodi se u dva kruga testiranja. Proces odlučivanja proveden je u dva kruga ispitivanja; rezultati eksperimenta odlučivanja metodom Delphi prikazani su u tablici 4-6 (prvi krug ispitivanja) i tablici 4-7 (drugi krug ispitivanja). U istraživanju je sudjelovalo pet ispitanika u rolama voditelja projekta i arhitekta (S1, S2, S3, S4 i S5). Omjeri atributa nužni su radi određivanja numeričke vrijednosti svakog atributa. Vrijednosti za svaki od tri atributa definirane su u linearno proporcionalnom omjeru. Početne težine računaju se na temelju pravila opisanih u sljedećem odlomku.

Tablica 4-6 Analiza rezultata prvog kruga odlučivanja metodom Delphi

Oznaka	Tehnički faktori	S1	S2	S3	S4	S5	Srednja vrijednost raspona produkt.	Medijan raspona produkt.	Mod vrijednost raspona produkt.	Raspon vrijednosti
		Raspon produktivnosti								
UCR-T1	Kompleksnost arhitekture	1.55	1.85	1.45	1.5	1.95	1.66	1.55	#N/A	1.45 - 1.95
UCR-T2	Podržavanje više platformi	1.35	1.65	1.4	1.4	1.6	1.48	1.4	1.4	1.35 - 1.65
UCR-T3	Integracija s novim sustavima	1.3	1.5	1.4	1.1	1.3	1.32	1.3	1.3	1.1 - 1.5

Oznaka	Faktori okoline	S1	S2	S3	S4	S5	Srednja vrijednost raspona produkt.	Medijan raspona produkt.	Mod vrijednost raspona produkt.	Raspon vrijednosti
		Raspon produktivnosti								
UCR-F1	Iskustvo u poznavanju aplikacije	1.5	1.25	1.4	1.3	1.1	1.31	1.3	#N/A	1.1 - 1.5
UCR-F2	Iskustvo tima	1.3	1.2	1.25	1.2	1.3	1.25	1.25	1.3	1.2 - 1.3
UCR-F3	Zrelost zahtjeva	1.2	1.3	1.1	1.15	1.2	1.19	1.2	1.2	1.1 - 1.3
UCR-F4	Lociranost tima	1.25	1.05	1	1.15	1.05	1.1	1.05	1.05	1 - 1.25
UCR-F5	Veličina tima	1.1	1.1	1.2	1.1	1.05	1.11	1.1	1.1	1.05 - 1.2
UCR-F6	Održavanje projektne dokumentacije	1.05	1.15	1.3	1.1	1.25	1.17	1.15	#N/A	1.05 - 1.3
UCR-F7	Kohezija tima	1.1	1.15	1.05	1	1.05	1.07	1.05	1.05	1 - 1.15

Oznaka	Slučaj uporabe (smanjenje posla u odnosu na novi slučaj uporabe)	S1	S2	S3	S4	S5		Srednja vrijednost	Medijan	Mod vrijednost	Raspon vrijednosti
UC_S	Sličan slučaj uporabe	0.9	0.7	0.8	0.85	0.75		0.8	0.8	#N/A	0.7 - 0.9
UC_R	Identičan slučaj uporabe	0.6	0.55	0.55	0.7	0.5		0.58	0.55	0.55	0.5 - 0.7

Tablica 4-7 Analiza rezultata drugog kruga odlučivanja metodom Delphi

Oznaka	Tehnički faktori	S1	S2	S3	S4	S5		Srednja vrijednost raspona produkt.	Medijan raspona produkt.	Mod vrijednost raspona produkt.	Raspon vrijednosti
		Raspon produktivnosti									
UCR-T1	Kompleksnost arhitekture	1.5	1.65	1.45	1.4	1.6		1.52	1.5	#N/A	1.4 - 1.65
UCR-T2	Podržavanje više platformi	1.35	1.5	1.4	1.2	1.4		1.37	1.4	1.4	1.2 - 1.5
UCR-T3	Integracija s novim sustavima	1.2	1.3	1.3	1.15	1.4		1.27	1.3	1.3	1.15 - 1.4

Oznaka	Faktori okoline	S1	S2	S3	S4	S5		Srednja vrijednost raspona produkt.	Medijan raspona produkt.	Mod vrijednost raspona produkt.	Raspon vrijednosti
		Raspon produktivnosti									
UCR-F1	Iskustvo u poznavanju aplikacije	1.35	1.25	1.25	1.2	1.15		1.24	1.25	1.25	1.15 - 1.35
UCR-F2	Iskustvo tima	1.25	1.25	1.2	1.3	1.3		1.26	1.25	1.25	1.2 - 1.3
UCR-F3	Zrelost zahtjeva	1.15	1.2	1.1	1.15	1.15		1.15	1.15	1.15	1.1 - 1.2
UCR-F4	Lociranost tima	1.05	1.05	1.1	1.15	1.15		1.1	1.1	1.05	1.05 - 1.15
UCR-F5	Veličina tima	1.05	1.1	1.1	1.1	1.05		1.08	1.1	1.1	1.05 - 1.1
UCR-F6	Održavanje projektne dokumentacije	1.05	1.15	1.1	1.15	1.1		1.11	1.1	1.15	1.05 - 1.15

UCR-F7	Kohezija tima	1	1.1	1.05	1	1.05		1.04	1.05	1	1	-	1.1

Oznaka	**Slučaj uporabe (smanjenje posla u odnosu na novi slučaj uporabe)**	**S1**	**S2**	**S3**	**S4**	**S5**		**Srednja vrijednost**	**Medijan veličina**	**Mod vrijednost**	**Raspon vrijednosti**
UC_S	Sličan slučaj uporabe	0.8	0.7	0.7	0.8	0.7		0.74	0.7	0.7	0.7 - 0.8
UC_R	Identičan slučaj uporabe	0.6	0.6	0.55	0.65	0.6		0.6	0.6	0.6	0.55 - 0.65

4.3.1. Linearno proporcionalne težine atributa

Tehničkim faktorima i faktorima okoline dodijeljene su tri moguće vrijednosti atributa; za matematički model procjene težine (parametri) tih vrijednosti su postavljene na način da su linearno proporcionalne (*L* se odnosi na atribut *slab*, *M* se odnosi na atribut *umjeren*, a *H* se odnosi na atribut *jak*):

$$L = 1 + x, \text{ M} = 1, \text{ H} = 1 - \text{x} \tag{4.8}$$

Za tehničke faktore vrijedi da atribut „slab“ zahtjeva manju količinu posla nego atribut „jak“ (L < H). Stoga, težina atributa „slab“ je manja nego težina atributa „jak“, a raspon produktivnosti se računa na sljedeći način:

$$PR_{medijan} = H/L \tag{4.9}$$

Pri čemu je $PR_{medjian}$ iznosi medijan raspona produktivnosti za svaki tehnički faktor iz drugog kruga metode Delphi.

Slijedno:

$$x = (1 - PR_{medijan}) / (1 + PR_{medijan}),$$

$$L = (1 - PR_{medijan}) / (1 + PR_{medijan}) + 1,$$

$$M = 1,$$

$$H = 1 - (1 - PR_{medijan}) / (1 + PR_{medijan}).$$

Za faktore okoline vrijedi da je za atribut „slab“ potrebna veća količina posla nego za atribut „jak“. Posljedično, iznos atributa „slab“ veći je od iznosa atributa „jak“ (L > H). Faspon produktivnosti se računa na sljedeći način:

$$PR_{medijan} = L/H \tag{4.10}$$

Pri čemu je $PR_{medjian}$ iznosi medijan raspona produktivnosti za svaki faktor okoline iz drugog kruga metode Delphi.

Slijedno:

$$x = (PR_{medijan} - 1) / (1 + PR_{medijan}),$$

$$L = (PR_{medijan} - 1) / (1 + PR_{medijan}) + 1,$$

$$M = 1,$$

$$H = 1 - (PR_{medijan} - 1) / (1 + PR_{medijan}).$$

Za novu klasifikaciju slučaja uporabe prema iskoristivosti (sličan i identičan slučaj uporabe) težinom se određuje medijan vrijednosti raspona produktivnosti za UC_S i UC_R.

4.3.2. Proces kalibracije

Cilj kalibracije je odrediti veličinu UCR izraženu u čovjek – sat za model UCR. Za svaki od projekata u kalibracijskom skupu veličina UCR računa se prema jednadžbi:

$$UCR[\check{c}ovjek - sat](Pi) = \frac{Stvarna\ količina\ posla\ (Pi)\ [\check{c}ovjek - sat]}{[UCR](Pi)} \quad (4.11)$$

Pri čemu Pi predstavlja projekte u kalibracijskom setu.

Povijesni podaci projekata iz programa A iz kalibracijskog skupa korišteni su za izračun veličine UCR izražene u čovjek – sat. Sakupljeni su povijesni podaci 16 projekata koji su zadovoljili kriterij uporabe modela UCR. Skup čine dva ili više projekata sa sličnim obimom posla te su se na jednom ili više projekata višestruko koristili artefakti inicijalnog projekta.

Veličina UCR izražena u čovjek – sat razlikuje se za inicijalne projekte u odnosu na slijedne projekte. Veličina UCR za inicijalne i slijedne projekte računa se prema jednadžbama 4.12 i 4.13:

$$UCR(inicijalni\ projekt) = \frac{\sum UCR\ (Pi\ inicijalni\ projekt)}{n(Pi\ inicijalni\ projekt)} \quad (4.12)$$

$$UCR(slijedni\ projekt) = \frac{\sum UCR\ (Pi\ slijedni\ projekt)}{n(Pi\ slijedni\ projekt)} \quad (4.13)$$

Pri čemu n predstavlja ukupan broj inicijalnih, odnosno slijednih projekata.

Tablica 4-8 prikazuje konačne težine svih ulaznih faktora modela UCR.

Tablica 4-8 Konačne težine ulaznih faktora modela UCR

Indeks	Opis aktera	Težina
A_L	Sustav koji komunicira definiranim aplikacijskim programskim sučeljem (Application Programming Interface - API)	1

A_M	Sustav koji komunicira protokolom TCP/IP	2
A_H	Sustav koji interakciju ostvaruje preko grafičkog korisničkog sučelja	3

Indeks	**Slučajevi uporabe (kriterij složenosti)**	**Težina**
UC_L	3 ili manje transakcija	5
UC_M	Od 4 do 7 transakcija	10
UC_H	Od 8 do 15 transakcija	15
UC_C	16 ili više transakcija	20

Indeks	**Slučajevi uporabe (kriterij iskoristivosti)**	**Težina**
UC_N	Novi slučaj uporabe	1.0
UC_S	Slični slučaj uporabe	0.7
UC_R	Identični slučaj uporabe	0.6

Indeks	**Tehnički faktori**	**L**	**M**	**H**
UCR-T1	Arhitektura softvera	0.80	1.00	1.20
UCR-T2	Podržavanje više platformi	0.83	1.00	1.17
UCR-T3	Integracija s novim sustavima	0.87	1.00	1.13

Indeks	**Faktori okoline**	**L**	**M**	**H**
UCR-F1	Iskustvo u poznavanju aplikacije	1.11	1.00	0.89
UCR-F2	Iskustvo tima	1.11	1.00	0.89
UCR-F3	Zrelost zahtjeva	1.07	1.00	0.93
UCR-F4	Lociranost tima	1.05	1.00	0.95
UCR-F5	Veličina tima	1.05	1.00	0.95
UCR-F6	Održavanje projektne dokumentacije	1.05	1.00	0.95
UCR-F7	Kohezija tima	1.02	1.00	0.98

UCR (veličina) za inicijalni projekt	10	čovjek – sat
UCR (veličina) za slijedni projekt	5.5	čovjek – sat

5. Validacija modela procjene količine posla

5.1. Proces provedbe eksperimenta

Eksperiment u programskom inženjerstvu je empirijsko ispitivanje kojim se manipulira jedan faktor ili varijabla proučavane postavke. Različita postupanja se primjenjuju na subjekte, zadržavajući ostale varijable konstantama te se pritom mjeri utjecaj na izlazne varijable [56]. Rezultat provođenja eksperimenta su statističke analize kojima se donose zaključci o proučavanim postavkama. Metode za statističko zaključivanje koriste se u eksperimentima radi demonstracije da jedna metoda postiže bolje rezultate od druge s iskazanom statističkom značajnošću [57] [58] [59].

Analiza studijskog primjera (*case study*) opisuje se kao tehnika u kojoj su identificirani ključni faktori koji mogu imati utjecaj na rezultat te se posljedično dokumentira njihova aktivnost [60] [61]. Istraživanje studijom slučaja je metoda promatranja, tj. provodi se promatranjem projekta ili aktivnosti koja se provodi.

Eksperiment je formalno, rigorozno i kontrolirano istraživanje [56]. Razlika između eksperimenta i studije slučaja je u razini kontrole konteksta [62]. U eksperimentu namjerno se izazivaju različite situacije i cilj je razlikovati uvjete tih situacija. U studiji slučaja, kontekstom upravlja stvarni projekt koji se promatra [56].

Za usporedbu dviju metoda procjene, istraživanje se može provesti u okviru eksperimenta ili studije slučaja, ovisno o razmjeru evaluacije, mogućnosti izoliranja faktora te mogućnosti nasumičnog razvrstavanja u skupine (randomizacije) [56].

S obzirom da se u usporedbi primjene modela UCP i UCR može točno utvrditi koji subjekti su u koje vrijeme i pri kojim uvjetima koristili neku od metoda, validaciju modela procjene moguće je provesti u sklopu eksperimenta.

Eksperimenti su prikladni za istraživanje različitih aspekata [63] [64], uključujući sljedeće primjere:

- potvrda teorija, tj. testiranje postojećih teorija,
- potvrda uvriježenog mišljenja (*conventional wisdom*), tj. testiranje ljudskog mišljenja,
- istraživanje odnosa (povezanosti),

- evaluacija preciznosti modela, tj. testiranje da je preciznost modela u skladu s očekivanim rezultatom,
- validacija mjera, tj. osigurati da pojedina mjera stvari mjeri ono što se očekuje.

Provedba eksperimenta uključuje sljedeće faze [56]:

1. Obim posla (*scoping*)

   U inicijalnoj fazi definira se problem istraživanja te cilj provođenja eksperimenta. Izražava se hipoteza, koja još ne mora biti formalno postavljena, ali mora biti jasna. Odgovara se na pitanja poput što se istražuje, koja je svrha i perspektiva, gdje se istraživanje provodi (kontekst).
2. Planiranje (*planning*)

   U drugoj fazi definira se dizajn eksperimenta, instrumentacija i prijetnje valjanosti (*threats to validity*). U sklopu dizajna, postavlja se formalna hipoteza, uključujući nul-hipotezu i alternativnu hipotezu. Utvrđuju se neovisne (ulazne) i zavisne (izlazne) varijable te njihove očekivane vrijednosti. Time se određuje ljestvica mjerenja koja uvjetuje metode za kasniju statističku analizu. Nominiraju se subjekti istraživanja.

   Primjereni objekti istraživanja se odabiru i pripremaju za provođenje eksperimenta. Uspostavljaju se procedure mjerenja.

   Određuju se očekivane prijetnje valjanosti istraživanja. Prijetnje valjanosti mogu se podijeliti u četiri klase: interne (unutarnje), vanjske, konstruktne i one za donošenje zaključka (zaključne?).
3. Izvedba (*operation*)

   Nakon faze dizajna, izvedba se provodi u tri koraka: priprema, izvršenje i validacija podataka. Prikupljaju se mjerenja provedbe eksperimenta.
4. Analiza i interpretacija (*analysis and interpretation*)

   Prikupljena mjerenja se analiziraju, radi neformalnog razumijevanja podataka koristi se opisna statistika čime se podaci mogu vizualizirati. Testira se hipoteza te se interpretiraju rezultati provođenja eksperimenta.
5. Dokumentiranje i prezentacija rezultata (*presentation and package*)

Rezultati istraživanja se dokumentiraju (moguće i u sklopu znanstvenog članka), bilježe se naučene lekcije te se bilježe uvjeti za eventualnu replikaciju eksperimenta. Rezultati se prezentiraju zainteresiranim stranama.

5.2. Opis i cilj validacije modela UCR

Cilj postupka validacije je evaluacija procijenjene količine posla iz perspektive projektnog tima u kontekstu industrijskih i akademskih projekata. U sklopu validacijskog procesa, model UCR se primjenjuje na inicijalnim i slijednim projektima.

5.2.1. Planiranje eksperimenta

Validacija modela UCR provodi se u sklopu eksperimenta koji evaluira preciznost procijenjene količine posla pri uporabi različitih modela procjene. Faktor u eksperimentu je model procjene, dok su postupci (*treatments*) postavljeni u okviru modela UCP i UCR. Subjekti su projektni timovi iz industrije, koji su dio globalne IT organizacije (za programe A i B) te doktorandi na Fakultetu elektrotehnike i računarstva, Sveučilišta u Zagrebu, Hrvatska (za program C). Objekti za validaciju su povijesni podaci iz validacijskog skupa projekata iz programa A, B i C.

Stručnjaci koji su sudjelovali u definiranju faktora i težina modela UCR bili su članovi određenih projekata unutar programa A (međutim, nisu imali veze s programom B niti C). Da bi se izbjegla potencijalna pristranost u validaciji modela, projekti u kojima su ti stručnjaci sudjelovali nisu u validacijskom skupu, stoga ti stručnjaci nisu među subjektima eksperimenta.

Točnost modela procjene promatra se pomoću MRE (veličina relativne greške). MRE se računa pomoću jednadžbe 2.1.

U sklopu eksperimenta definira se sljedeća hipoteza te mjerenja nužna za evaluaciju hipoteze:

1. Nul-hipoteza, H_0: Ne postoji razlika u točnosti procijenjene količine posla (mjerene pomoću MRE) pri primjeni dvaju modela procjene – UCP i UCR.
   H_0: *MRE (UCP) = MRE (UCR)*
   Alternativna hipoteza:

   H_1: *MRE (UCP) ≠ MRE (UCR)*

Procijenjene količine posla modelima UCP i UCR, te stvarna količina promatranih projekata su neovisne varijable unutar eksperimenta. Zavisna varijabla je MRE. Povijesni podaci projekata sadrže vrijednosti nezavisnih varijabli za ispitivanje hipoteze:

- procijenjena količina posla modelima UCP i UCR i stvarna količina posla promatranih projekata izražene su u veličini čovjek – sat, mjereno na omjernoj skali (*ratio scale*).

Svaki subjekt koristi oba postupka (oba modela UCP i UCR), što je temelj za korištenje eksperimenta s uparenim dizajnom [56] za testiranje hipoteze. Subjekti koji su sudjelovali u programu A procijenili su količinu posla projekata najprije pomoću modela procjene UCP, dok su subjekti koji su sudjelovali u programima B i C procijenili količinu posla projekata najprije pomoću modela procjene UCR. Dakle, ustanovljen je uravnoteženi dizajn jer imamo otprilike isti broj ispitanika u prvom i drugom postupku. Upareni dizajn analizira se uparenim t-testom.

Cook i Campbell [56] [65] su definirali četiri vrste prijetnji validaciji: unutarnju, vanjsku, konstruktivnu i prijetnju za donošenje zaključaka.

Prijetnje validaciji modela

Pouzdanost mjera smatra se prijetnjom za donošenje zaključka eksperimenta. Slučajeve uporabe definira i klasificira projektni tim, čime se postavljaju ulazni faktora oba modela, UCP i UCR. Stoga, kako bi se postigla dosljednost između inicijalnog i slijednih projekata, potrebno je primjenjivati dosljedna pravila u definiciji i klasifikaciji slučajeva uporabe.

Instrumentacija je unutarnja prijetnja valjanosti eksperimenta. Povijesni podaci o stvarnoj količini posla projekta moraju se prikupljati i prijaviti na isti način u različitim programima i različitim projektnim timovima.

Unutar konstruktivne valjanosti eksperimenta, postoji prijetnja pretpostavljanju hipoteze. Iako subjekti nisu bili dio stručne skupine za procjenu, neki su subjekti primijetili trend smanjenja napora u slijednim projektima.

Relativno mali broj dostupnih projekata za validaciju modela UCR (mali broj inicijalnih i slijednih projekata) smatra se vanjskom prijetnjom za valjanost rezultata.

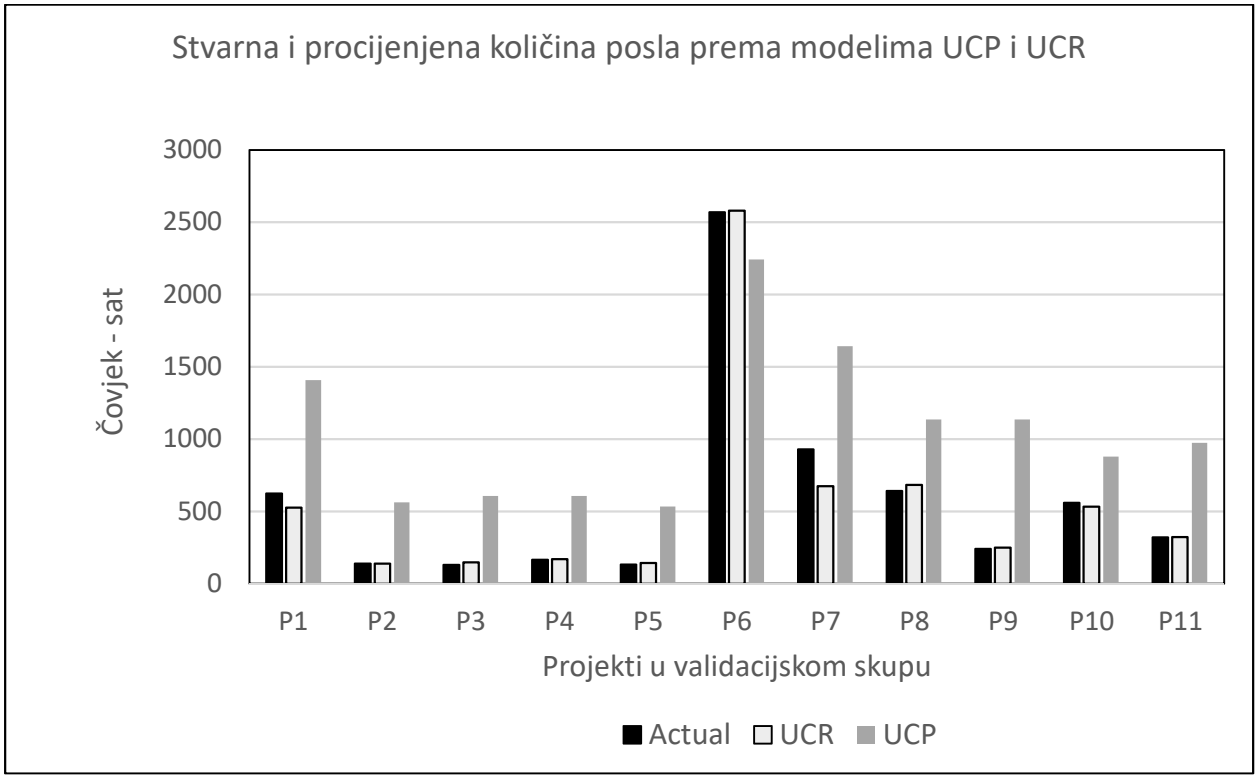
Izvedba eksperimenta – prikupljanje podataka

Povijesni podaci projekata prikupljeni su na dva načina: (1) voditelji projekata pružili su informacije o stvarnoj količini posla za svaki projekt i (2) voditelji projekata (u nekim slučajevima zajedno s arhitektima) dali su informacije o procijenjenoj količini posla prema modelima UCP i UCR. Podaci su prezentirani istraživaču te se osiguralo da su svi subjekti točno izvijestili o povijesnim podacima i da su primijenili UCP i UCR modele ispravnim redoslijedom. Takav je pristup korišten radi smanjenja utjecaja dviju prijetnji valjanosti: pouzdanosti mjera i instrumentacije.

Grafička vizualizacija podataka

Prikupljeni podaci o procijenjenoj količini posla projekata (prema modelima UCP i UCR) i stvarna količina posla prikazani su na Slika 5-1. Samo za jedan projekt unutar validacijskog seta (P6) stvarna i procijenjena količina posla (prema oba modela) znatno su veći u odnosu na druge projekte. Ti se podaci ne smatraju ekstremom (*outliner*) i neće biti isključeni iz skupa podataka jer je promatrani projekt imao širi obim posla nego drugi projekti unutar validacijskog skupa. Pored

toga, MRE po UCR modelu za projekt P6 je ispod 1%, što je još jedan argument da te podatke ne smatramo ekstremom.



Slika 5-1 Procijenjena i stvarna) količina posla prema modelima UCP i UCR

Parametrijsko testiranje (uparen t-test)

Uparen t-test proveden je prema izračunu u tablici 5-1 [56].

Tablica 5-1 Izračun t-testa

Stavka	Opis
Ulaz	Uparene vrijednosti: $(x_1, y_1), (x_2, y_2) \ldots (x_n, y_n)$, gdje x_i predstavlja MRE prema modelu UCP za projekt i, a y_i predstavlja MRE prema modelu UCR za projekt i.
H_0	$\mu_d = 0$, gdje je $d_i = x_i - y_i$, očekivana razlika srednjih vrijednosti 0

Izračuni	Račun $t0 = \frac{\bar{d}}{Sd/(\sqrt{n})}, gdje\ je\ Sd = \sqrt{\frac{\sum_{i=1}^{n}(d_i - \overline{d})^2}{n-1}}$, $\bar{d}$ je srednja vrijednosti razlike, S_d standardna devijacija, n je veličina uzorka, a $t0$ je t kvantila sa *n-1* stupnjeva slobode.
Kriteriji	Obostrano (H_1: $\mu_d \neq 0$): odbaci H_0 ako je $\lvert t_0 \rvert > t_{\frac{\alpha}{2},n-1}$. Ovdje, $t_{\alpha,f}$ je gornji α postotni bod t distribucije sa f stupnjeva slobode. Jednostrano (H_1: $\mu_d > 0$): odbaci H_0 ako je $\lvert t_0 \rvert > t_{\alpha,n-1}$.

MRE količine posla projekata u validacijskom skupu, procijenjene prema modelima UCP i UCR, prikazane su u tablici 5-2.

Tablica 5-2 MRE projekata prema modelima UCP i UCR

	Skup podataka za validaciju										
	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11
MRE (UCP)	125.5%	307.2%	365.4%	266.7%	306.9%	12.7%	76.9%	77.3%	372.9%	56.6%	204.1%
MRE (UCR)	15.8%	1.2%	13.6%	2.4%	9.0%	0.5%	27.3%	6.6%	3.9%	5.1%	0.6%

Nul-hipoteza navodi da nema razlike u procjeni točnosti (mjereno pomoću MRE) procijenjene količine posla pri primjeni modela UCP i UCR. Alternativa hipoteza navodi da postoji razlika.

Rezultati ispitivanja uparenih uzoraka prikazani su u tablici 5-3.

Tablica 5-3 Validacija modela UCR prema uparenom t-testu

Paired Samples Test

	Paired Differences					t	df	Sig. (2-tailed)
	Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference Lower	Upper			
UCP MRE - UCR MRE	1.896678	1.342326	.404727	.994891	2.798465	4.686	10	.001

Broj stupnjeva slobode je *f = n -1 = 11 – 1 = 10, t0 = 4.686 i t0.025,10 = 2.228* [31]. S obzirom da je *t0 > t0.025,10* moguće je odbaciti nul-hipotezu na razini 0.05.

Ograničen broj projekata (uzoraka) u sklopu provjere valjanosti može predstavljati vanjsku prijetnju valjanosti. Međutim, prema rezultatima uparenih t-testova, moguće je odbiti nul-hipotezu.

Rezultati eksperimenta

Odbijanjem nul-hipoteze, eksperiment pokazuje da postoji razlika u točnosti procjene (mjereno pomoću MRE), procijenjene količine posla pri primjeni modela UCP i UCR.

P-vrijednost jednaka je 0.0009. Prema konvencionalnim kriterijima, ta se razlika smatra statistički značajnom.

Točnost modela procjene također je promatrana ovim kriterijima: *MMRE* (srednja veličina relativne greške) i *PRED* (postotak predviđanja unutar 20%) procijenjenog projekta (20).

Sve vrijednosti veličine relativnih grešaka koriste se za izračun vrijednosti *MMRE* prema jednadžbi:

$$MMRE = \frac{1}{n} \sum_{i=1}^{n} MRE \tag{5.1}$$

Relativna greška srednje veličine računa se tako da pokazuje relativni iznos po kojem je procijenjena količina posla podcijenjena ili precijenjena u odnosu na stvarnu količinu posla. *MMRE* se u većini istraživačkih radova koristi kao kriterij ocjenjivanja zbog svojstava nezavisnih jedinica što znači da je *MMRE* neovisan o jedinicama procijenjene količine posla kao što su čovjek – sat, čovjek – mjesec itd. *MMRE* je značajan alat za sažetak statistike i vrlo je važan pokazatelj u modelu procjene količine posla [19].

PRED (x) je postotak procjena koji su unutar x% stvarne količine posla. *PRED* (20) je postotak procijenjene količine posla koji ju unutar 20% iznosa stvarne količine posla.

Obje mjere, *MMRE* i *PRED* (20) pokazuju obećavajuće rezultate za projekte unutar validacijskog seta pri primjeni modela UCR.

Tablica 5-4 Rezultati eksperimenta pri primjeni modela UCR (MMRE, MdMRE i PRED (20)

MMRE (UCR)	7.81%
MdMRE (UCR)	5.07%
PRED (20) (UCR)	91%

5.3. **Primjena modela procjene UCR**

Početni preduvjet za procjenu količine posla pomoću modela UCR je formiranje tima stručnjaka s adekvatnim znanjem procesa procjene količine posla i iskustvom u projektima koji višestruko koriste prethodno razvijene artefakte. S druge strane, ne može se očekivati da postoji veliki broj takvih stručnjaka unutar organizacije.

Za UCR model procjenitelji trebaju postaviti sve ulazne faktore za svaki projekt (inicijalni i slijedni projekti):

- broj aktera i njihova klasifikacija prema kriteriju složenosti,

- broj slučajeva uporabe i njihova klasifikacija po složenosti i kriteriju iskoristivosti,

- tehnički faktori koji se odnose na softverski proizvod,

- faktori okoline koji se odnose na projektni tim.

Ako je potrebno, stručni tim može prilagoditi model UCR za svoje specifično okruženje slijedeći korake:

1) procjena ulaznih faktora u predloženom modelu UCR od strane stručnog tima (pregled svih aspekata koji utječu na količinu posla potrebnu u njihovom projektnom okruženju),

2) potencijalno dodavanje ili uklanjanje ulaznih faktora,

3) podešavanje TCF_R ili EFR jednadžbi (jednadžbe 4.4 i 4.5) ako su promijenjeni ulazni faktori,

4) postavljanje težina atributa za nove ulazne faktore provođenjem eksperimenta Delphi,

5) kalibracija veličine UCR izražene u čovjek – sat.

Drugi potencijalni scenarij primjene modela UCR je izvedba isključivo posljednjeg koraka iz gore opisanog postupka adaptacije: kalibracija veličine UCR (izraženo u čovjek – sat) za određeni

projektni tim. Veličina UCR označava produktivnost određenog projektnog tima [66] i na temelju povijesnih podataka o stvarnoj količina posla i ulaznim faktorima za ostale projekte u toj organizaciji, različita vrijednost veličine UCR (za inicijalne i slijedne projekte) može biti primjenjiva za drugu organizaciju.

5.4. **Iskustvo primjene modela UCR**

Primjena UCR modela pokazuje poboljšane rezultate procijenjene količine posla u usporedbi s modelom UCP za promatrane projekte unutar studijskog primjera: apsolutne vrijednosti *MRE* i *MMRE* za model UCR su niže od apsolutnih vrijednosti *MRE* i *MMRE* za UCP model, a *PRED (20)* za UCR je veći od *PRED (20)* za UCP model. U praksi preciznije procjene količine posla omogućuju projektnom timu poboljšanje cjelokupnog procesa upravljanja projektima - planiranje, raspoređivanje, upravljanje resursima i procjenu troškova. Projekti čiji su se povijesni podaci koristili za kalibraciju i validaciju modela dominantno su male veličine u smislu obima posla projekata (mali ili umjereni broj slučajeva uporabe, do 22 slučajeva uporabe) i veličine tima (do pet članova tima), tako da bi se mogla očekivati određena odstupanja u procjeni količini posla za srednje ili velike projekte.

Kao dio budućeg rada, postoji mogućnost da se za projekte srednje ili velike veličine proširi ljestvica atributa faktora u modelu UCR (npr. uvođenjem atributa *vrlo slab* i *vrlo jak*). Veličina projekta i složenost projekta povećavaju se od malih do velikih projekata: projekti srednje i velike veličine imaju umjereno visok broj isporučenih artefakata koji su obično tehnički složeniji, broj članova tima se povećava, a rokovi isporuke povećavaju. Dodatna granularnost atributa omogućila bi preciznije razlikovanje projekata u smislu tehničke složenosti proizvoda i karakteristika projektnog tima.

6. Postupak dinamičke prilagodbe parametara modela za procjenu količine posla zasnovan na strojnom učenju

6.1. Uvod

Pojam strojnog učenja (*Machine Learning* – ML) Arthur Samuel opisao je još sredinom 20. stoljeća kao „područje u kojem se računalima daje sposobnost učenja bez eksplicitnog programiranja“ [67]. Nakon desetljeća napretka u domeni strojnog učenja koje je inicijalno obilježeno algoritmima do kasnijeg dovođenja u vezu s teorijom optimizacije, današnji pojam strojnog učenja veže se uz granu umjetne inteligencije koja se bavi oblikovanjem algoritama koji svoju učinkovitost poboljšavaju na temelju empirijskih podataka.

Mitchell je pojam strojnog učenja opisao sljedećom definicijom: program uči iz iskustva E u odnosu na zadatak T i mjeru sposobnosti P, ako se njegova sposobnost na zadatak T, mjerena s P, poboljšava s iskustvom E [68].

Tri su osnovne podjele modela strojnog učenja s obzirom na vrstu učenja: **modeli nadziranog strojnog učenja** (*supervised machine learning*), **modeli strojnog učenja bez nadzora** (*unsupervised machine learning*) i **modeli podržanog učenja** (*reinforcment learning*) [69].

U **nadziranom učenju** polazišna točka za definiciju modela je skup za učenje (*training set*) koji se sastoji od ulaznih podataka i odgovarajućih izlaznih podataka. Na tom skupu za učenje provodi se algoritam učenja; na temelju rezultata algoritma model je sposoban točno odrediti izlazne vrijednosti na skupu za učenje, ali i za neke nove, potpuno nepoznate podatke [69].

Modeli učenja bez nadzora najčešće se odnose na modele u kojima se pronalaze sličnosti među ulaznim podacima te sukladno tome grupiraju u različite skupine. Služe za detekciju anomalija u podacima te prepoznavanju uzoraka [69].

Modeli podržanog učenja temelje se na dinamičkoj interakciji s okolinom od koje se prikupljaju odgovori za pojedine upite, odnosno ponašanja (često putem nagrada za željeno ili kazni za neželjeno ponašanje). Temeljem povratne informacije okoline, model se uči ponašanju u toj okolini i pokušava maksimizirati nagradu koja je podešena na način da vodi ka zadanom cilju [69].

Aktivnosti modela strojnog učenja dijele se na sljedeće tipove ovisno o željenoj vrsti izlaznih podataka, odnosno ovisno o vrsti proučavanog problema koji se želi riješiti [70]: u **klasifikaciji** su ulazni podaci unaprijed podijeljeni na dvije ili više klasa te se modelom pridružuju nepoznati ulazni podaci s jednom ili više definiranih klasa – najčešće pod nadzorom. U **regresiji** se primjenjuju matematičke funkcije koje opisuju zavisnost jedne varijable o jednoj ili više drugih varijabli. U takvim modelima izlazni podaci nisu diskretni već kontinuirani (također se radi o nadziranom učenju). U **grupiranju** (*clustering*) skup ulaznih podataka se grupira, međutim za razliku od klasifikacije, grupe nisu ranije poznate (učenje bez nadzora). Slični primjeri (slični po nekom svojstvu) svrstavaju se u istu grupu, a različiti primjeri u različite grupe. Svrha grupiranja jest nalaženje "prirodnih" (intrinzičnih) grupa u skupu neoznačenih podataka [71]. **Smanjenje dimenzija** odnosi se na transformaciju inicijalnog prikaza sadržaja u nižu dimenziju prikaza tog sadržaja uz očuvanje nekih svojstava inicijalnog prikaza (npr. procesiranje digitalnih slika).

6.2. Opis problema primjene algoritamskih modela za procjenu količine posla

Pri definiciji algoritamskog modela za procjenu količine posla koristi se znanje, odnosno iskustvo prikupljeno u određenom vremenskom periodu. Algoritamski model čine ulazni faktori, težine ulaznih faktora i postavljeni matematički model (algoritam) kojim se računa procijenjena količina posla. Nakon definicije modela, isti se koristi u daljnjim projektima te ne postoji povratna veza između rezultata njegove primjene u pojedinoj organizaciji te definicije samog modela. Kod uspješne primjene algoritamskog modela (preciznih rezultata) preinake na modelu nisu potrebne, no u praksi je uočen problem da kod nekih organizacija preciznost procjene količine posla opada s vremenom na novim projektima.

Istraživanja u domeni procjene količine posla za razvoj programskog proizvoda posljednjih godina sve veći fokus daju metodama strojnog učenja [72]; metode zasnovane na strojnom učenju svrstavaju se među tri glavne kategorije metoda procjene količine posla [73] [74] [75] pored ranije spomenutih metoda ekspertne procjene i algoritamskih modela. Primarni cilj tih metoda je poboljšanje procjene količine posla pomoću definicije sustava učenja temeljenog na jednoj ili više metoda strojnog učenja, a koji koristi povijesne podatke količine posla za razvoj programskog proizvoda [76].

Empirijske studije o primjeni modela strojnog učenja donose neujednačene zaključke o preciznosti procijenjene količine posla pri usporedbi rezultata modela strojnog učenja i ostalih vrsta modela procjene te usporedbi između različitih modela strojnog učenja [72]. Neujednačenost u zaključcima različitih studija ide u prilog prethodno usvojenoj najboljoj praksi za procjenu količine posla: razvoj i primjena dvaju ili više modela procjene različitih kategorija (ekspertna procjena, algoritamski model ili model zasnovan na strojnom učenju) kako bi se došlo do što preciznije procjene količine posla.

6.3. Proces oblikovanja modela procjene količine posla zasnovanog na nadziranom strojnom učenju

Primjenom sustava učenja model procjene količine posla moguće je poboljšati kroz vrijeme koristeći nove povijesne podatke. Na taj način proces procjene se nadograđuje i ažurira s novim projektima, što ga diferencira od modela procjene temeljenog na algoritamskom modelu.

Oblikovanje modela zasnovanog na nadziranom strojnom učenju uključuje dvije faze: fazu učenja i fazu testiranja [77]. U praksi, faza testiranja označava često i fazu primjene modela.

Faza učenja (*training*) podrazumijeva proces u kojem algoritmi zasnovani na strojnom učenju „uče“ pomoću određenog skupa podataka za učenje. Algoritam pronalazi obrasce u podacima za učenje na temelju kojih povezuje atribute ulaznih podataka s odgovorom (odgovor koji se želi predvidjeti) i kreira model koji slijedi te obrasce. Podaci za učenje pored ulaznih podataka (parametara) moraju sadržavati i točan odgovor, poznat kao cilj ili ciljni atribut. U ovoj fazi termin „model strojnog učenja“ odnosi se na artefakt koji je kreiran procesom treniranja.

U drugoj fazi testiranja model kreiran po izvršenju faze učenja obrađuje testni skup podataka i temeljem definiranih obrazaca sam predviđa odgovor. S obzirom da su i za testni skup podataka unaprijed poznati odgovori, usporedbom ranije poznatih odgovora i onih koje je izračunao model, može se odrediti preciznost procjene modela prema greški procjene.

Za razvoj modela za procjenu količine posla zasnovanog na strojnom učenju potrebno je provesti sljedeće korake [78] [79]:

1. **Prikupljanje povijesnih podataka**

   Faza obuhvaća početno prikupljanje podataka, opis podataka, istraživanje podataka i verifikaciju kvalitete podataka iz projektne dokumentacije. Postavljaju se poslovni zahtjevi i ciljevi, određuju se kriteriji uspješnosti provedbe obrade podataka.

   Konkretno za model procjene količine posla, u povijesne podatke ubrajaju se iznos stvarne količine posla za razvoj proizvoda po završetku projekta te ostali podaci kojima se određuju vrijednosti ulaznih podataka modela (npr. broj i vrste slučajeva uporabe i aktera, broj članova tima, itd.).

2. **Priprema podataka**

   Aktivnosti ove faze obuhvaćaju izdvajanje i obradu podataka koji se dovode u vezu s procesom procjene količine posla (podaci o programskom proizvodu koji se razvija, projektnom timu i stvarnoj količini posla razvoja proizvoda po završetku projekta).

   Izdvajanjem podataka selektiraju se tzv. ulazni faktori modela procjene količine posla jer se radi o podacima koji utječu na stvarnu količinu posla. Stvarna količina posla postavlja se kao ciljna vrijednost, odnosno stvara se uzročno – posljedična veza između ulaznih podataka i očekivanog rezultata.

   Iz daljnjeg procesa razvoja modela isključuju se nepotpuni podaci i ekstremne vrijednosti.

   S obzirom na tip povijesnih podataka, ulazni faktori i rezultat, klasificiraju se prema tipu mjerenja (nominalni, diskretni, kontinuirani, itd.).

3. **Slučajna podjela podataka na skupove podataka za učenje i testiranje**

   Iz dostupnog skupa podataka, potrebno je nasumično odrediti koji podaci pripadaju skupu podataka za učenje, a koji skupu podataka za testiranje. U literaturi nisu precizno definirani omjeri podjele podataka u dva skupa; spominju se omjeri 80:20 (80% podataka je u skupu za učenje, a 20% u skupu za testiranje), kao i pridruživanje dvije trećine podataka skupu podataka za učenje, dok preostalu jednu trećinu čini skup podataka za testiranje [80] [81].

4. **Oblikovanje modela za procjenu količine posla**

U ovoj fazi primjenjuju se razne tehnike modeliranja prilikom kojih se oblikuju i procjenjuju razni prediktivni modeli koristeći prethodno definirane skupove podataka za učenje i testiranje. Oblikovanje modela je iterativan proces u kojem se parametri modela kalibriraju na optimalne vrijednosti.

5. **Evaluacija (procjena) modela za procjenu količine posla**

   Nakon oblikovanja modela, potrebno je procijeniti pridonose li rezultati obrade podataka postizanju preciznije procjene količine posla. Pri kraju ove faze potrebno je utvrditi na koji način se koriste rezultati obrade podataka u projektnom okruženju.

6. **Primjena modela za procjenu količine posla**

   Ovisno o zahtjevima, u fazi primjene oblikovani model može poslužiti za generiranje izvješća o obrađenim podacima. U slučaju da je svrha modela proširenje znanja o podacima, stečeno znanje potrebno je organizirati i predstaviti na način da projektni timovi koje ih koriste na temelju istih mogu donijeti odluke.

Među često korištenim metodama za procjenu količinu posla razvoja programskog proizvoda nalaze se razne tehnike regresije koje rezultiraju linearnim i nelinearnim modelima (neuronske mreže, stablo odlučivanja) [81] [82].

Metoda linearne regresije koristi se pri procjeni količine posla u slučajevima kada su ulazna i izlazna varijabla linearno korelirane. Višestruka linearna regresija podrazumijeva linearnu korelaciju između više ulaznih varijabli i jedne izlazne varijable. Metoda pronalazi funkciju koja najbolje pridružuje stvarnu vrijednost zavisne (izlazne) varijable s jednom ili više neovisnih (ulaznih) varijabli. Ova metoda u osnovi pomaže shvatiti kako se vrijednost zavisne varijable mijenja u odnosu na vrijednosti nezavisnih varijabli.

Pod pojmom „neuronske mreže“ smatraju se umjetne neuronske mreže koje su nastale po principima bioloških neuronskih mreža. Djeluju simuliranjem velikog broja međusobno povezanih procesorskih jedinica koje nalikuju apstraktnim verzijama neurona. Umjetne neuronske mreže imaju sposobnost učenja i modeliranja nelinearnih i složenih odnosa te se mogu primijeniti u raznim problemima procjenjivanja u kojima su odnosi između ulaza i izlaza nelinearni. Nakon učenja iz inicijalnih ulaznih podataka i njihovih odnosa, umjetna neuronska mreža može donijeti

zaključke za nove veze na novim podacima, čime se model generalizira i omogućuje predviđanja za nove podatke.

Postoje više metoda umjetnih neuronskih mreža, dok se za procjenu količinu posla razvoja programskog proizvoda prema provedenom istraživanju literature najčešće koriste [83]:

- mreže bez povratnih veza (*feed – forward neural networks*),
- mreže s povratnim vezama (*recurrent neural networks*),
- radijalne mreže (*radial basis function - RBF network*),
- „neuro – fuzzy“ mreže *(neuro-fuzzy networks).*

Metoda stablo odlučivanja nastala je na temelju statističkih metoda raspoznavanja uzoraka. Široku primjenu pronalazi u rješavanju prediktivnih problema uz nadzor učenja. Prediktivni problemi uključuju predviđanje vrijednosti izlazne varijable u budućnosti, prepoznavanje uzoraka, regresiju više značajki i razlikovnu analizu. Kao i kod ostalih metoda, za predviđanje vrijednosti izlazne varijable koristi se skup ulaznih varijabli.

U sklopu studijskog primjera (Poglavlje 6.4) oblikovat će se modeli procjene količine posla na temelju metoda linearne regresije, umjetnih neuronskih mreža i stabla odlučivanja.

6.3.1. Alat IBM SPSS Modeler

Modeli strojnog učenja namijenjeni su obradi velike količine podataka te je stoga za razvoj modela preporuka koristiti računalne alate koji u sebi imaju ugrađeni široki skup algoritama temeljenih na strojnom učenju. Za potrebe razvoja modela za procjenu količine posla zasnovanog na strojnom učenju u studijskom primjeru koristit će se softverski program za rudarenje podataka (*data mining*) i analizu teksta (*text analysis*) IBM SPSS Modeler. Proces oblikovanja modela neovisan je o alatu SPSS te ga je moguće provesti i u drugim dostupnim alatima.

Alat SPSS Modeler koristi se za izgradnju modela za predviđanje i provođenje raznih analitičkih zadataka. Ima vizualno sučelje koje korisnicima omogućuje uporabu statističkih algoritama i algoritama za analizu podataka. Jedan od njegovih glavnih ciljeva je uklanjanje nepotrebne složenosti u transformacijama podataka te učiniti složene predvidljive modele jednostavnima za korištenje. Alat je korišten u brojnim studijama predviđanja te ima široku primjeni ne samo u IKT, već i u drugim industrijama [84] [85] [86] [87] [88] [89] [90].

SPSS ima niz ugrađenih metoda modeliranja temeljenih na umjetnoj inteligenciji, strojnom učenju i statističkim metodama. Metode modeliranja dijele se u tri kategorije: klasifikacija (*classification*), pridruživanje (*association*) i segmentacija (*segmentation*) [79].

Modeli klasifikacije koriste vrijednosti jednog ili više ulaznih polja za predviđanje vrijednosti jednog ili više izlaznih (ciljnih) polja. Neki primjeri ovih tehnika su: stabla odluke (stablo C-RT, QUEST, CHAID i C5.0 algoritmi), regresije (linearni, logistički, generalizirani linearni i Cox regresijski algoritmi), umjetne neuronske mreže, strojevi za podršku vektora i Bayesove mreže.

Modeli klasifikacije pomažu organizacijama predvidjeti poznati rezultat. Tehnike modeliranja uključuju pravilo indukcije, identifikaciju podskupine, statističke metode i generiranje višestrukih modela.

Modeli pridruživanja (asocijacije) nalaze uzorke među povezanim podacima. Modeli konstruiraju skup pravila koji definiraju odnose među podacima. Asocijacije se mogu pronaći ručno, međutim algoritmi s pravilima pridruživanja to čine mnogo brže i mogu istražiti složenije uzorke. Modeli Apriori i Carma primjeri su uporabe takvih algoritama. Druga vrsta asocijacijskog modela jest model detekcije slijeda, koji pronalazi uzastopne obrasce u vremenskim strukturiranim podacima. Modeli pridruživanja najkorisniji su u predviđanju višestrukih ishoda kada se određeni zaključci povezuju s nizom uvjeta.

Modeli segmentacije (također poznati kao "modeli grupiranja") dijele podatke u segmente zapisa (*clusters*) prema sličnim obrascima ulaznih podataka. Primjeri modela segmentacije su Kohonen mreže, grupiranje K-srednjih vrijednosti (*K-Means clustering*), dvo-stupanjsko grupiranje i otkrivanje anomalija. Modeli segmentacije korisni su u slučajevima gdje je specifičan rezultat nepoznat. Usmjereni su na identificiranje skupina sličnih zapisa i označavanje zapisa prema grupi kojoj pripadaju bez prethodnog znanja o skupinama i njihovim karakteristikama. Za razliku od ostalih tehnika modeliranja, modeli segmentiranja nemaju unaprijed definirano izlazno ili ciljno polje za predviđanje modela. Za takve modele nema ispravnih ili pogrešnih odgovora. Njihova je vrijednost određena njihovom sposobnošću uočavanja specifičnih grupacija u podacima i korisnim opisima tih grupacija.

6.4. Studijski primjer - oblikovanje modela za procjenu količine posla temeljenog na strojnom učenju

Naredno poglavlje posvećeno je oblikovanju novog modela za procjenu količine posla zasnovanog na strojnom učenju nazvanog UCR_ML. Model UCR_ML će koristeći više tehnika strojnog učenja procijeniti količinu posla na temelju ulaznih podataka završenih projekata. U tom modelu dinamički će se prilagođavati težine atributa (parametara) ranije definiranih faktora u modelu UCR radi izračuna količine posla te će se ti rezultati usporediti s procjenom količine posla osnovnog modela UCR.

Oblikovanje modela temelji se na koracima obrade podataka opisanim u poglavlju 6.3.

Prikupljanje povijesnih podataka za razvoj modela UCR_ML

Za potrebe definicije modela UCR_ML korišteni su ranije prikupljeni i obrađeni povijesni podaci ukupno 29 projekata (isti povijesni podaci korišteni su za kalibraciju i validaciju modela UCR).

Priprema i unos podataka za razvoj modela UCR_ML

Ulazni faktori modela UCR čine ulazne faktore modela UCR_ML, prikazani su u Tablici 6-1:

Tablica 6-1 Ulazni faktori modela UCR_ML

UAW_R	$UUCW_R$	UCR-T1	UCR-T2	UCR-T3	UCR-F1	UCR-F2	UCR-F3	UCR-F4	UCR-F5	UCR-F6	UCR-F7

- UAW_R se računa prema jednadžbi 4.1,
- $UUCW_R$ se računa prema jednadžbi 4.2,
- tehnički faktori (*UCR_T1*, *UCR_T2* i *UCR_T3*) imaju vrijednost dodijeljenog atributa (L za *slab*, M za *umjeren* i H za *jak*), prema opisu atributa u tablici 4.3,
- faktori okoline (*UCR_F1*,…, *UCR_F7*) imaju vrijednost dodijeljenog atributa (L za *slab*, M za *umjeren* i H za *jak*), prema opisu atributa u tablici 4.4.

S obzirom da je broj dostupnih projekata ograničen, ulazni faktori koji karakteriziraju atribute aktera objedinjuju se u ulazni faktor UAW_R (prema jednadžbi 4.1) u modelu UCR_ML. S druge strane, ulazni faktori koji karakteriziraju atribute slučajeva uporabe objedinjuju se u ulazni faktor $UUCW_R$ (prema jednadžbi 4.2) u modelu UCR_ML. Svaki od tehničkih faktora i faktora okoline

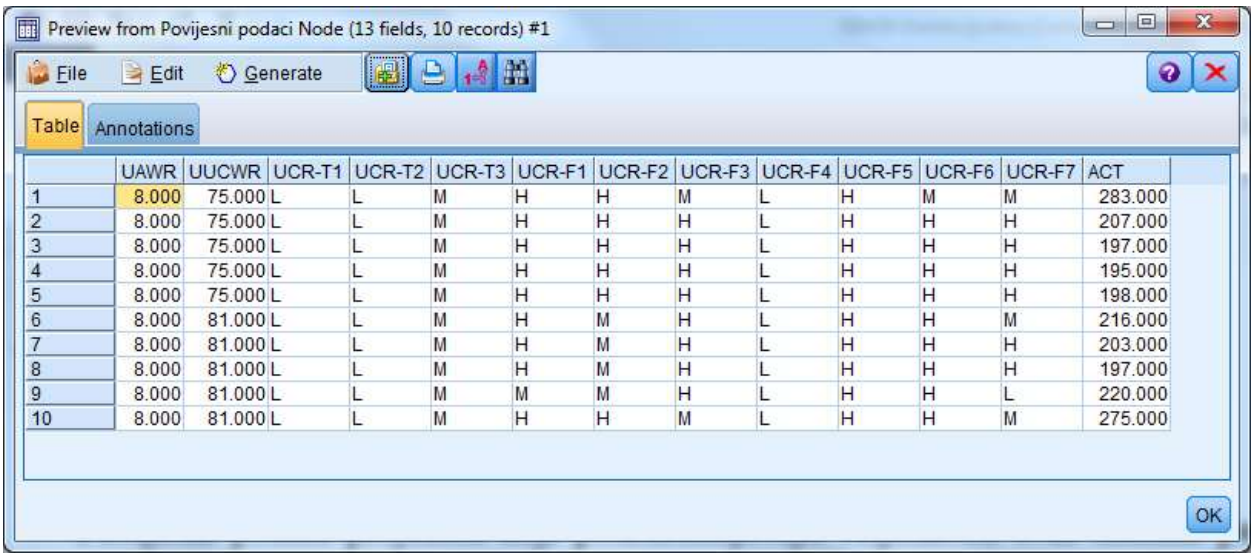
ima više poznatih vrijednosti, stoga se svaki tehnički faktor i faktor okoline uključuje kao ulazni faktor modela UCR_ML.

Težine atributa (parametri) se unutar modela UCR_ML dinamički prilagođavaju modelima, stoga u modelu UCR_ML nema zasebnog indikatora (atributa) među ulaznim faktorima radi li se o inicijalnom ili slijednom projektu. Međutim, višestruka iskoristivost se neposredno odražava kroz izračun $UUCW_R$ (u izračunu se postavljaju vrijednosti slučajeva uporabe prema klasifikaciji iskoristivosti).

Obradom podataka zaključeno je da nema atributa ulaznih faktora s ekstremnim vrijednostima koji bi eventualno uzrokovali izuzimanje povijesnih podataka nekog projekta.

Faktorima UAW_R i $UUCW_R$ dodjeljuju se numeričke vrijednosti faktora, prema jednadžbama u UCR modelu (jednadžbe 4.1 i 4.2). Ulazne vrijednosti tehničkih faktora i faktora okoline predstavljaju dodijeljeni atribut (*low*, *medium* ili *high*). Težine tehničkih faktora i faktora okoline definirane u modelu UCR neće se koristiti u modelu UCR_ML; one će se dinamički definirati u raznim modelima strojnog učenja.

Unos povijesnih podataka i dodijeljenih vrijednosti prikazan je na slici 6-1 (prikazuju se samo podaci za deset od ukupno dvadeset i devet projekata), dok je u alatu označen čvorom „Povijesni podaci“.

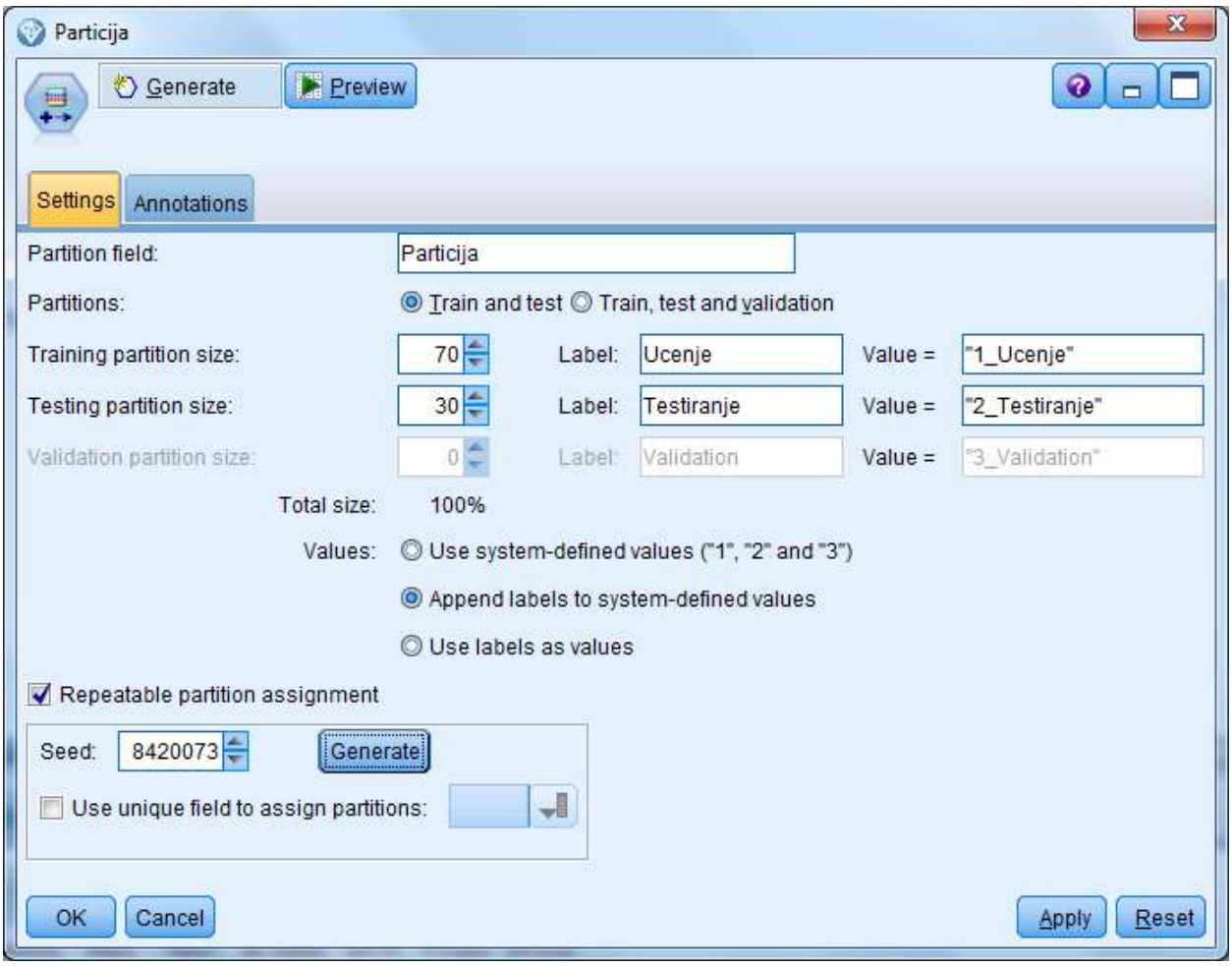


	UAWR	UUCWR	UCR-T1	UCR-T2	UCR-T3	UCR-F1	UCR-F2	UCR-F3	UCR-F4	UCR-F5	UCR-F6	UCR-F7	ACT
1	8.000	75.000	L	L	M	H	H	M	L	H	M	M	283.000
2	8.000	75.000	L	L	M	H	H	H	L	H	H	H	207.000
3	8.000	75.000	L	L	M	H	H	H	L	H	H	H	197.000
4	8.000	75.000	L	L	M	H	H	H	L	H	H	H	195.000
5	8.000	75.000	L	L	M	H	H	H	L	H	H	H	198.000
6	8.000	81.000	L	L	M	H	M	H	L	H	H	M	216.000
7	8.000	81.000	L	L	M	H	M	H	L	H	H	H	203.000
8	8.000	81.000	L	L	M	H	M	H	L	H	H	H	197.000
9	8.000	81.000	L	L	M	M	M	H	L	H	H	L	220.000
10	8.000	81.000	L	L	M	H	H	M	L	H	H	M	275.000

Slika 6-1 Unos povijesnih podataka u model UCR_ML

Podjela podataka u skupove za učenje i testiranje

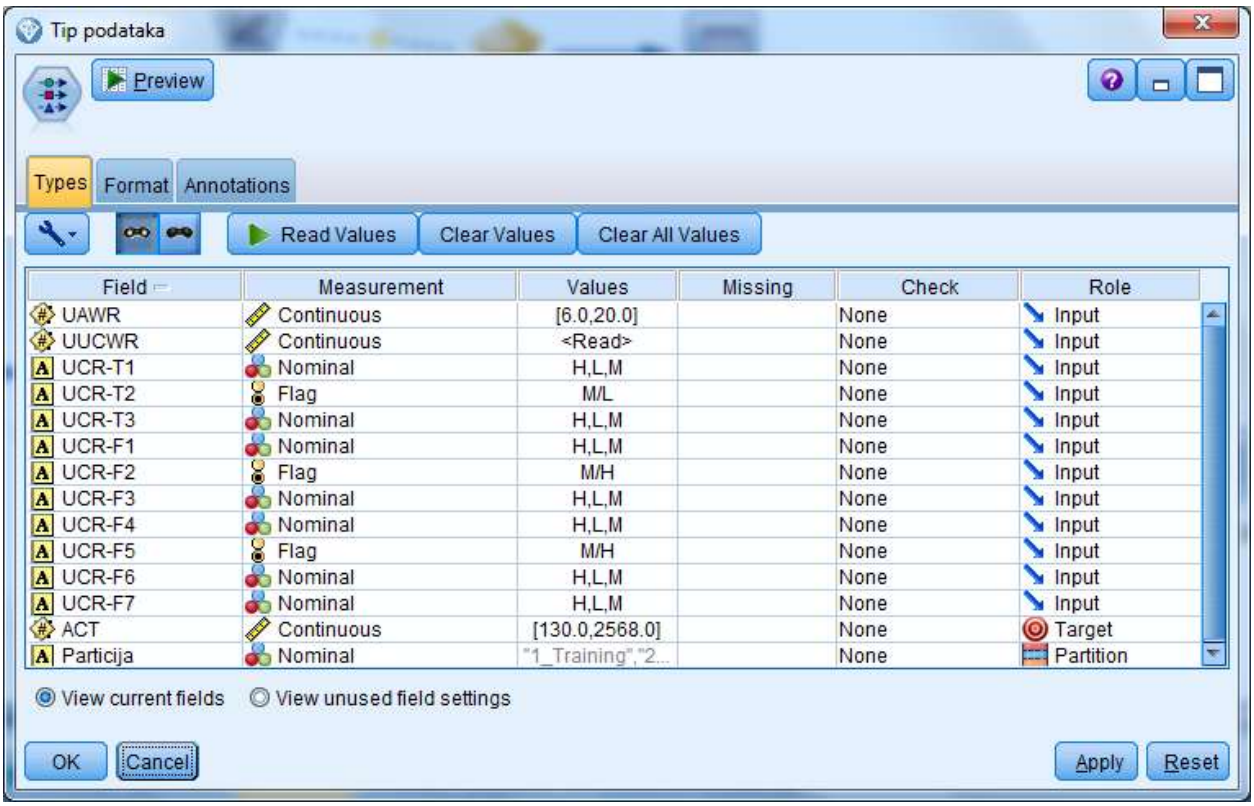
Povijesni podaci projekata koji podrazumijevaju vrijednosti svih ulaznih parametra i stvarnu količinu posla nasumično su podijeljeni u omjeru 70:30 u skup podataka za učenje i skup podataka za testiranje (Slika 6-2) u sklopu čvora „Particija“.



Slika 6-2 Podjela povijesnih podataka u skupove za učenje i testiranje

Kategorizacija podataka

Nakon unosa vrijednosti povijesnih podataka i podjele istih u skupove za učenje i testiranje, u sklopu čvora „Tip podataka“ slijedi daljnja kategorizacija varijabli budućeg modela za procjenu količine podataka. Stvarna količina posla (faktor „ACT“) predstavlja ciljani atribut, odnosno zavisnu varijablu koja će predstavljati rezultat modela, kao što je prikazano na slici 6-3 ostali faktori predstavljaju ulazne neovisne varijable modela za procjenu količine posla.

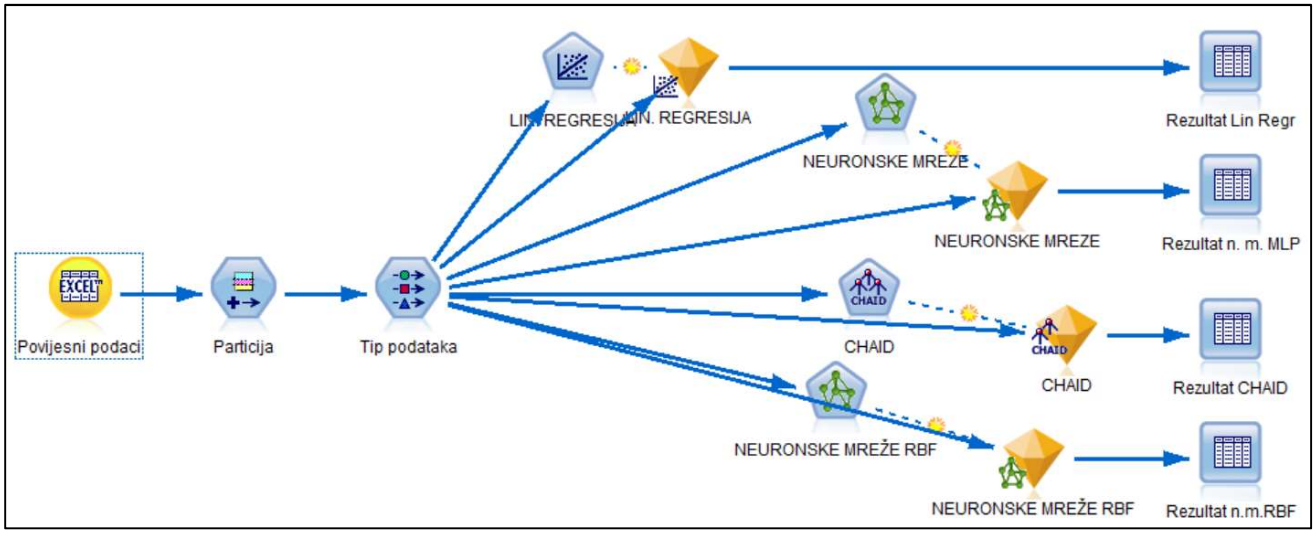


Slika 6-3 Definicija ciljnog atributa u algoritmu za učenje za studijski primjer

Slika 6-3 pokazuje klasifikaciju tipa faktora (*measurement*) prema postavljenim vrijednostima. Svi faktori s numeričkim vrijednostima (tj. numeričke varijable) klasificirani su kao kontinuirane varijable (*continuous*), dok su faktori s vrijednostima *L, M i H* klasificirani kao nominalni jer se odnose na kategorički diskretne podatke.

Oblikovanje modela UCR_ML

Tipovi faktora, kao i broj nezavisnih i zavisnih varijabli uvjetuju i odabir prikladne statističke metode, odnosno modela strojnog učenja. Modeli klasifikacije primjereni su izbor za izgradnju modela procjene količine posla (UCR_ML) s obzirom da su takvi modeli usmjereni na razne vrste procjena jedne ili više varijabli. Među modelima klasifikacije unutar alata SPSS, odabrani su oni koji daju numeričku vrijednost ciljnog rezultata. Koristeći ranije postavljene ulazne podatke, provedeno je modeliranje prema niže navedenim metodama. Rezultati svake od metoda nalaze se u nastavku. Proces modeliranja u SPPS-u prikazan je na slici 6-4.



Slika 6-4 Proces modeliranja za studijski primjer

1) Linearni regresijski model

Modelom se predviđa kontinuirani cilj koji se temelji na linearnim odnosima između cilja i jednog ili više ulaznih podataka. Ograničavajuća karakteristika modela je što kao ulazna polja može koristiti samo faktore s numeričkim (kontinuiranim) vrijednostima: UAW_R i $UUCW_R$, dok su ostali ulazni faktori zanemareni. Niže u tablici 6-2 prikazan je sažetak modela.

Tablica 6-2 Sažetak linearno regresijskog modela

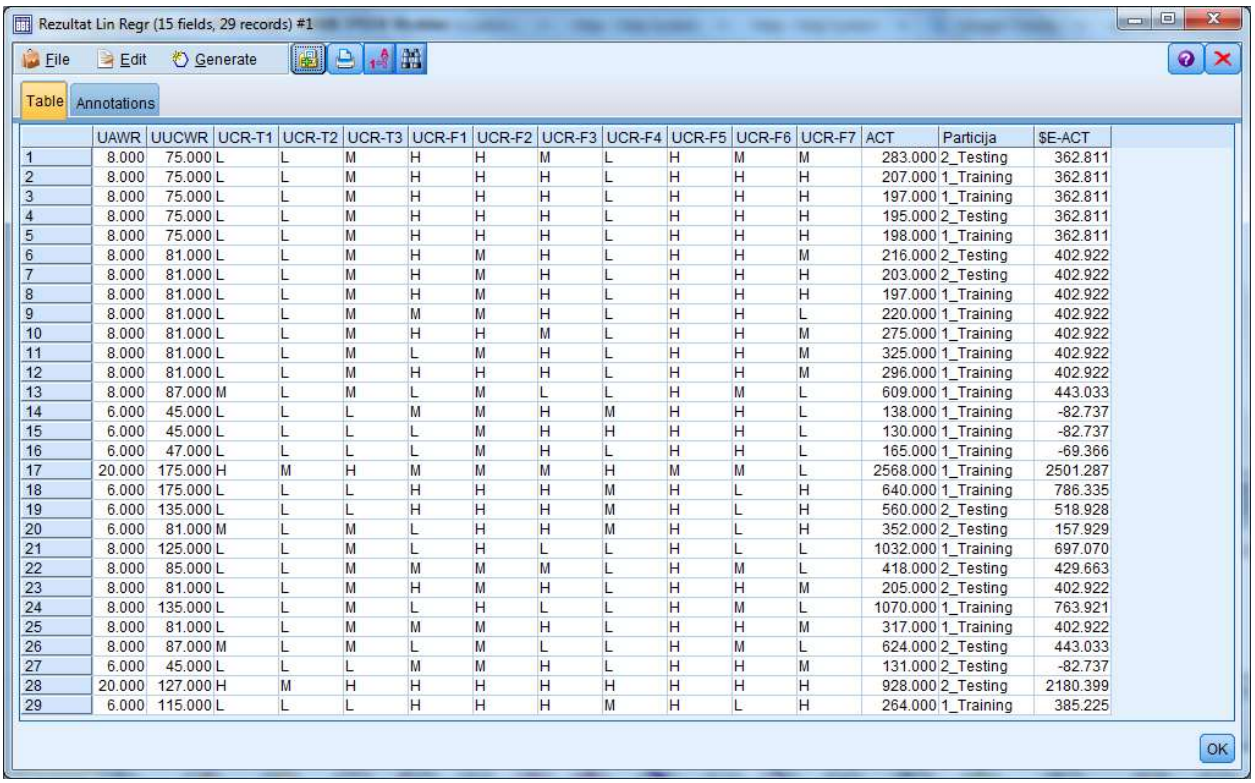
Model	R	R^2
1	0.947[a]	0.896

a. Prediktori: (konstante), UUCW, UAWR

Linearni korelacijski koeficijent izražen je oznakom R, dok je R^2 korelacijski koeficijent. R^2 ima vrijednost između 0 i 1; veći R znači veću snagu modela u objašnjavanju regresijske funkcije i dakle, bolju predikciju zavisne varijable.

Rezultati procjene količine posla linearno regresijskim modelom prikazani su u tablici 6-3, polje $E-ACT.

Tablica 6-3 Procjena količine posla uporabom linearno regresijskog modela



Rezultat Lin Regr (15 fields, 29 records) #1

File Edit Generate

Table Annotations

	UAWR	UUCWR	UCR-T1	UCR-T2	UCR-T3	UCR-F1	UCR-F2	UCR-F3	UCR-F4	UCR-F5	UCR-F6	UCR-F7	ACT	Particija	$E-ACT
1	8.000	75.000	L	L	M	H	H	M	L	H	M	M	283.000	2_Testing	362.811
2	8.000	75.000	L	L	M	H	H	H	L	H	H	H	207.000	1_Training	362.811
3	8.000	75.000	L	L	M	H	H	H	L	H	H	H	197.000	1_Training	362.811
4	8.000	75.000	L	L	M	H	H	H	L	H	H	H	195.000	2_Testing	362.811
5	8.000	75.000	L	L	M	H	H	H	L	H	H	H	198.000	1_Training	362.811
6	8.000	81.000	L	L	M	H	M	H	L	H	H	M	216.000	2_Testing	402.922
7	8.000	81.000	L	L	M	H	M	H	L	H	H	H	203.000	2_Testing	402.922
8	8.000	81.000	L	L	M	H	M	H	L	H	H	H	197.000	1_Training	402.922
9	8.000	81.000	L	L	M	M	M	H	L	H	H	L	220.000	1_Training	402.922
10	8.000	81.000	L	L	M	H	H	M	L	H	H	M	275.000	1_Training	402.922
11	8.000	81.000	L	L	M	L	M	H	L	H	H	M	325.000	1_Training	402.922
12	8.000	81.000	L	L	M	H	H	H	L	H	H	M	296.000	1_Training	402.922
13	8.000	87.000	M	L	M	L	M	L	L	H	M	L	609.000	1_Training	443.033
14	6.000	45.000	L	L	L	M	M	H	M	H	H	L	138.000	1_Training	-82.737
15	6.000	45.000	L	L	L	L	M	H	H	H	H	L	130.000	1_Training	-82.737
16	6.000	47.000	L	L	L	L	M	H	L	H	H	L	165.000	1_Training	-69.366
17	20.000	175.000	H	M	H	M	M	M	H	M	M	L	2568.000	1_Training	2501.287
18	6.000	175.000	L	L	L	H	H	H	M	H	L	H	640.000	1_Training	786.335
19	6.000	135.000	L	L	L	H	H	H	M	H	L	H	560.000	2_Testing	518.928
20	6.000	81.000	M	L	M	L	H	H	M	H	L	H	352.000	2_Testing	157.929
21	8.000	125.000	L	L	M	L	H	L	L	H	L	L	1032.000	1_Training	697.070
22	8.000	85.000	L	L	M	M	M	M	L	H	M	L	418.000	2_Testing	429.663
23	8.000	81.000	L	L	M	H	M	H	L	H	H	M	205.000	2_Testing	402.922
24	8.000	135.000	L	L	M	L	H	L	L	H	M	L	1070.000	1_Training	763.921
25	8.000	81.000	L	L	M	M	M	H	L	H	H	M	317.000	1_Training	402.922
26	8.000	87.000	M	L	M	L	M	L	L	H	M	L	624.000	2_Testing	443.033
27	6.000	45.000	L	L	L	M	M	H	L	H	H	M	131.000	2_Testing	-82.737
28	20.000	127.000	H	M	H	H	H	H	H	H	H	H	928.000	2_Testing	2180.399
29	6.000	115.000	L	L	L	H	H	H	M	H	L	H	264.000	1_Training	385.225

OK

2) Umjetne neuronske mreže

Unutar alata SPSS moguće je postaviti dvije vrste umjetnih neuronskih mreža s obzirom na zadani algoritam: višeslojnu perceptron mrežu bez povratnih veza (*Multilayer perceptron* - MLP) te radijalnu mrežu. Oba algoritma svrstavaju se u nadzirani tip učenja. Mreža bez povratnih veza najjednostavnija je umjetna neuronska mreža. Višeslojne mreže imaju osim ulaznog i izlaznog sloja i jedan ili više skrivenih slojeva neurona.

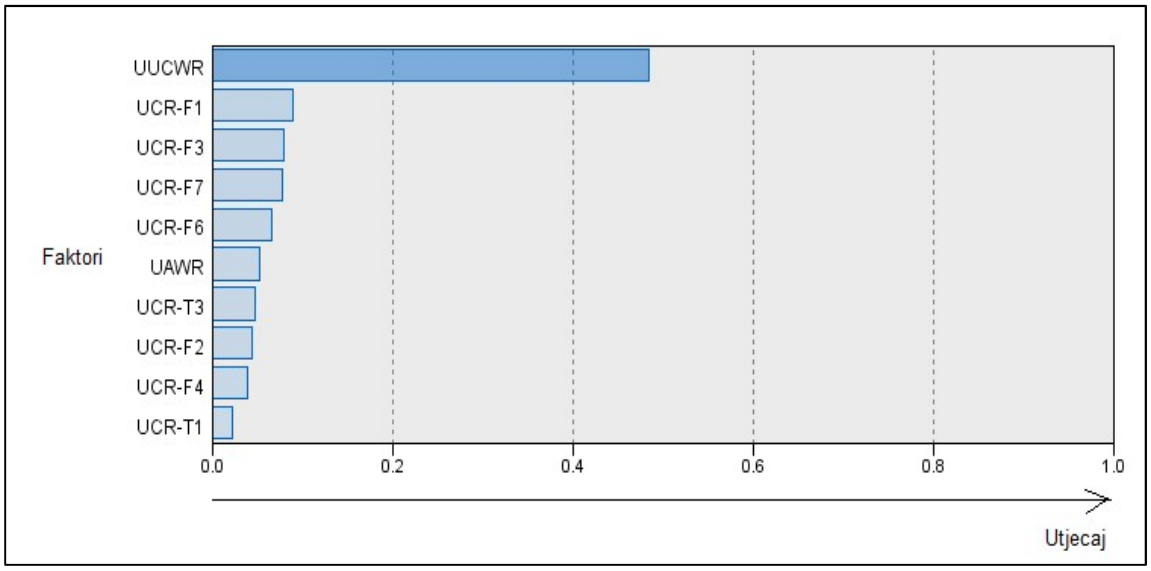
Koriste se za rješavanje širokog spektra problema gdje se učenje pod nadzorom odvija pomoću algoritma s povratnom propagacijom pogreške (*back propagation* – BP). Kod višeslojnih mreža koje koriste algoritam BP učenje se interpretira kao problem optimizacije (minimizacije srednje kvadratne pogreške).

U radijalnim mrežama učenje se interpretira kao problem aproksimacije funkcije s više argumenata. Osnovna radijalna mreža ima tri sloja: ulazni sloj, skriveni sloj te izlazni

sloj. Transformacija od ulaznog do skrivenog sloja je nelinearna, a transformacija od skrivenog do izlaznog sloja je linearna.

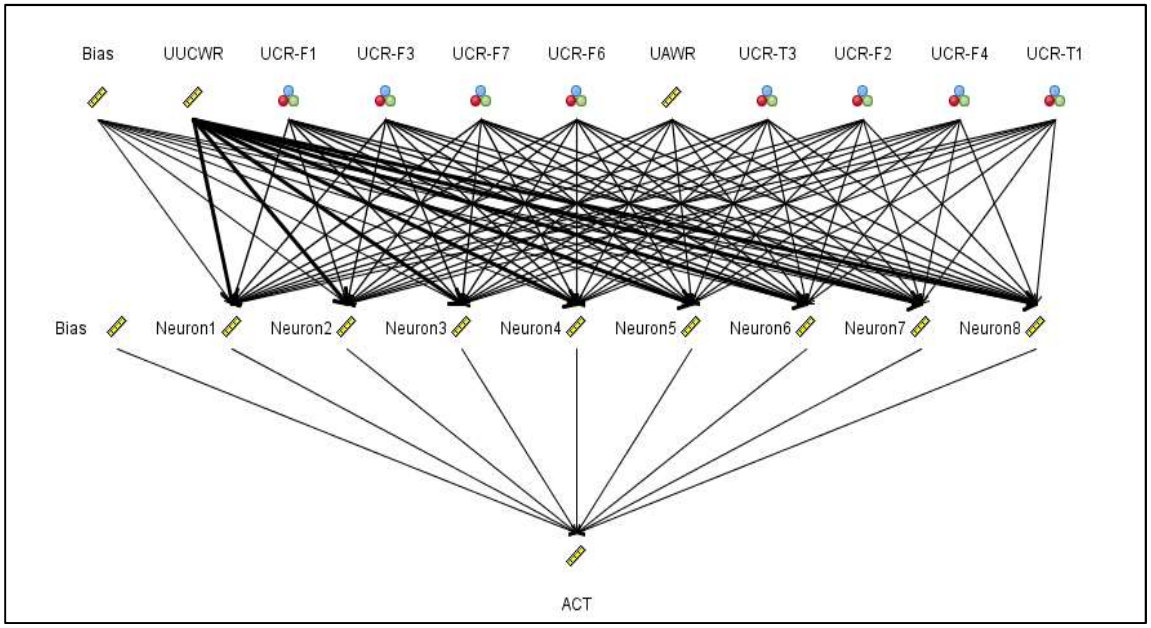
Višeslojna perceptron mreža

U mreži su za ulazna polja postavljeni svi ulazni faktori modela UCR osim UCR-T2 i UCR-F5 (za faktore UCR-T2 i UCR-F5 nije bilo dovoljno različitih vrijednosti za dostupne povijesne podatke). Na slici 6-5 prikazan je utjecaj svakog od faktora na traženi rezultat količine posla unutar neuronske mreže: vrijednost faktora UUCWR ima najveći utjecaj (najveću težinu), a faktor UCR-T1 ima najmanji utjecaj (najmanju težinu) u računanju procijenjene količine posla.



Slika 6-5 Utjecaj faktora na rezultat višeslojne perceptron neuronske mreže

Slika 6-6 prikazuje kreiranu neuronsku mrežu. Mreža ima ukupno 10 ulaznih polja, jedan skriveni sloj s 8 neurona te jedan izlaz (stvarna količina posla). Funkcija prijenosa se primjenjuje radi određivanja izlaza neurona skrivenog sloja te se najčešće koriste sigmoidne aktivacijske funkcije. U primjeru ove višeslojne mreže koristi se aktivacijska funkcija tangens hiperbolni.



Slika 6-6 Model višeslojne neuronske mreže bez povratne veze

Rezultati procjene količine posla višeslojnim neuronskim mrežama bez povratnih veza prikazani su u tablici 6-4, polje $N-ACT.

Tablica 6-4 Procjena količine posla uporabom višeslojnih neuronskih mreža bez povratnih veza

Rezultat n. m. MLP (15 fields, 29 records)

File Edit Generate

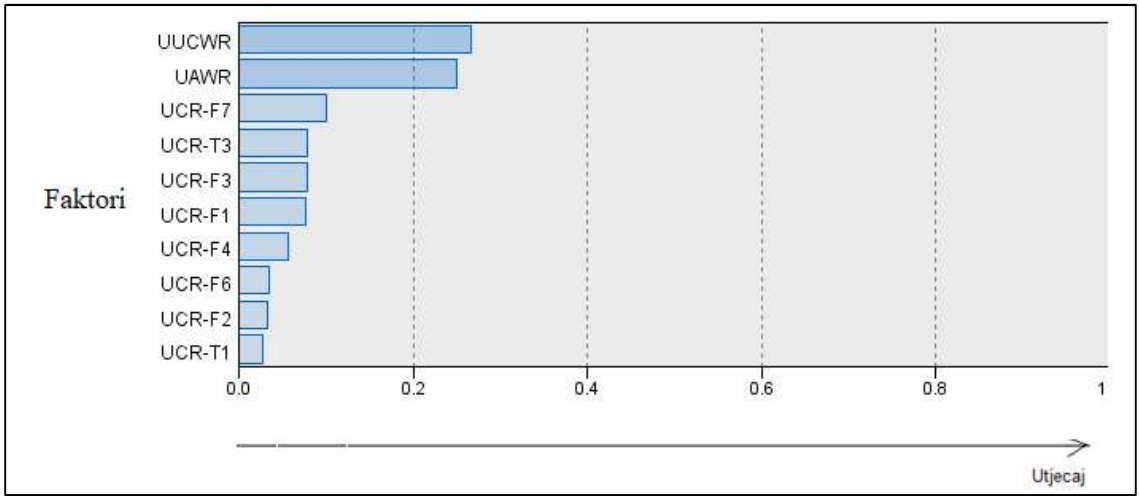
Table Annotations

	UAWR	UUCWR	UCR-T1	UCR-T2	UCR-T3	UCR-F1	UCR-F2	UCR-F3	UCR-F4	UCR-F5	UCR-F6	UCR-F7	ACT	Particija	$N-ACT
1	8.000	75.000	L	L	M	H	H	M	L	H	M	M	283.000	2_Testing	465.467
2	8.000	75.000	L	L	M	H	H	H	L	H	H	H	207.000	1_Training	243.949
3	8.000	75.000	L	L	M	H	H	H	L	H	H	H	197.000	1_Training	243.949
4	8.000	75.000	L	L	M	H	H	H	L	H	H	H	195.000	2_Testing	243.949
5	8.000	75.000	L	L	M	H	H	H	L	H	H	H	198.000	1_Training	243.949
6	8.000	81.000	L	L	M	H	M	H	L	H	H	M	216.000	2_Testing	293.962
7	8.000	81.000	L	L	M	H	M	H	L	H	H	H	203.000	2_Testing	258.977
8	8.000	81.000	L	L	M	H	M	H	L	H	H	H	197.000	1_Training	258.977
9	8.000	81.000	L	L	M	M	M	H	L	H	H	L	220.000	1_Training	300.902
10	8.000	81.000	L	L	M	H	H	M	L	H	H	M	275.000	1_Training	351.014
11	8.000	81.000	L	L	M	L	M	H	L	H	H	M	325.000	1_Training	346.261
12	8.000	81.000	L	L	M	H	H	H	L	H	H	M	296.000	1_Training	328.960
13	8.000	87.000	M	L	M	L	M	L	L	H	M	L	609.000	1_Training	538.664
14	6.000	45.000	L	L	L	M	M	H	M	H	H	L	138.000	1_Training	142.027
15	6.000	45.000	L	L	L	L	M	H	H	H	H	L	130.000	1_Training	$null$
16	6.000	47.000	L	L	L	L	M	H	L	H	H	L	165.000	1_Training	246.525
17	20.000	175.000	H	M	H	M	M	M	H	M	M	L	2568.000	1_Training	$null$
18	6.000	175.000	L	L	L	H	H	H	M	H	L	H	640.000	1_Training	694.190
19	6.000	135.000	L	L	L	H	H	H	M	H	L	H	560.000	2_Testing	454.330
20	6.000	81.000	M	L	M	L	H	H	M	H	L	H	352.000	2_Testing	356.551
21	8.000	125.000	L	L	M	L	H	L	L	H	L	L	1032.000	1_Training	982.072
22	8.000	85.000	L	L	M	M	M	M	L	H	M	L	418.000	2_Testing	391.654
23	8.000	81.000	L	L	M	H	M	H	L	H	H	M	205.000	2_Testing	293.962
24	8.000	135.000	L	L	M	L	H	L	L	H	M	L	1070.000	1_Training	1031.832
25	8.000	81.000	L	L	M	M	M	H	L	H	H	M	317.000	1_Training	245.798
26	8.000	87.000	M	L	M	L	M	L	L	H	M	L	624.000	2_Testing	538.664
27	6.000	45.000	L	L	L	M	M	H	L	H	H	M	131.000	2_Testing	82.637
28	20.000	127.000	H	M	H	H	H	H	H	H	H	H	928.000	2_Testing	$null$
29	6.000	115.000	L	L	L	H	H	H	M	H	L	H	264.000	1_Training	364.170

OK

Radijalne neuronske mreže

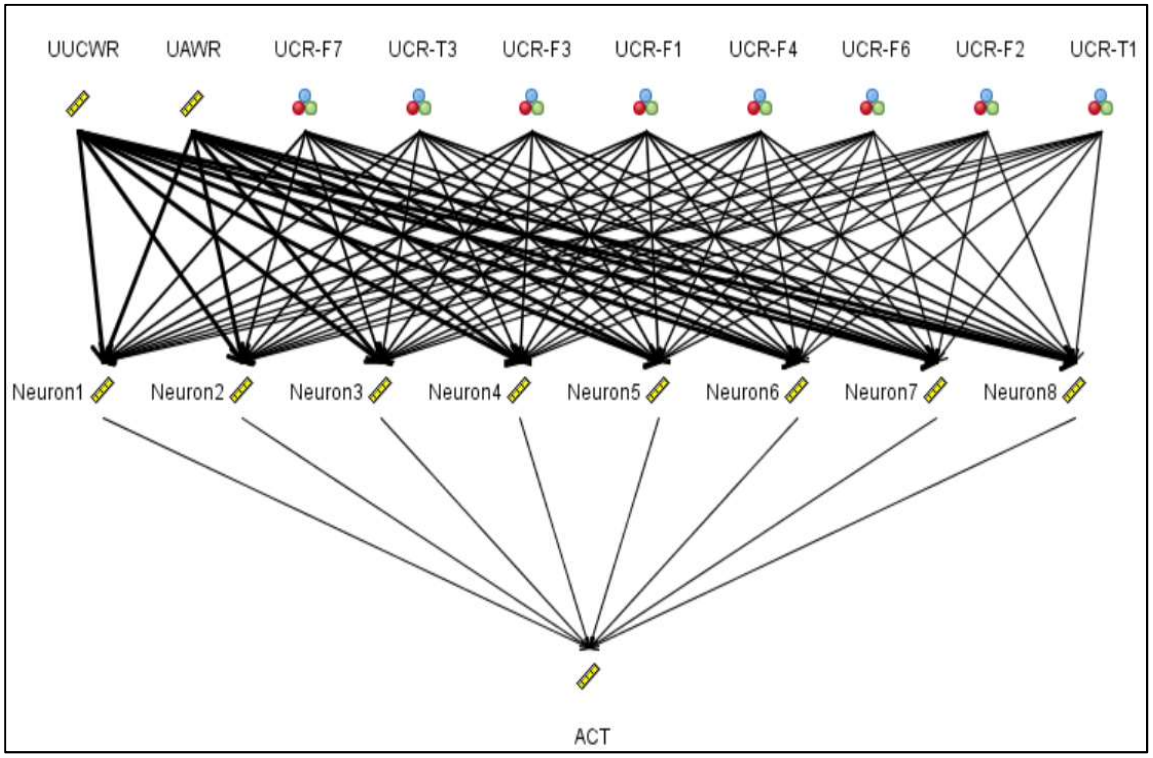
U mreži su za ulazna polja postavljeni svi ulazni faktori modela UCR osim UCR-T2 i UCR-F5 (za faktore UCR-T2 i UCR-F5 nije bilo dovoljno različitih vrijednosti za dostupne povijesne podatke). Na slici 6-7 prikazan je utjecaj svakog od faktora na traženi rezultat količine posla unutar neuronske mreže: vrijednost faktora UUCWR ima najveći utjecaj (najveću težinu), a faktor UCR-T1 ima najmanji utjecaj (najmanju težinu) u računanju procijenjene količine posla.



Slika 6-7 Utjecaj faktora na rezultat radijalne neuronske mreže

Mreža ima ukupno 10 ulaznih polja, jedan skriveni sloj s 8 neurona te jedan izlaz (stvarna količina posla). Radijalna funkcija prijenosa ove mreže je *softmax* funkcija (koja se postavlja kod normaliziranja sume skrivenih slojeva do vrijednosti 1).

Slika 6-8 Model radijalne neuronske mreže



Rezultati procjene količine posla radijalnim neuronskim mrežama prikazani su u tablici 6-5, polje $N-ACT.

Tablica 6-5 Procjena količine posla uporabom radijalnih neuronskih mreža

Rezultat n.m.RBF (15 fields, 29 records)

File Edit Generate

Table Annotations

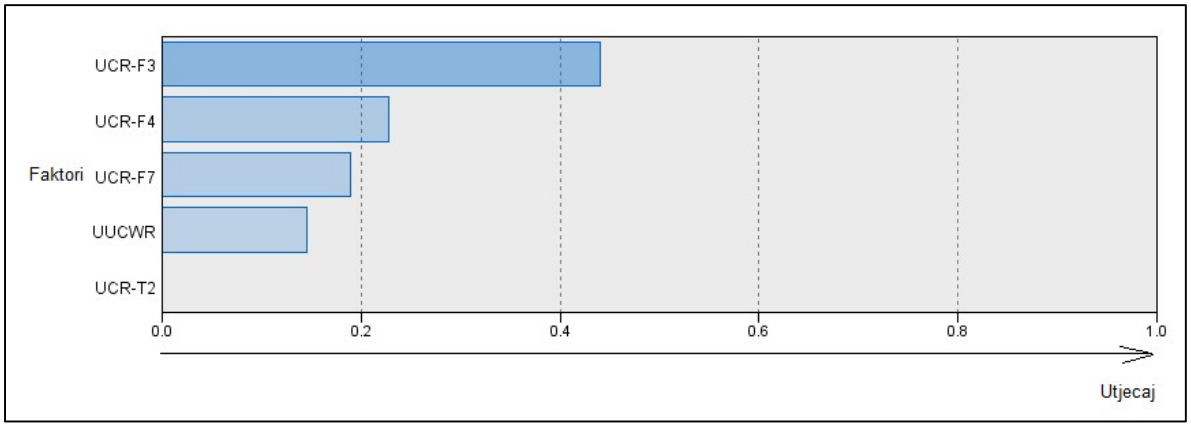
	UAWR	UUCWR	UCR-T1	UCR-T2	UCR-T3	UCR-F1	UCR-F2	UCR-F3	UCR-F4	UCR-F5	UCR-F6	UCR-F7	ACT	Particija	$N-ACT
1	8.000	75.000	L	L	M	H	H	M	L	H	M	M	283.000	2_Testing	275.000
2	8.000	75.000	L	L	M	H	H	H	L	H	H	H	207.000	1_Training	199.750
3	8.000	75.000	L	L	M	H	H	H	L	H	H	H	197.000	1_Training	199.750
4	8.000	75.000	L	L	M	H	H	H	L	H	H	H	195.000	2_Testing	199.750
5	8.000	75.000	L	L	M	H	H	H	L	H	H	H	198.000	1_Training	199.750
6	8.000	81.000	L	L	M	H	M	H	L	H	H	M	216.000	2_Testing	324.996
7	8.000	81.000	L	L	M	H	M	H	L	H	H	H	203.000	2_Testing	199.750
8	8.000	81.000	L	L	M	H	M	H	L	H	H	H	197.000	1_Training	199.750
9	8.000	81.000	L	L	M	M	M	H	L	H	H	L	220.000	1_Training	220.000
10	8.000	81.000	L	L	M	H	H	M	L	H	H	M	275.000	1_Training	275.000
11	8.000	81.000	L	L	M	L	M	H	L	H	H	M	325.000	1_Training	325.000
12	8.000	81.000	L	L	M	H	H	H	L	H	H	M	296.000	1_Training	258.610
13	8.000	87.000	M	L	M	L	M	L	L	H	M	L	609.000	1_Training	609.000
14	6.000	45.000	L	L	L	M	M	H	M	H	H	L	138.000	1_Training	138.000
15	6.000	45.000	L	L	L	L	M	H	H	H	H	L	130.000	1_Training	$null$
16	6.000	47.000	L	L	L	L	M	H	L	H	H	L	165.000	1_Training	140.848
17	20.000	175.000	H	M	H	M	M	M	H	M	M	L	2568.000	1_Training	$null$
18	6.000	175.000	L	L	L	H	H	H	M	H	L	H	640.000	1_Training	452.000
19	6.000	135.000	L	L	L	H	H	H	M	H	L	H	560.000	2_Testing	452.000
20	6.000	81.000	M	L	M	L	H	H	M	H	L	H	352.000	2_Testing	452.000
21	8.000	125.000	L	L	M	L	H	L	L	H	L	L	1032.000	1_Training	1070.000
22	8.000	85.000	L	L	M	M	M	M	L	H	M	L	418.000	2_Testing	220.001
23	8.000	81.000	L	L	M	H	M	H	L	H	H	M	205.000	2_Testing	324.996
24	8.000	135.000	L	L	M	L	H	L	L	H	M	L	1070.000	1_Training	1070.000
25	8.000	81.000	L	L	M	M	M	H	L	H	H	M	317.000	1_Training	272.500
26	8.000	87.000	M	L	M	L	M	L	L	H	M	L	624.000	2_Testing	609.000
27	6.000	45.000	L	L	L	M	M	H	L	H	H	M	131.000	2_Testing	139.303
28	20.000	127.000	H	M	H	H	H	H	H	H	H	H	928.000	2_Testing	$null$
29	6.000	115.000	L	L	L	H	H	H	M	H	L	H	264.000	1_Training	452.000

OK

3) CHAID metoda (*Chi-square Automatic Interaction Detection*)

Metoda generira stabla odluke koristeći hi-kvadrat test za prepoznavanje optimalnih raspada. CHAID može generirati ne-binarna stabla, što znači da neki dijelovi imaju više od dvije grane. Rezultat i ulazni podaci mogu biti numerički rasponi (kontinuirani) ili kategorički.

Manji broj tehničkih faktora i faktora okoline (UUCWR, UCR-T2, UCR-F3, UCR-F4 i UCR-F7) koriste se u modelu CHAID. Slika 6-8 prikazuje utjecaj (od većeg prema manjem) svakog od faktora na traženi rezultat unutar modela CHAID: vrijednost faktora UCR-F3 ima najveći utjecaj (najveću težinu), a faktor UCR-T2 ima najmanji utjecaj (najmanju težinu) u računanju procijenjene količine posla.



Slika 6-9 Utjecaj faktora na rezultat modela CHAID

Rezultati procjene količine posla modelom CHAID prikazani su u tablici 6-6, polje $R-ACT.

Tablica 6-6 Procjena količine posla uporabom modela CHAID

Rezultat CHAID (15 fields, 29 records)

File Edit Generate

Table Annotations

	UAWR	UUCWR	UCR-T1	UCR-T2	UCR-T3	UCR-F1	UCR-F2	UCR-F3	UCR-F4	UCR-F5	UCR-F6	UCR-F7	ACT	Particija	$R-ACT
1	8.000	75.000	L	L	M	H	H	M	L	H	M	M	283.000	2_Testing	200.667
2	8.000	75.000	L	L	M	H	H	H	L	H	H	H	207.000	1_Training	200.667
3	8.000	75.000	L	L	M	H	H	H	L	H	H	H	197.000	1_Training	200.667
4	8.000	75.000	L	L	M	H	H	H	L	H	H	H	195.000	2_Testing	200.667
5	8.000	75.000	L	L	M	H	H	H	L	H	H	H	198.000	1_Training	200.667
6	8.000	81.000	L	L	M	H	M	H	L	H	H	M	216.000	2_Testing	303.250
7	8.000	81.000	L	L	M	H	M	H	L	H	H	H	203.000	2_Testing	227.000
8	8.000	81.000	L	L	M	H	M	H	L	H	H	H	197.000	1_Training	227.000
9	8.000	81.000	L	L	M	M	M	H	L	H	H	L	220.000	1_Training	227.000
10	8.000	81.000	L	L	M	H	H	M	L	H	H	M	275.000	1_Training	303.250
11	8.000	81.000	L	L	M	L	M	H	L	H	H	M	325.000	1_Training	303.250
12	8.000	81.000	L	L	M	H	H	H	L	H	H	M	296.000	1_Training	303.250
13	8.000	87.000	M	L	M	L	M	L	L	H	M	L	609.000	1_Training	609.000
14	6.000	45.000	L	L	L	M	M	H	M	H	H	L	138.000	1_Training	138.000
15	6.000	45.000	L	L	L	L	M	H	H	H	H	L	130.000	1_Training	130.000
16	6.000	47.000	L	L	L	L	M	H	L	H	H	L	165.000	1_Training	165.000
17	20.000	175.000	H	M	H	M	M	M	H	M	M	L	2568.000	1_Training	2568.000
18	6.000	175.000	L	L	L	H	H	H	M	H	L	H	640.000	1_Training	640.000
19	6.000	135.000	L	L	L	H	H	H	M	H	L	H	560.000	2_Testing	640.000
20	6.000	81.000	M	L	M	L	H	H	M	H	L	H	352.000	2_Testing	227.000
21	8.000	125.000	L	L	M	L	H	L	L	H	L	L	1032.000	1_Training	1032.000
22	8.000	85.000	L	L	M	M	M	M	L	H	M	L	418.000	2_Testing	227.000
23	8.000	81.000	L	L	M	H	M	H	L	H	H	M	205.000	2_Testing	303.250
24	8.000	135.000	L	L	M	L	H	L	L	H	M	L	1070.000	1_Training	1070.000
25	8.000	81.000	L	L	M	M	M	H	L	H	H	M	317.000	1_Training	303.250
26	8.000	87.000	M	L	M	L	M	L	L	H	M	L	624.000	2_Testing	609.000
27	6.000	45.000	L	L	L	M	M	H	L	H	H	M	131.000	2_Testing	165.000
28	20.000	127.000	H	M	H	H	H	H	H	H	H	H	928.000	2_Testing	2568.000
29	6.000	115.000	L	L	L	H	H	H	M	H	L	H	264.000	1_Training	227.000

OK

Evaluacija modela

Oblikovanje modela je iterativni proces, stoga su testirani rezultati nekoliko modela prije donošenja konačne odluke o konačnom modelu koji će se primjenjivati u budućnosti.

Preciznost modela UCR_ML mjerena je pomoću MMRE (jednadžba 5.1) i PRED(20).

Rezultati MMRE svake metode prikazani su u tablici 6-7.

Tablica 6-7 Evaluacija metoda strojnog učenja na testnom skupu podataka

Linearna regresija	Testni skup projekata										
Projekti	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11
MRE (UCR_ML)	28%	86%	87%	98%	7%	55%	3%	97%	29%	163%	135%
MMRE (UCR)	71.65%										
MdMRE (UCR)	51.97%										
PRED(20) (UCR)	18%										

Neuronske mreže MLP	Testni skup projekata										
Projekti	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11
MRE (UCR_ML)	64%	25%	36%	28%	19%	1%	6%	43%	14%	37%	N/A
MMRE (UCR)	27.37%										
MdMRE (UCR)	18.83%										
PRED(20) (UCR)	36%										

Neuronske mreže Radijalne	Testni skup projekata										
Projekti	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11
MRE (UCR_ML)	3%	2%	50%	2%	19%	28%	47%	59%	2%	6%	N/A
MMRE (UCR)	21.80%										
MdMRE (UCR)	22.70%										
PRED(20) (UCR)	55%										

CHAID	Testni skup projekata										
Projekti	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11
MRE (UCR_ML)	29%	3%	40%	12%	14%	36%	46%	48%	2%	26%	177%
MMRE (UCR)	39.34%										
MdMRE (UCR)	48.34%										
PRED(20) (UCR)	36%										

Među promatranim metodama strojnog učenja, najbolje (najpreciznije) rezultate procjene količine posla je postigla metoda radijalnih neuronskih mreža; za tu metodu vrijednost MMRE je najmanja, a PRED(20) ima najveću vrijednost.

Kao i kod razvoja modela UCR, pri oblikovanju modela UCR_ML primjetan je mali broj dostupnih projekata s povijesnim podacima, što predstavlja prijetnju valjanosti. Za modele koji se temelje na strojnom učenju veliki (signifikantni) broj povijesnih podataka bitan je preduvjet za razvoj modela s preciznim predviđanjem.

Veći broj projekata s povijesnim podacima poboljšao bi preciznost modela, što posebice vrijedi za modele zasnovane na strojnom učenju.

Primjena modela

Nakon oblikovanja modela i dalje je unutar poslovne organizacije potrebno prikupljati nove podatke kojima će se postojeći model nadograđivati. Za razliku od algoritamskog modela, modeli procjene temeljeni na strojnom učenju imaju mogućnosti unaprjeđenja kroz vrijeme unosom novih povijesnih podataka s obzirom da mu se težine faktora dinamički prilagođavaju unutar odabrane metode strojnog učenja.

U primjeni modela UCR_ML preporuka je proširivati skup podataka za učenje novim povijesnim podacima kako bi se preciznost procijenjene količine posla povećala s većim brojem zapisa (povijesni podaci jednog projekta predstavljaju jedan zapis podataka).

Za korištenje modela UCR_ML procjenitelji trebaju postaviti sve ulazne faktore za svaki projekt, što uključuje :

- broj aktera i njihova klasifikacija prema kriteriju složenosti za izračun UAW_R ,

- broj slučajeva uporabe i njihova klasifikacija po složenosti i kriteriju iskoristivosti, za izračun $UUCW_R$,

- tehničke faktore koji se odnose na programski proizvod, za postavljanje atributa faktora UCR-T1 do UCR-T3,

- faktore okoline koji se odnose na projektni tim, za postavljanje atributa faktora UCR-F1 do UCR-F7.

Ako je potrebno, stručni tim može prilagoditi model UCR za svoje specifično okruženje slijedeći korake:

1) Modifikacija ulaznih faktora u predloženom modelu UCR_ML od strane stručnog tima. Pri velikoj količini povijesnih podataka, moguće je postaviti vrijednost svakog ulaznog faktora modela UCR. Umjesto izračunate vrijednosti UAW_R, moguće je definirati zasebne ulazne faktore za svaki tip aktera te postaviti njihove količine kao vrijednost ulaznog faktora. Prema istom principu, umjesto izračunate vrijednosti $UUCW_{R,}$ moguće je definirati zasebne ulazne faktore za svaki tip slučajeva uporabe (za svaku klasifikaciju) te postaviti njihove količine kao vrijednost ulaznog faktora.
2) Potencijalno dodavanje ili uklanjanje ulaznih faktora; za druge projektne organizacije moguće je da postoje drugi čimbenici koji imaju znatan utjecaj na količinu posla, a koji bi se mogli ugraditi u model procjene kroz novi ulazni faktor.

6.5. Validacija modela procjene UCR_ML

U sklopu validacijskog procesa, uspoređuju se rezultati dvaju modela: UCR i UCR_ML. Od opisanih metoda strojnog učenja u UCR_ML, najpreciznije rezultate dao je model radijalnih neuronskih mreže te će se ti rezultati koristiti u narednoj usporedbi. Provode se koraci eksperimenta opisani u poglavlju 5.2. U sklopu studijskog primjera iz prethodnih poglavlja, modeli UCR i UCR_ML primjenjuju se na inicijalnim i slijednim projektima; faktor u ovom eksperimentu je model procjene, dok su postupci postavljeni u okviru modela UCR i UCR_ML. Subjekti i objekti za validaciju su isti kao i u eksperimentu provedenom u poglavlju 5.2. s obzirom da se validacija provodi na istom skupu povijesnih projektnih podataka za validaciju.

Točnost modela procjene promatra se pomoću MRE (veličina relativne greške). MRE se računa pomoću jednadžbe 2.1.

U sklopu eksperimenta definira se sljedeća hipoteza te mjerenja nužna za evaluaciju hipoteze:

1. Nul-hipoteza, H_0: Ne postoji razlika u točnosti procijenjene količine posla (mjerene pomoću MRE) pri primjeni dvaju modela procjene – UCR i UCR_ML.
   H_0: *MRE (UCR) = MRE (UCR_ML)*
   Alternativna hipoteza:

   H_1: *MRE (UCR) ≠ MRE (UCR_ML)*

Procijenjene količine posla modelima UCR i UCR_ML, te stvarna količina promatranih projekata su neovisne varijable unutar eksperimenta. Zavisna varijabla je MRE. Povijesni podaci projekata sadrže vrijednosti nezavisnih varijabli za ispitivanje hipoteze:

- procijenjena količina posla modelima UCR i UCR_ML i stvarna količina posla promatranih projekata izražene su u veličini čovjek – sat, mjereno na omjernoj skali (*ratio scale*).

Za svaki subjekt korištena su oba postupka (oba modela UCR i UCR_ML), što je temelj za korištenje eksperimenta s uparenim dizajnom [56] za testiranje hipoteze. Upareni dizajn analizira se uparenim t-testom.

Relativno mali broj dostupnih projekata za validaciju modela UCR_ML (mali broj inicijalnih i slijednih projekata) smatra se vanjskom prijetnjom za valjanost rezultata.

Prikupljanje podataka

Povijesni podaci projekata prikupljeni su na dva načina: (1) za model UCR poznati su iz ranije provedenog eksperimenta. Za model UCR_ML korišteni su rezultati modela oblikovanog u alatu SPSS.

Parametrijsko testiranje (uparen t-test)

Uparen t-test proveden je prema izračunu u tablici 5-1 [56].

MRE količine posla projekata u validacijskom skupu, procijenjene prema modelima UCR i UCR_ML, prikazane su u tablici 6-8. Promatraju se podaci onih projekata koji su istovremeno dio skupa za validaciju modela UCR te dodijeljeni skupu za testiranje modela UCR_ML. Oba uvjeta su ispunjena za sljedeće projekte: P1, P2, P4, P5, P8 i P9.

Tablica 6-8 MRE projekata prema modelima UCR i UCR_ML

	Skup podataka za validaciju					
	P1	P2	P4	P5	P8	P9
MRE (UCR)	15.80%	1.20%	2.40%	9.00%	6.60%	3.90%
MRE (UCR_ML)	19.30%	28.40%	47.40%	58.50%	2.40%	6.30%

MMRE (UCR)	7.00%
MdMRE (UCR)	5.37%

MMRE (UCR_ML)	27.05%
MdMRE (UCR_ML)	22.38%

Nul-hipoteza navodi da nema razlike u procjeni točnosti (mjereno pomoću MRE) procijenjene količine posla pri primjeni modela UCP i UCR. Alternativa hipoteza navodi da postoji razlika.

Rezultati ispitivanja uparenih uzoraka prikazani su u tablici 6-9.

Tablica 6-9 Validacija modela UCR_ML prema uparenom t-testu

Paired Samples Test

	Paired Differences					t	df	Sig. (2-tailed)
	Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the ... Lower	Upper			
UCR MRE - UCR_ML MRE	-.199849	.231472	.094498	-.442764	.04307	-2	5	.088

Broj stupnjeva slobode je $f = n - 1 = 6 - 1 = 5$, $t0 = -2$ i $t0.025,5 = 2.571$ [31]. S obzirom da je $t0 > t0.025,5$ moguće je odbaciti nul-hipotezu.

Ograničen broj projekata (uzoraka) u sklopu provjere valjanosti može predstavljati vanjsku prijetnju valjanosti. Međutim, prema rezultatima uparenih t-testova, moguće je odbiti nul-hipotezu.

Rezultati eksperimenta

Odbijanjem nul-hipoteze, eksperiment pokazuje da postoji razlika u točnosti procjene (mjereno pomoću MRE), procijenjene količine posla pri primjeni modela UCR i UCR_ML.

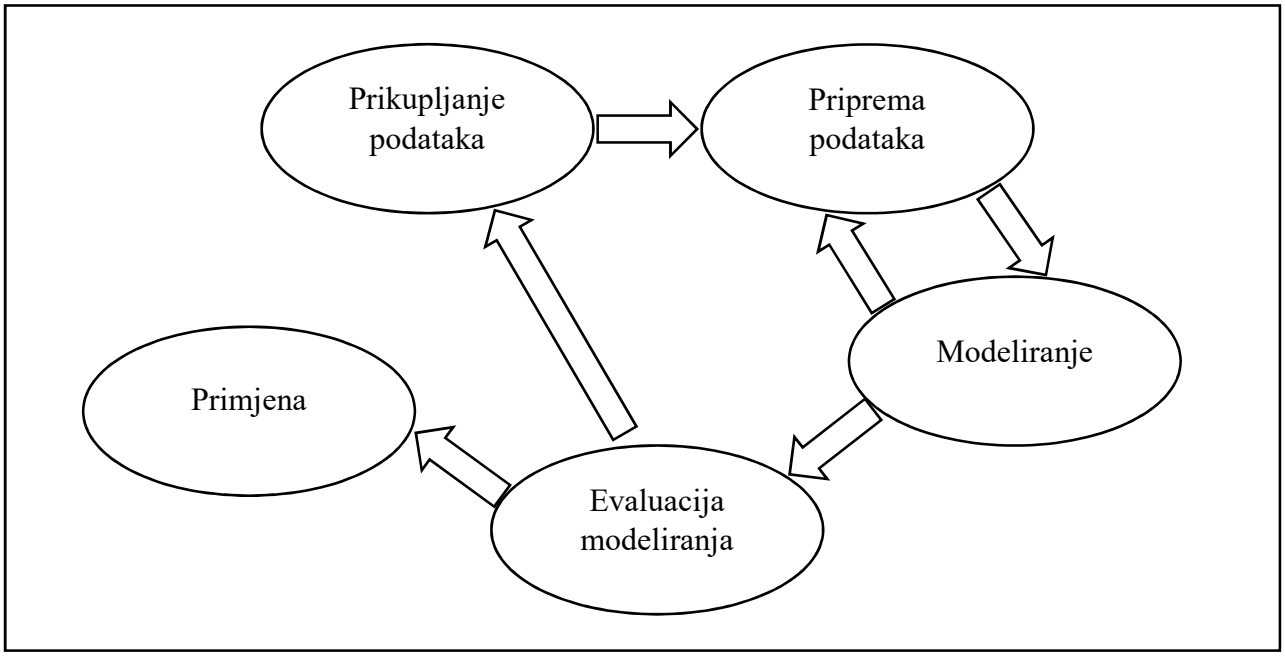
Točnost modela procjene također je promatrana ovim kriterijima: *MMRE* i *PRED(20).*

Sve vrijednosti veličine relativnih grešaka koriste se za izračun vrijednosti *MMRE* jednadžbi 5.1.

Obje mjere, *MMRE* i *PRED* (20) pokazuju bolje rezultate za model UCR.

6.6. Prednosti i nedostaci razvoja modela za procjenu količine posla zasnovanog na strojnom učenju

Oblikovanje modela za procjenu količine posla zasnovanog na strojnom učenju treba promatrati kao iterativni proces; za precizniju procjenu nužno ga je s vremenom nadopunjavati novim povijesnim podacima i ispitivati vezu između podataka za obradu i primjerene metode temeljene na strojnom učenju (Slika 6-9).



Slika 6-10 Proces oblikovanja modela za procjenu količine posla temeljenog na strojnom učenju

Metode zasnovane na strojnom učenju omogućuju dinamičku prilagodbu parametara modela te pritom nije potrebno razvijati matematičke modele kao što je slučaj pri algoritamskim metodama.

U sklopu studijskog primjera (opisanog u poglavlju 6.4) model zasnovan na strojnom učenju bilježi nešto slabije rezultate od algoritamskog modela UCR. Pod prijetnjom valjanosti rezultata zabilježen je relativno mali skup povijesnih podataka pa su posljedično mali skupovi podataka za učenje i testiranje. Može se zaključiti da su za organizacije s ograničavajućim brojem projekata sa sličnim obimom posla, a koje žele razviti sebi pogodan model procjene količine posla, primjereniji algoritamski modeli. Međutim, organizacije s većim brojem povijesnih podataka, opisani proces razvoja modela mogu lako primijeniti i po potrebi prilagoditi ulazne faktore.

7. Zaključak

Jedna od važnih komponenti planiranja i isporuke uspješnog projekta jest i precizna procjena količine posla. Procijenjena količina posla izravno utječe na više aspekata životnog ciklusa razvojnog procesa: utječe na projektne planove, raspored isporuke projektnih artefakata te se koristi kao jedan od ulaznih parametra za određivanje proračuna projekta i tržišne cijene. U praksi je često procijenjena količina posla manja od stvarne količine posla na kraju projekta. Analiza često korištenih tehnika za procjenu pokazala je da su ti modeli primarno namijenjeni razvoju novog programskog rješenja.

Mogućnost korištenja ranije razvijenih artefakata kroz više projekata javlja se kod programskih proizvoda s istim ili vrlo sličnim zahtjevima. Prema trenutnim spoznajama, do sada se niti jedan od modela procjene koji su opisani u literaturi nije bavio aspektima iskoristivosti ranije razvijenih artefakata za potrebe razvoja drugog programskog proizvoda.

Ovim doktorskim istraživanjem predložen je novi model procjene količine posla za razvoj programskih rješenja zasnovan na višestrukom korištenju slučajeva uporabe i značajkama projektnog tima, nazvan model UCR (*Use Case Reusability model*). Model UCR nastao je modifikacijom postojećeg Karnerovog modela za procjenu količine posla UCP [14] te dodavanjem elemenata koji opisuju aspekt višestruke iskoristivosti. Uvedena je nova klasifikacija slučajeva uporabe prema iskoristivosti koja je rezultat zasebnog znanstvenog doprinosa: postupka analize zahtjeva novog programskog proizvoda i specifikacije slučajeva uporabe već isporučenih programskih rješenja s ciljem procjene njihove iskoristivosti pri razvoju novog programskog proizvoda. Novom klasifikacijom slučajevi uporabe dijele se na nove, slične i identične slučajeve uporabe.

Model UCR namijenjen je projektima u kojima se koriste prethodno razvijeni artefakti poput programskog koda i ostale projektne dokumentacije (npr. model dizajna, model podataka, odluke o arhitekturi sustava i testne specifikacije). Razina iskoristivosti drugih artefakata povećava se s brojem identičnih ili sličnih slučajeva uporabe. Slučajevi uporabe se definiraju u ranoj fazi razvojnog projekta i stoga se ovaj model procjene može koristiti već u fazi planiranja projekta. Predloženi model UCR uključuje parametre dobivene od stručnjaka s iskustvom u procjeni

količine posla. Prvo su definirani ulazni faktori modela i njihovi atributi, a potom su metodom *Delphi* određene težine svih faktora (parametri). U konačnici, veličina UCR izražena u čovjek – sat definirana je kalibracijom koristeći povijesne podatke projekata o ulaznim faktorima UCR-a i stvarnoj količini posla. Veličina UCR (čovjek – sat) razlikuje se za inicijalne i slijedne projekte.

Validacijom modela UCR u sklopu empirijskog istraživanja (eksperimenta) evaluirana je procijenjena količina posla iz perspektive projektnog tima u kontekstu industrijskih i akademskih projekata (za ukupno 29 projekata). U sklopu validacijskog procesa, model UCR se primijenio na inicijalnim i slijednim projektima iz tri različita i nepovezana programa. Faktor u eksperimentu je bio model procjene, dok su postupci postavljeni u okviru modela UCP i UCR. Subjekti su bili projektni timovi iz industrije te akademske zajednice. Nul-hipoteza eksperimenta je postavljena da nema razlike u preciznosti procijenjene količine posla za modele UCP i UCR. Alternativa hipoteza navodi da postoji razlika.

Nakon provedbe eksperimenta, primjena modela UCR pokazuje poboljšane rezultate procijenjene količine posla usporedbi s modelom UCP za promatrane projekte unutar studijskog primjera: apsolutne vrijednosti relativnih grešaka (MRE) i srednje veličine relativne greške (MMRE) za model UCR su niže od istih vrijednosti za UCP model. Na temelju rezultata parametrijskog testa, moguće je odbiti nul-hipotezu.

Projekti čiji su se povijesni podaci koristili za kalibraciju i validaciju modela dominantno su mali u smislu obima posla projekata (mali ili umjereni broj slučajeva uporabe, do 22 slučajeva uporabe) i veličine tima (do pet članova tima), tako da bi se mogla očekivati određena odstupanja u procjeni količini posla za srednje ili velike projekte. Kao dio budućeg rada, za projekte srednje ili velike veličine predlaže se proširenje ljestvice atributa faktora u modelu UCR (npr. uvođenjem atributa *vrlo slab* i *vrlo jak*). Veličina projekta i složenost projekta povećavaju se od malih do velikih projekata: projekti srednje i velike veličine imaju umjereno visok broj isporučenih artefakata koji su obično tehnički složeniji, broj članova tima se povećava, a rokovi isporuke povećavaju. Dodatna granularnost atributa omogućila bi preciznije razlikovanje projekata u smislu tehničke složenosti proizvoda i karakteristika projektnog tima.

U sklopu trećeg znanstvenog doprinosa, definiran je model procjene količine posla koji se temelji na metodama strojnog učenja UCR_ML. Pri definiciji algoritamskog modela za procjenu količinu

posla koristi se znanje, odnosno iskustvo prikupljeno u određenom vremenskom periodu. Nakon definicije modela, isti se koristi u daljnjim projektima te ne postoji povratna veza između rezultata njegove primjene u pojedinoj organizaciji te definicije samog modela. Metode zasnovane na strojnom učenju omogućuju dinamičku prilagodbu parametara modela te pritom nije potrebno razvijati matematičke modele kao što je slučaj pri algoritamskim metodama. Odnos između ulaznih faktora, kao i veza između ulaznih faktora i ciljanog rezultata grade se na temelju povijesnih podataka te modeli sami postavljaju težinu pojedinih faktora, odnosno njihov utjecaj na količinu posla. Razvoj modela koji se temelji na metodama strojnog učenja je iterativan proces te se model može unaprjeđivati unosom novih povijesnih podataka u skup podataka za učenje kroz vrijeme.

U sklopu studijskog primjera opisan je proces oblikovanja modela procjene zasnovanog na metodama strojnog učenja. Nakon prikupljanja i obrade povijesnih podataka, isti su podijeljeni u skupove za učenje i treniranje modela. Primjenom regresijskih modela (metode linearne regresije, umjetne neuronske mreže i stablo odlučivanja) oblikovan je model UCR_ML. U sklopu evaluacije rezultata, za dane povijesne podatke najpreciznije rezultate procijenjene količine posla dala je metoda radijalnih neuronskih mreža.

Ograničenje pri oblikovanju modela predstavlja relativno mali skup povijesnih podataka jer su posljedično mali skupovi podataka za učenje i testiranje. Na temelju validacije modela procjene UCR_ML, odnosno usporedbom s rezultatima procjene modela UCR, može se zaključiti da su za organizacije s ograničenim brojem projekata sa sličnim obimom posla primjereniji algoritamski modeli. Međutim, kao što i stručnjaci za procjenu savjetuju, primjena više različitih modela procjene pridonosi preciznijoj procjeni količine posla. Sukladno tome, organizacije s malom količinom povijesnih podataka ne bi trebale isključiti uporabu modela zasnovanog na metodama strojnog učenja, već ih nakon oblikovanja, nadograđivati novim povijesnim podacima.

Organizacije mogu lako primijeniti opisani proces razvoja modela UCR i UCR_ML i po potrebi ih prilagoditi svojim potrebama. U primjeni modela UCR stručni tim za procjenu može dodati novi ili ukloniti koji od predloženih faktora, postaviti težine atributa za nove ulazne faktore i kalibrirati veličinu UCR izraženu u čovjek – sat. Kod uporabe modela UCR_ML, organizacije također mogu

prema potrebi modificirati ulazne faktore s obzirom na karakteristike programskog proizvoda i projektnog tima.

U praksi preciznije procjene količine posla omogućuju projektnom timu poboljšanje cjelokupnog procesa upravljanja projektima - planiranje, raspoređivanje, upravljanje resursima i procjenu troškova. Stoga će se očekuje da će modeli procjene količine posla, razvijeni u sklopu ovog doktorskog istraživanja, imati daljnju primjenu u industrijskim projektima.

Literatura

[1] Moløkken, K. and M. Jorgensen: A review of software surveys on software effort estimation. International Symposium on Empirical Software Engineering. 2003. Rome, Italy: Simula Res. Lab. Lysaker Norway.

[2] Grimstad, S.: Understanding of Estimation Accuracy in Software Development Projects. 11th IEEE International Software Metrics Symposium (METRICS 2005)

[3] Jorgensen, M.; Grimstad, S.: The Impact of Irrelevant and Misleading Information on Software Development Effort Estimates: A Randomized Controlled Field Experiment, Software Engineering, IEEE Transactions on Year: 2011, Volume: 37, Issue: 5, Pages: 695 - 707, DOI: 10.1109/TSE.2010.78

[4] Jorgensen, M.; Grimstad, S.: Software Development Estimation Biases: The Role of Interdependence Software Engineering, IEEE Transactions on Year: 2012, Volume: 38, Issue: 3 Pages: 677 - 693, DOI: 10.1109/TSE.2011.40

[5] Jorgensen, M.: What We Do and Don't Know about Software Development Effort Estimation, Software, IEEE, Year: 2014, Volume: 31, Issue: 2, Pages: 37 - 40, DOI: 10.1109/MS.2014.49

[6] P. Clements and L. Northrop, Software product lines: practices and patterns: Addison-Wesley Longman Publishing Co., Inc., 2001.

[7] F. J. v. d. Linden, K. Schmid, and E. Rommes, Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering: Springer-Verlag New York, Inc., 2007.

[8] Fazal-e-Amin, Ahmad Kamil Mahmood, Alan Oxley: A Proposed Reusability Attribute Model for Aspect Oriented Software Product Line Components

[9] Nasir, M: A Survey of Software Estimation Techniques and Project Planning Practices, Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2006. SNPD 2006. Seventh ACIS International Conference on Year: 2006 Pages: 305 - 310, DOI: 10.1109/SNPD-SAWN.2006.11

[10] R. Agarwal, Manish Kumar t, Yogesh, S. Mallick, RM. Bharadwaj, D. Anantwar Infosys Technologies Limited, Calcutta, India, “Estimating software projects”, ACM SIGSOFT Software Engineering Notes Volume 26, Issue 4 (July 2001), Pages: 60 – 67

[11] Barry Boehm, Chris Abts and Sunita Chulani University of Southern California, Los Angeles, USA, IBM Research, 650 Harry Road, San Jose, CA, USA,” Software development cost estimation approaches –A survey”. Annals of Software Engineering volume 10, issue 1-4 (2000) pages 177–205, Year of publication: 2000 ISSN: 1022-7091

[12] Milicic, D.; Wohlin, C.: Distribution patterns of effort estimations, Euromicro Conference, 2004. Proceedings. 30th, Year: 2004, Pages: 422 - 429, DOI: 10.1109/EURMIC.2004.1333398

[13] Kusumoto, S.; Matukawa, F.; Inoue, K.; Hanabusa, S.; Maegawa, Y. : Estimating effort by use case points: method, tool and case study, Software Metrics, 2004. Proceedings. 10th International Symposium on Year: 2004, Pages: 292 - 299, DOI: 10.1109/METRIC.2004.1357913

[14] Rastogi, H.; Dhankhar, S.; Kakkar, M.: A Survey on Software Effort Estimation Techniques, Confluence The Next Generation Information Technology Summit (Confluence), 2014 5th International Conference - Year: 2014, Pages: 826 - 830, DOI: 10.1109/CONFLUENCE.2014.6949367

[15] G. Karner: Resource Estimation for Objectory Projects, 1993.

[16] Muhammad Usman, Emilia Mendes, Francila Weidt and Ricardo Britto: Effort Estimation in Agile Software Development: A Systematic Literature Review, PROMISE '14, September 2014, Torino, Italy

[17] K. Ribu, “Estimating Object-Oriented Software Projects with Use Cases,” 2008.

[18] Bente Anda, Hege Dreiem, Magne Jørgensen, and Dag Sjøberg: Estimating Software Development Effort based on Use Cases – Experience from Industry. In M. Gogolla, C. Kobryn (Eds.): UML 2001 - The Unified Modeling Language. Springer-Verlag. 4th International Conference, Toronto, Canada, October 1-5, 2001, LNCS 218, 2001.

[19] Mudasir Manzoor Kirmani, Abdul Wahid: Impact of Modification Made in Re-UCP on Software Effort Estimation, Journal of Software Engineering and Applications, 2015, 8, 276-289

[20] Nunes, N. J. (2010): iUCP - estimating interaction design projects with enhanced use case points. Proceedings of the 8th international conference on Task Models and Diagrams for User Interface Design, Springer, 131–145

[21] Periyasamy, K. & Ghode, A. (2009): Cost Estimation Using Extended Use Case Point (e-UCP) Model, International Conference on Computational Intelligence and Software Engineering

[22] Kirmani, M.M. and Wahid, A. (2015): Revised Use Case Point (Re-UCP) Model for Software Effort Estimation, International Journal of Advanced Computer Science and Applications (IJASCA), Vol. 6, No. 3, pp 65 – 71.

[23] Pedro Faria, Eduardo Miranda: Expert Judgment in Software Estimation during the Bid Phase of a Project – An Exploratory Survey, Software Measurement and the 2012 Seventh International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2012

[24] S.-W. Lin and V. M. Bier: "A study of expert overconfidence", Reliability Engineering & System Safety, vol. 93, no. 5, pp. 711-721, May, 2008

[25] Mary A. Meyer, Jane M. Booker: Eliciting and Analyzing Expert Judgment: A Practical Guide

[26] Christopher Rush and Rajkumar Roy: Expert judgement in cost estimating: Modelling the reasoning process, Concurrent Engineering. 2001; 9: 271 – 284

[27] Hughes, R. T. (1996). 'Expert judgement as an estimating method.' Information and Software Technology, 38, 67-75

[28] Lederer, A. L., and Prassad, J. (1998). 'A causal model for software cost estimating error.' IEEE Transactions on Software Engineering, 24 (2), 137-148.

[29] Alistair Cockburn: Writing Effective Use Cases. Addison-Wesley, 2000.

[30] Charles Richter: Designing Flexible Object-Oriented Systems with UML. Macmillan Technical Publishing, 2001.

[31] Alistair Cockburn. Structuring use cases with goals. Humans and Technology, 1997.

[32] John Cheesman and John Daniels. UML Components, A simple Process for Specifying Component-based Software. Addison-Wesley, 2000.

[33] Martin Fowler. UML Distilled. Addison-Wesley, 1997.

[34] Steve Sparks and Kara Kaspczynski: The art of sizing projects, Sun World.

[35] Lugo García, José Alejandro, García Pérez, Ana María Delgado Martínez, Ramsés Gestión de indicadores en proyectos de software. Perspectivas actuales y futuras Revista Cubana de Ciencias Informáticas, vol. 3, núm. 3-4, julio-diciembre , 2009, pp. 19-25

[36] Karner, G. Metrics for Objectory. Diploma thesis, University of Linköping, Sweden. No. LiTH-IDA-Ex-9344:21. December 1993.

[37] Albrecht, A.J. Measuring Application Development Productivity. Proceedings of joint SHARE, GUIDE and IBM Application Development Symposium. 1979.

[38] Schneider and Winters: Applying use Cases. Addison-Wesley, 1998.

[39] Richard D. Stutzke, Estimating Software-Intensive Systems: Projects, Products, and Processes, 2005.

[40] A guide to the Project Management Body of Knowledge (PMBOK GUIDE), Fifth edition, Project Management Institute (PMI)

[41] Frakes, W.B. and K. Kang, Software Reuse Research: Status and Future. IEEE Transactions on Software Engineering, 2005. 31(7): p. 529 - 536.

[42] Schmidt, D.C. and F. Buschmann. Patterns, Frameworks, and Middleware: Their Synergistic Relationships. in 25th International Conference on Software Engineering (ICSE 2003). 2003. Portland, Oregon, USA: IEEE Computer Society.

[43] Software reuse process, http://lombardhill.com/what_reuse.htm/ (11. srpnja 2016.)

[44] J. Maras, M. Štula, I. Crnković: Towards specifying pragmatic software reuse, ECSAW '15 Proceedings of the 2015 European Conference on Software Architecture Workshops, Article No. 54

[45] R. Holmes and R. J. Walker. Systematizing pragmatic software reuse, ACM Transactions on Software Engineering and Methodology (TOSEM), 21(4):20, 2012

[46] Raccoon, L., Fifty Years of Progress in Software Engineering. ACM SIGSOFT Software Engineering Notes, 1997. 22(1): p. 88-104.

[47] Schmid, K. and I. John. Developing, Validating and Evolving an Approach to Product Line Benefit and Risk Assessment in EuroMicro'02 - 28th Euromicro Conference, 2002.

[48] Clements, P.C. and L.M. Northrop, Software Product Lines – Practices and Patterns. 2001: Addison-Wesley.

[49] Northrop, L.M., SEI's Software Product Line Tenets, IEEE Software, 2002(July/August 2002): p. 32-40.

[50] Desouza, K.C., J.J. Raider, and T.H. Davenport, Intellectual Asset Reuse in Software Development, in Research Note. 2003, Accenture Institute for Strategic Change.

[51] Mohagheghi, P., Anda, B., Conradi, R.: Effort Estimation of Use Cases for Incremental Large-Scale Software Development, Proceeding ICSE '05 Proceedings of the 27th international conference on Software engineering, Pages 303-311

[52] Kemerer, C.F. An empirical validation of software cost estimation models. CACM, 30, 5 (May 1987), 416 - 429

[53] Fink, A.: The survey Handbook, 2nd edn. SAGE, Thousand Oaks/London (2003)

[54] ICT Competence Network for Innovative Services for Persons with Complex Communication Needs, http://whttp://www.ict-aac.hr/index.php/en/projects/ict-aac (February 15, 2018)

[55] Jurica Babic, Ivan Slivar, Zeljka Car and Vedran Podobnik: Prototype-driven Software Development Process for Augmentative and Alternative Communication Applications, 2015 13th International Conference on Telecommunications (ConTEL), 2015, 1 - 8

[56] Wholin, C. et al., Experimentation in software engineering, Springer-Verlag, 2012.

[57] Montgomery, D.C.: Design and Analysis of Experiments, 5th edn. Wiley, New York (2000)

[58] Robson, C.: Real World Research: A Resource for Social Scientists and Practitioners – Researches, 2nd edn. Blackwell, Oxford/Madden (2002)

[59] Siegel, S. Castellan, J.: Nonparametric Statistics for Behavioral Sciences, 2nd edn. McGraw-Hill International Editions, New York (1988)

[60] Stake. R.E.: The Art of Case Study Research. SAGE Publications, Thousand Oaks (1995)

[61] Yin, R.K.: Case Study Research Design and Methods, 4th edn. Sage Publications, Beverly Hills (2009)

[62] Petersen, K., Wholin C.: Context in industrial software engineering research. In: Proceedings of the 3rd ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, Lake Buena Vista, pp. 401-404 (2009)

[63] Hannay, J.E., Sjoberg, D.I.K., Dyba, T.: A systematic review of theory use in software engineering experiments. IEEE Trans. Soft. Eng. 33(2), 87-107 (2007), doi. 10.1109/TSE.2007.12

[64] Sjoberg, D.I.K., Dyba, T, Anda, B., Hannay, J.E.: Building theories in software engineering. In: Shull, F. Singer, J., Sjoberg, D. (eds.) Guide to Advanced Empirical Software Engineering, Springer, London (2008)

[65] Cook T.D., Campbell, D.T.: Quasi-experimentation – Design and Analysis Issues for Field Settings, Houghton Mifflin Company, Boston (1979).

[66] Luís M. Alves; André Sousa; Pedro Ribeiro; Ricardo J. Machado: An empirical study on the estimation of software development effort with use case points, 2013 IEEE Frontiers in Education Conference (FIE), 2013, 101-107

[67] Arthur L. Samuel. Some studies in machine learning using the game of checkers, IBM Journal of Research And Development, 71–105, 1959.

[68] Thomas M. Mitchell. Machine Learning. McGraw-Hill, Inc., New York, NY, USA, 1st edition, 1997.

[69] Russell, Stuart; Norvig, Peter (2003) [1995]. Artificial Intelligence: A Modern Approach (2nd ed.). Prentice Hall. ISBN 978-0137903955.

[70] Mohri, Mehryar; Rostamizadeh, Afshin; Talwalkar, Ameet (2012). Foundations of Machine Learning. USA, Massachusetts: MIT Press. ISBN 9780262018258.ž

[71] https://www.fer.unizg.hr/_download/repository/SU-13-Grupiranje.pdf (dohvaćeno 10. listopada 2018.)

[72] Jianfeng Wena, Shixian Li, Zhiyong Lin, Yong Hu, Changqin Huang: Systematic literature review of machine learning based software development effort estimation models, (Journal of) Information and Software Technology 54 (2012) 41–59

[73] E. Mendes, I. Watson, C. Triggs, N. Mosley, S. Counsell, A comparative study of cost estimation models for web hypermedia applications, Empirical Software Engineering 8 (2) (2003) 163–196

[74] I.F.B. Tronto, J.D.S. Silva, N.S. Anna, An investigation of artificial neural networks based prediction systems in software project management, Journal of Systems and Software 81 (3) (2008) 356–367

[75] M.O. Elish, Improved estimation of software project effort using multiple additive regression trees, Expert Systems with Applications 36 (7) (2009) 10774–10778

[76] Bilge Baskeles; Burak Turhan; Ayse Bener: Software effort estimation using machine learning methods, 2007 22nd international symposium on computer and information sciences, 2007, Pages: 1 – 6

[77] Ethem Alpaydin (2009.), Introduction to Machine Learning, The MIT Press

[78] https://www.ibm.com/support/knowledgecenter/en/SS3RA7_15.0.0/com.ibm.spss.modeler.help/mainwindow_projectcrisptab.htm (8.travnja 2018.)

[79] https://www.ibm.com/support/knowledgecenter/en/SS3RA7_17.0.0/clementine/modelling_nodes.html (29. travnja 2018.)

[80] Karna, Hrvoje; Gotovac, Sven: Modeling Expert Effort Estimation of Software Projects, Proceedings of the 22nd Conference on Software, Telecommunications and Computer Networks (SoftCOM 2014)

[81] Karel Dejaeger; Wouter Verbeke; David Martens; Bart Baesens: Data Mining Techniques for Software Effort Estimation: A Comparative Study, IEEE Transactions on Software Engineering, Year: 2012 , Volume: 38 , Issue: 2, Pages: 375 - 397

[82] M. Jørgensen and M. Shepperd, "A Systematic Review of Software Development Cost Estimation Studies," IEEE Trans. Software Eng., vol. 33, no. 1, pp. 33-53, Jan. 2007

[83] Haitham Hamza; Amr Kamel; Khaled Shams: Software Effort Estimation Using Artificial Neural Networks: A Survey of the Current Practices, 2013 10th International Conference on Information Technology: New Generations, 2013

[84] Ying Li, Zongli Zhang, Yue Zhao: Analysis on influencing factors of consumers' purchasing behavior online for furniture: a case study on furniture malls and business centers in Harbin, ICEC '16: Proceedings of the 18th Annual International Conference on Electronic Commerce: e-Commerce in Smart connected World, August 2016

[85] Satwinder Singh, Rozy Singla: Comparative Performance of Fault-Prone Prediction Classes with K-means Clustering and MLP, ICTCS '16: Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies, March 2016

[86] C. Wang Guomin: Analysis of Chinese and American Gymnastics Based on SPSS Differentiation Model, 2017 International Conference on Robots & Intelligent System (ICRIS), 2017

[87] D. Zhifang He; Shuiping Chen: Application of spss software on mental health education for community residents, 2015 10th International Conference on Computer Science & Education (ICCSE), 2015

[88] E. Yajing Li; Jiayi Yao: The quantity prediction of 4G customers of China mobile communications corporation based on SPSS modeler, 2016 International Conference on Logistics, Informatics and Service Sciences (LISS)

[89] F. A. Ismail; N. Z. Mohd Fauzi; N. R. Shamsuddin; A. Abd ul Hadi; N. N. Azid; N. Mohd Razali: The analysis of job related stress and health related quality of life (HRQoL) of lecturers: Using SPSS 16 and Structural Equation Modeling, 2013 IEEE Business Engineering and Industrial Applications Colloquium (BEIAC), 2013

[90] G. MD Zakaria Rahman, MD. Nahiduzjaman Sajib, M.M.Safayet Hasan Rifat, Md Hossam-E-Haider, Md. Ali Azam Khan: Forecasting the Long Term Energy Demand of Bangladesh Using SPSS from 2011-2040, 2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT), 2016

Popis slika

Popis tablica

Životopis

Katija Rak (rod. Šeparović) rođena je 1983. godine u Zagrebu. Diplomirala je 2006. na Sveučilištu u Zagrebu, Fakultetu elektrotehnike i računarstva. Od 2006. godine radi u IBM Hrvatska na različitim pozicijama: poslovni konzultant za implementaciju informacijskih sustava (do 2011.), projektni menadžer (od 2012. do 2016.) te kao rukovoditelj poslovne podrške u Client Innovation Centru od 2017. do danas. Sudjelovala je na globalnim projektima implementacije poslovnog informacijskog sustava i vodila timove na projektima razvoja softverskih rješenja.

Tijekom 2012. godine započinje znanstveno – istraživački rad iz područja upravljanja projektima, s naglaskom na procjenu, mjerenje i praćenje količine posla na projektima razvoja programskih proizvoda.

Objavila je više radova i sudjelovala na internacionalnim konferencijama.

Popis objavljenih djela

1. Rak K, Car Ž, Lovrek I.
   Effort estimation model for software development projects based on use case reuse. J Softw Evol Proc. 2018; e2119. https://doi.org/10.1002/smr.2119
2. Šeparović, Katija; Car, Željka.
   Success model of information system implementation // Proceedings of 32nd International

   Convention on Information and Communication Technology, Electronics and Microelectronics

   Golubić, Stjepan ; Mikac, Branko ; Hudek, Vlasta (ur.).

   Rijeka : MIPRO, 2009. (predavanje,međunarodna recenzija,objavljeni rad,znanstveni).

3. Zagajšek, Barbara; Šeparović, Katija; Car, Željka.

   Requirements Management Process Model for Software Development Based on Legacy

System Functionalities // Proceedings of the 9th International Conference on Telecommunications ConTEL 2007.

Zagreb, 2007. (predavanje,međunarodna recenzija,objavljeni rad,znanstveni).

4. Novosel, Luka; Šeparović, Katija; Car, Željka.

Student Project Management: PM Agent Development // Proceedings of the Conference Telecommunications & Information MIPRO 2006 / Golubić, Stjepan; Mikac, Branko; Hudek, Vlasta (ur.). Rijeka : MIPRO, 2006. 134-140 (predavanje, međunarodna recenzija, objavljeni rad, znanstveni).

Biography

Katija Rak (Šeparović) was born in 1983 in Zagreb. In 2006 she received her diploma from the Faculty of Electrical Engineering and Computing, University of Zagreb. She has been working in IBM Croatia since 2006 at various business roles: SAP information system Consultant (till 2011), Project Manager (from 2012 to 2016) and Operations Manager of Client Innovation Center from 2017 till now. She led multicultural teams in global software implementation projects.

In 2012 she started her research in project management area within ICT, with the focus on project effort in terms of estimation, measurement and monitoring activities. She published several papers within her research domain and participated in international conferences.