UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

EVELIN VATOVEC KRMAC

MODEL PROCESA PONOVNE UPORABE PROGRAMSKIH KOMPONENT

DOKTORSKA DISERTACIJA

MENTOR: doc. dr. TOMAŽ MOHORIČ
SOMENTOR: prof. dr. MARKO PAVLIHA

LJUBLJANA, 2005

"Information technology policy
is too important to leave entirely to lawyers
and it's also too important to leave entirely to technologists."

P. Samuelson, 1998

KAZALO VSEBINE

KAZALO SLIK

KAZALO TABEL

POVZETEK

Današnje zahteve tržišč programske opreme postavljajo njihove razvijalce še bolj kot kdajkoli poprej pred izjemno zahtevne naloge. Programske sisteme, v čim večji možni meri prilagojene potrebam in načinu poslovanja organizacije, je treba izdelati v čim krajšem času in za čim nižjo ceno. Sistemi morajo biti čimbolj interoperativni, prilagodljivi, nadgradljivi, odprti, njihovo vzdrževanje in spreminjanje pa preprosto in učinkovito. Zato so razvijalci prisiljeni posegati po tehnologijah in metodah razvoja, ki jim omogočajo doseganje omenjenih zahtev.

Ponovna uporaba, vpeljana v razvojni proces planirano, organizirano in sistematično, to omogoča. Pri njeni vzpostavitvi se je treba lotiti vrste različnih problemov: tehničnih, tehnoloških, upravljavskih, ekonomskih, pa tudi pravnih. V dosedanjih raziskavah je bilo reševanje pravnih problemov najbolj zapostavljeno, čeprav lahko slednji zelo otežijo, zavirajo ali celo onemogočijo ponovno uporabo.

Obstoječe pravne ureditve s področja varstva pravic intelektualne lastnine nezadostno ali pa sploh ne obravnavajo koncepta ponovne uporabe programskih komponent in vseh njegovih posebnosti. Ni realno pričakovati, da bi vse porajajoče se pravno obarvane probleme sploh lahko zajeli v predmetni zakonodaji oziroma bi to trajalo predolgo. Za ustrezno zaščito in upravljanje pravic intelektualne lastnine na programskih komponentah je tako potrebna sinergija med pravom in tehnologijo, kar pomeni, da je treba poskrbeti tudi za ustrezne tehnološke rešitve in mehanizme za odpravljanje ali omejevanje kršitev pravic intelektualne lastnine in drugih pravnih problemov, ki nastajajo med procesom ponovne uporabe. Mehanizmi, ki so opisani v disertaciji, predstavljajo primer omenjene sinergije.

Namen pričujoče disertacije je predstaviti modele in ogrodja, na temelju katerih lahko vzpostavimo celosten proces ponovne uporabe, v katerem ima osrednjo vlogo knjižnica

ponovno uporabnih komponent z vsemi storitvami, ki jih ponuja na eni strani proizvajalcem komponent, na drugi strani pa potrošnikom komponent ali ponovnim uporabnikom. Predlagani modeli in ogrodja so postavljeni po vzoru standardov, so splošni in sami po sebi ponovno uporabni.

Med najpomembnejše prispevke te disertacije sodi model knjižnice ponovno uporabnih komponent. Knjižnica naj bi združevala varne, preskušene, kakovostne ponovno uporabne komponente domene, njena struktura naj bi slonela na modelih domene, podpirala naj bi sledenje uporabe komponent in izdelave različic komponent, osveževanje spremne dokumentacije, zbiranje povratnih informacij o komponentah in storitvah knjižnice, zbiranje potreb in zahtev ponovnih uporabnikov, programsko podprto sklepanje licenčnih pogodb za komponente, povezovanje v komponentna tržišča oziroma povezovanje z drugimi knjižnicami domene, varno distribucijo komponent ter vse potrebne postopke za sprejem komponent (preskušanje, ugotavljanje ponovne uporabnosti in ustreznosti, certificiranje, označevanje lastništva in pravic na komponenti idr.). Postopki in opravila so opredeljeni tako, da se je možno v čim večji meri izogniti kršitvam pravic na komponentah.

Ker vse odnose med udeleženci v procesu ponovne uporabe ter opredelitev pravic na posameznih komponentah uravnavajo pogodbe, so v disertaciji podana tudi ponovno uporabna ogrodja za posamezne vrste pogodb, ki jih je mogoče standardizirati in katerih sklepanje je mogoče programsko podpreti.

Rezultati disertacije prispevajo h globalnejšemu pogledu na problematiko ponovne uporabe, predstavljajo iztočnico za pravne strokovnjake pri spreminjanju obstoječih pravnih ureditev ali postavljanju nove pravne ureditve *sui generis* za področje informacijske tehnologije ter pravic intelektualne lastnine, hkrati pa predstavljajo dobro metodološko podlago za postavitev celostnega in učinkovitega procesa ponovne uporabe, ki ga je mogoče nadgraditi z ekspertnim sistemom, v katerega bazi znanj bo vsebovano znanje o ponovno uporabnih komponentah ter izkušnjah pri njihovi ponovni uporabi.

SUMMARY

Today's software market demands are more than ever confronting software developers with extremely challenging tasks. Programming systems need to be correspondingly adapted to meet the requirements of the organization. They must be produced in the shortest possible time and at the lowest possible cost. The systems must be as interoperable as possible, as adaptable, upgradeable and open as possible, their maintenance and modification as simple and as effective as possible. This situation forces software developers to use technologies and methods that allow them to satisfy the software market demands of today.

Planned, organized, and systematic integration of software reuse into the software development process makes this possible. On the implementation of reuse, many different problems have to be tackled: technical, technological, those concerning management as well as economical and legal issues. Investigations so far have been neglecting solutions to legal problems most of all, although legal issues can disturb and hinder or even prevent the reuse of software components.

The existing legal regulations about the protection of intellectual property rights do not sufficiently address – or do not address at all – the concept of the reuse of software components and all its specificities. It would be unrealistic to expect that the relevant legislation could cover all the emerging problems containing legal knots; it would also take too much time to do so. Proper protection and management of intellectual property rights on software components require a synergy between law and technology. Corresponding technological solutions and mechanisms need to be provided to abolish or at least reduce any sort of infringement of intellectual property rights and other legal problems that appear during the reuse process. The mechanisms described in this thesis present an example of such a synergy.

The objective of the thesis was to present models and frameworks as the basis for the entire process of reuse. The central role in this process plays the library of reusable components with all the services that it offers to the component producers on the one hand, and to the component users or reusers on the other. The suggested models and frameworks follow the standards, they are general and in themselves reusable.

One of the major contributions in this thesis is the model of the library of re-usable components. The library should incorporate safe, proven, high quality reusable components of the domain, its structure should rest on domain models. It should support the tracking of component use and modifications, and the updating of supplementary documentation. It should support the collecting of feedback on the library's components and services, and of the needs and requirements of reusers. Licence agreements for components, connection into component markets and interconnecting with other libraries of the domain, safe distribution of the components and all the procedures necessary for accepting the components (testing, assessment of reusability and suitability, certifying, defining and indicating the components' ownership and the related rights, etc.) should also be software supported. The definitions of procedures and activities largely allow avoiding any infringement of the rights on the components.

The thesis also offers reusable frameworks for various kinds of agreements, since agreements regulate all the relationships between the participants in the reusing process, and the definitions of the rights concerning separate components. Agreements can be standardized and signed by means of program support.

The results of the thesis contribute to a more global view on reuse challenges; for legal experts they represent a cue to change the existing legal regulation or to establish a new legal regulation *sui generis* for the area of information technology and intellectual property rights. At the same time, they represent a solid methodological ground for setting up a whole and effective reuse process that can be upgraded by an expert system containing a knowledge base equipped with the information on reusable components, and with experiences concerning their reusing.

1 UVOD

Uspešnost in učinkovitost poslovanja sodobnih podjetij sta v veliki meri odvisni od učinkovitosti njihovih poslovnih informacijskih sistemov. Prilagajanja in odzivanja na nove tržne razmere in zahteve, konkurenco, nove tehnologije in podobno zahtevajo prilagodljiv, robusten, nadgradljiv informacijski sistem, ki ga je mogoče na čim bolj preprost način spreminjati in prilagajati novim potrebam. Razvoj takih informacijskih sistemov pa zahteva uporabo sodobnih, ustrezno prilagojenih metodologij razvoja.

Dinamičnost spreminjanja tržnih zahtev vpliva tudi na razvijalce programskih sistemov, ki se morajo tem spremembam dovolj hitro odzivati. Posledice takih potreb po hitrem spreminjanju, prilagajanju in razvoju informacijskih sistemov se kažejo že vrsto let v t.i. krizi programske opreme. Tržišče sili razvijalce programske opreme k skrajševanju razvojnega časa, časa plasiranja izdelka na tržišče, k povečevanju raznolikosti ponudbe, standardiziranosti, interoperativnosti, odprtosti, zniževanju cen ter prilagajanju svojih izdelkov in njihovega razvoja novim tehnologijam ipd. Rešitve se porajajo v obliki novih razvojnih pristopov, s katerimi naj bi razvoj predvsem skrajšali in pocenili. Eden takih pristopov, ki je predmet pričujoče disertacije, je ponovna uporaba.

1.1 PREDSTAVITEV RAZISKOVALNEGA PODROČJA

Ponovna uporaba programske opreme (ang. *software reuse*) ni razvojna tehnika, ampak proces, s katerim nadgradimo, dopolnimo in razširimo obstoječi razvojni proces, zato pogosteje govorimo o vključevanju ponovne uporabe v razvoj programske opreme. Če je to vključevanje načrtovano, postopno, če se vsi udeleženci v tem procesu, tudi najvišje vodstvene strukture razvojne organizacije ali podjetja, ustrezno izobrazijo in seznanijo s spremembami, lahko govorimo o sistematični ponovni uporabi [59]. Za razliko od priložnostne ponovne uporabe, ko je ponovna uporaba prepuščena volji in znanju posameznega razvijalca ali razvojne ekipe, je sistematična ponovna uporaba

institucionaliziran organizacijski pristop k načrtnemu razvoju programskih komponent za ponovno uporabo. Razvite komponente je nato treba vzdrževati in skladno uporabljati, tako da obdržijo visoko stopnjo ponovne uporabnosti.

Med največje pridobitve vzpostavitve sistematične ponovne uporabe sodijo povečanje produktivnosti, kakovosti, zmogljivosti in funkcionalnosti razvite programske opreme, znižanje stroškov ter povečanje učinkovitosti razvoja in bodočega vzdrževanja. Naštete pridobitve so predvsem posledica dobrega načrtovanja, preskušanja in dokumentiranja ponovno uporabne programske opreme ter njene večje razumljivosti [55], v luči pridobitev pa je ključnega pomena upravljanje teh komponent in doseganje njihove razpoložljivosti za nadaljnjo ponovno uporabo.

Vzpostavitev sistematičnega razvoja na temelju ponovne uporabe zahteva velika začetna vlaganja, tako finančna kakor organizacijska, zahteva pa tudi že določeno zrelost razvojnega procesa, ki ga nadgrajujemo z aktivnostmi ponovne uporabe. V praksi se je izkazalo [41, 43, 45 idr.], da so za sistematično ponovno uporabo poleg kritične mase ponovno uporabnih komponent potrebni še primerna organizacijska infrastruktura in upravljanje razvojnega procesa programske opreme.

1.2 DOSEDANJE RAZISKAVE

Raziskavam organizacijskih, upravljavskih in tehničnih problemov razvoja na temelju ponovne uporabe je bilo v zadnjem desetletju posvečene precej pozornosti, kar se zrcali v mnogih knjigah, znanstvenih in strokovnih prispevkih ter konferencah in delavnicah, posvečenih ponovni uporabi, pa tudi v raznih izdelkih, ki so nastali kot nadgradnja teh raziskav [nekaj teh virov je zbranih v seznamu A literature te disertacije]. Med najpogosteje in najgloblje obravnavana področja ali tematike zagotovo lahko uvrstimo vključevanje ponovne uporabe v objektno usmerjen način razvoja programskih sistemov in objektno usmerjene jezike, merjenje ponovne uporabe, analizo domene, komponentne tehnologije, vzorce, ogrodja, skupine izdelkov[1], iskalne mehanizme knjižnic ponovno uporabnih komponent, razvojna in knjižnična orodja, ki podpirajo ponovno uporabo, mehanizme vizualizacije

[1] Ang. *product lines*.

delovanja izvršnih (kodnih) komponent knjižnice, knjižnice črnih komponent ter v zadnjem času njihovo preseljevanje v spletno okolje in njihovo medsebojno povezovanje.

V vseh teh letih in desetletjih je bilo predvsem v teoriji zgrajenih veliko tehnologij, metod in smernic za povečanje deleža ponovne uporabe v razvojnem procesu [1, 9, 12, 13, 19, 29, 31, 34, 47, 50, 52, 53 idr.]. Žal pa se jih je le malo udejanilo v praksi, saj so razhajanja med teorijo in prakso tudi tu precejšnja [18, 34, 36, 42]. Poleg tega primanjkuje odzivov, mnenj in zahtev iz prakse [18, 36], kar še bolj otežuje ugotavljanje vzrokov, zaradi katerih se ponovna uporaba, kljub vsem prednostim, ki se ji pripisujejo, ne uspe razmahniti v napovedanih okvirih. Prieto-Diaz [43] je že pred desetimi leti napovedal, da se ponovna uporaba danes ne bo več obravnavala kot neka nadgradnja razvojnega procesa, ampak bo že glavni sestavni del procesa programskega inženirstva, kar pa je le deloma res.

Seveda se poraja vprašanje, kje in kateri so vzroki, da se te napovedi niso še povsem uresničile. Uvajanje sistematične ponovne uporabe je zahteven in dolgotrajen proces, ki terja reševanje problemov tehnične, tehnološke, upravljavske, ekonomske in pravne narave. Kakor pravi Jakkola [20], je ponovna uporaba multidisciplinaren pristop h gradnji programskih izdelkov. Za razliko od problemov pravne narave je bilo v reševanje oziroma izogibanje drugim vrstam težav usmerjenih veliko raziskav in napora. Zelo velik napredek je bil do danes dosežen v smeri razumevanja kritičnih tehnologij, metod, procesov in organizacijskih faktorjev, ki vodijo k učinkoviti ponovni uporabi [15,16]. Precej manj ali celo zelo malo je bilo narejenega za prepoznavanje in reševanje pravnih problemov, povezanih s ponovno uporabo programskih komponent, ki se nanašajo predvsem na stvarnopravne probleme (npr. lastninska pravica), kakor tudi na pravice intelektualne lastnine, še posebno poslovnih komponent, ki zahtevajo sklenitev licenčne pogodbe in plačilo licenčnine. Poleg tega mora biti poslovna logika, vgrajena v komponente, ustrezno zaščitena kot intelektualna lastnina. Ponovna uporaba poslovnih komponent predstavlja torej večje tveganje in višje stroške za razvijalce in za ponovne uporabnike teh komponent. Ravno zato je bilo ponovni uporabi in knjižnicam, ki bi hranile tovrstne komponente, do sedaj namenjenega malo prostora. Pravni problemi zahtevajo in vsiljujejo določene omejitve v razvojne procese in procese delovanja knjižnice, posebno tiste, ki se nanašajo na sprejemanje komponent in njihovo razpečevanje. Ponovna uporaba ne poraja novih vrst pravnih problemov, ki jih doslej ne bi srečevali, vendar

se z razširjeno in sistematično ponovno uporabo ti problemi močno namnožijo in postanejo znatno kompleksnejši. Zato v okolju ponovne uporabe predstavljajo enega večjih razlogov, zaradi katerega se razvijalci izogibajo uporabi vnaprej zgrajenih komponent, posebno poslovnih [25, 32].

Do kršitev pravic intelektualne lastnine lahko pride iz več razlogov. Med najosnovnejše gotovo sodi ta, da razvijalci in ponovni uporabniki ne poznajo oblik pravnega varstva programske opreme ter pravic, ki iz posamezne oblike izhajajo. Do veliko kršitev prihaja tudi zaradi nevzpostavljenosti ustreznih mehanizmov knjižnice, kot so npr. sledenje izvoru komponente, sledenje njeni ponovni uporabi, različicam ipd. ter odsotnost določenih aktivnosti in opravil v razvojnem procesu, s katerimi lahko vsaj delno preprečimo ali včasih celo onemogočimo tovrstne kršitve. Ne smemo seveda zanemariti dejstva, da kljub še tako dovršenim mehanizmom knjižnice in razvojnem procesu ostaja problem kršenja pravic drugih predvsem moralni problem posameznega udeleženca v procesu razvoja programske opreme. Reševanje sporov, povračil škod in drugih težav, ki nastanejo kot posledica kršitve pravic intelektualne lastnine pri ponovni uporabi, naj bi uravnavala ustrezna zakonodaja. Ob podrobnejšem pregledu oblik pravnega varstva [seznam C], zakonov in drugih pravnih aktov s področja intelektualne lastnine programske opreme [seznam D], pa lahko v njih kaj hitro ugotovimo velike vrzeli in neustreznosti teh ureditev za sodobna razvojna in digitalna okolja. Večina teh pomanjkljivosti in neustreznosti izhaja iz dejstva, da se področje informacijske tehnologije bliskovito razvija in spreminja, zakonodaja pa niti približno ne uspe slediti vsem spremembam ter se ustrezno prilagoditi. Drugi razlog za neustreznost je dejstvo, da obstoječa pravna ureditev ne obravnava varstva pravic na programski opremi, ampak se omejuje zgolj na varstvo pravic na računalniških programih, in še to večinoma računalniških programih kot avtorskih delih [88, 103]. Najmočnejši razlog za neustreznost predmetne zakonodaje pa so v njej postavljene omejitve spreminjanja, izboljševanja, reprodukcije, vzdrževanja ter sledenja programski opremi, ki pa predstavljajo osnovne aktivnosti pri ponovni uporabi programskih komponent.

Ker menim, da so pravni problemi eno glavnih zaviral pri razmahnitvi ponovne uporabe, ker pravni problemi ponovne uporabe doslej še niso bili v kontekstu razvoja programske opreme, sploh pa ne na temelju ponovne uporabe, nikoli celostno obravnavani, ker se zavedam, da je

tvorjenje svojevrstnih oblik pravnega varstva programske opreme, ki bi ustrezno obravnavalo tudi koncept ponovne uporabe, predolgo trajajoč proces, in ker pravnih problemov ni mogoče v celoti rešiti samo s posodobitvijo in ureditvijo pravnih aktov s področja intelektualne lastnine, ampak je treba poiskati tudi ustrezne tehnološke rešitve ali kombinacije obeh, v pričujoči disertaciji ustrezno nadgradim razvojni proces in mehanizme knjižnice ponovno uporabnih komponent, s katerimi rešim ta problem s tehničnega, tehnološkega in delno organizacijskega vidika, podam pa tudi osnovne iztočnice, ki naj bi jih pravni strokovnjaki upoštevali pri posodabljanju zakonodaje, ki ureja področje varovanja pravic na programski opremi. Nerešen ostane problem moralnih vrednot, na katere, žal, ne moremo vplivati z nobenim od predlaganih mehanizmov. V tej smeri je po mojem mnenju mogoče veliko narediti z ustreznim izobraževanjem in seznanjanjem razvijalcev s posledicami, ki jih take kršitve prinašajo posamezniku, pa tudi industriji ponovne uporabe nasploh.

1.3 CILJI DISERTACIJE

Za zaščito in ustrezno upravljanje pravic intelektualne lastnine na programskih komponentah je potrebna sinergija med pravom in tehnologijo. Pričujoča disertacija, v kateri sem se kot "nepravnica" lotila obravnave pravnih problemov, ki nastajajo kot posledica tehnoloških in tehničnih prijemov pri razvoju programske opreme, je primer take sinergije. V njej predlagam modele in ogrodja, na temelju katerih bi bilo mogoče dosegati sistematično ponovno uporabo ter odpraviti pravne probleme, ki lahko nastanejo v procesu ponovne uporabe. Izhodišče, ki sem si ga postavila, je, da bi bili ti modeli ponovno uporabni, kar pomeni, da bi (po vzoru iz standardov [144, 145, 146]) predpisali zgolj ogrodja ter ključne elemente za izvedbo potrebnih aktivnosti, ne pa posameznih korakov, zaporedij teh korakov, natančnih navodil ali "receptov", kako nek korak izvesti, pa tudi na nobeno konkretno metodologijo razvoja ali pristop k razvoju naj ne bi bili vezani. Izogniti sem se želela pretirani formalizaciji metod in postopkov, saj pogosto dosežemo s tako formalizacijo ravno nasproten učinek. Želim namreč doseči, da bi bilo možno te modele kot ogrodja preprosto prilagoditi razvojnim tehnikam, navadam, značilnostim domene in drugim specifikam posameznih razvojnih skupin in okolij. To pa je možno le, če so ti modeli dovolj splošno opredeljeni. Poseben poudarek želim v disertaciji dati tudi ponovni uporabi izkušenj in znanja neke domene ali množice sorodnih domen, predvsem v obliki ustrezno strukturiranih informacij, ki so vsebovane v spremljajoči

dokumentaciji komponente, od katere je v veliki meri odvisna kakovost komponente, s tem pa tudi stopnja njene ponovne uporabnosti.

V ta namen sem si za cilj postavila:

- *zgraditi celosten teoretični model procesa ponovne uporabe, ki bo slonel na komponentnem razvoju ter nadgraditi ta proces z aktivnostmi in postopki za preprečevanje in odpravo pravnih problemov pri ponovni uporabi;*
- *zgraditi model knjižnice ponovno uporabnih komponent, ki bo zagotavljala čim hitrejše iskanje ustrezne komponente ter shranjevanje za uspešno in učinkovito ponovno uporabo bistvenih informacij skupaj s komponento, pri čemer lahko knjižnica vsebuje različne vrste komponent, od prostih do poslovnih komponent, od črnih do belih;*
- *nadgraditi knjižnico ponovno uporabnih komponent z mehanizmi za dodeljevanje pravic, označevanje ter spremljanje lastništva in različic komponent ter z mehanizmi za oblikovanje in sklepanje licenčnih pogodb;*
- *definirati vsebino spremljajoče dokumentacije k ponovno uporabni komponenti, da bo njeno vključevanje v drug izdelek čim hitrejše in čim preprostejše.*

Pri oblikovanju v disertaciji predstavljenega okolja ponovne uporabe, torej procesa, katerega osrednji del predstavlja knjižnica ponovno uporabnih komponent, skušam modele in aktivnosti opredeliti tako, da bi v čim večji meri spodbujali in olajšali ponovno uporabo. To poskušam doseči s preprosto opredelitvijo in razumljivostjo konceptov, opredelitvijo modelov, katerih uporaba in prilagajanje konkretnim tehnikam bosta preprosti, delno ali popolno avtomatizacijo zahtevnih in dolgotrajnih postopkov (npr. oblikovanje in sklepanje licenčnih ali drugih pogodb), mehanizmi komuniciranja in povratnega poročanja med udeleženci v procesu ponovne uporabe, mehanizmi nagrajevanja in spodbujanja razvijalcev ponovno uporabnih komponent in uporabnikov teh komponent ipd.

Kot izhodišča za doseganje zastavljenih ciljev:

- identificiram in preučim možne pravne probleme, ki lahko onemogočijo ali otežijo vključitev ponovne uporabe v razvojni program, ter za njihovo odpravo poiščemo

ustrezne mehanizme, s katerimi nadgradim proces ponovne uporabe in knjižnico ponovno uporabnih komponent;

- specificiram procese, aktivnosti in opravila, s katerimi je mogoče nadgraditi poljubni razvojni življenjski cikel programske opreme in s tem doseči sistematično ponovno uporabo, ki ni omejena samo na en razvojni projekt, ampak je množica med seboj povezanih in odvisnih programskih izdelkov;
- razvoj programske opreme opredelim kot procesno usmerjen, domensko specifičen, arhitekturno usmerjen, temelječ na knjižnici ponovno uporabnih komponent;
- identificiram vrste pogodb, ki nastajajo v procesu ponovne uporabe, standardiziram vsebino pogodb ter predlagam mehanizem programsko podprtega sklepanja licenčnih pogodb kot najpogostejše oblike pogodbe v procesu ponovne uporabe.

Področji ponovne uporabe in pravic intelektualne lastnine, ki sta obravnavani v pričujoči disertaciji, sta izjemno obširni, zato se moram omejiti samo na zastavljene cilje ter določene elemente razvojnega procesa in določene mehanizme knjižnic – tiste, ki so bili v predhodnih raziskavah že dodobra obdelani, kot so iskalni mehanizmi po knjižnicah, šifriranje digitalnih vsebin, orodja, ocenjevalne tehnike, metrike ipd. – pa izpustiti ali omeniti le kot referenco.

1.4 STRUKTURA DISERTACIJE

Disertacija poleg uvoda, v katerem predstavim raziskovalno področje ter cilje, ki sem si jih zastavila, zajema še pet ključnih in zaključno poglavje.

Drugo poglavje z naslovom *Ponovna uporaba in ponovno uporabne komponente* namenjam predstavitvi osnovnih pojmov, povezanih s konceptom ponovne uporabe, strategijam ponovne uporabe ter primerjavi med njimi. Slednja podaja tudi osnovne parametre za izračun stroškov, na temelju katerih se lahko neka razvojna skupina odloči za posamezno strategijo ali kombinacije strategij. Podajam tudi tveganja in izzive, s katerimi se med ponovno uporabo soočajo posamezni udeleženci v procesu razvoja na temelju ponovne uporabe ali, kakor ga v disertaciji krajše poimenujemo, komponentnega razvoja.

Tretje poglavje z naslovom *Pravni vidiki programske opreme in njene ponovne uporabe* v celoti namenjam problematiki varstva pravic intelektualne lastnine. Sprva predstavim oblike varovanja intelektualne lastnine na računalniških programih, nato neupravičenost poenotenja koncepta računalniškega programa s konceptom programske opreme v veljavnih aktih s tega področja, ter odsotnost določb, ki bi se dale uporabiti v specifičnem primeru ponovne uporabe programske opreme. Podam tudi natančen in poglobljen pregled osnovnih družin pravnih problemov, ki se v praksi odražajo v obliki kršitev pravic iz naslova intelektualne lastnine.

Celotno četrto poglavje, *Proces ponovne uporabe,* posvetim razvojnemu procesu na temelju ponovne uporabe. Podam ogrodje razvojnega procesa, v katerega so vključene aktivnosti in opravila, specifični za ponovno uporabo. Opisano ogrodje je možno prilagoditi različnim metodologijam razvoja in razvojnim tehnikam. Dodane so tudi t.i. pravne aktivnosti, ki bi jih bilo treba med razvojem izvesti, da bi morebitne kršitve pravic intelektualne lastnine med ponovno uporabo v čim večji meri odpravili.

Peto poglavje, ki predstavlja osrednje poglavje disertacije, nosi naslov *Model knjižnice ponovno uporabnih komponent.* Knjižnica ponovno uporabnih komponent je namreč osrednji koncept v kontekstu ponovne uporabe ali ključni element v procesu ponovne uporabe. Pa ne samo zato, ker je vezni člen med razvojem za in razvojem s ponovno uporabo, pač pa tudi zato, ker so z njo povezani vsi mehanizmi in vzvodi za učinkovito in uspešno ponovno uporabo. Knjižnica je urejeno zbirališče ponovno uporabnih komponent, po katerih posegajo razvijalci aplikacij, ki igrajo vlogo ponovnih uporabnikov. Pri oblikovanju arhitekture, strukture in delovanja knjižnice je treba ponovnemu uporabniku zagotoviti učinkovito iskanje po knjižnici, dostop do vseh potrebnih informacij v zvezi s komponento in njenimi predhodnimi ponovnimi uporabami, komunikacijo s knjižnico in prek nje z razvijalci komponent, poleg tega pa še mehanizme, ki v čim večji meri preprečujejo kršenje pravic intelektualne lastnine med pregledovanjem vsebine knjižnice in funkcionalnosti posameznih komponent ter njeni ponovni uporabi. V tem poglavju predstavim model sodobne knjižnice in podrobno opredelim osnovne procese njenega delovanja, s posebnim poudarkom na medknjižničnem povezovanju.

V šestem poglavju *Pogodbe in licence za ponovno uporabo* predstavim koncepte pogodb za ponovno uporabo. Te pogodbe nastajajo kot rezultat ali posledica medsebojnih sodelovanj med udeleženci v procesu ponovne uporabe. Zgradim ponovno uporabna ogrodja najpogostejših vrst pogodb, to so licenčna pogodba med knjižnico in ponovnim uporabnikom, licenčna pogodba med razvijalcem ali posrednikom komponente in knjižnico, pogodba o včlanitvi v knjižnico ter pogodba o vzdrževanju. Izdelam tudi predlog za avtomatizacijo postopka podeljevanja ali sklepanja licenc, ki bi ta, sicer zelo administrativen in dolgotrajen postopek ter zaradi narave licenčnih pogodb (številne določbe, težko berljiv tekst ipd.) za marsikaterega ponovnega uporabnika celo odbijajoč, poenostavil in pohitril.

V zaključku sem strnila spoznanja in ugotovitve, ki sem jih pridelala v disertaciji, podala nekaj iztočnic za nadaljnje delo ter predlogov, ki bi pravnim strokovnjakom bili v pomoč pri bodisi preoblikovanju obstoječe ali oblikovanju nove zakonodaje ali pravnega sistema intelektualne lastnine.

2 PONOVNA UPORABA IN PONOVNO UPORABNE KOMPONENTE

2.1 DEFINICIJE OSNOVNIH POJMOV

Skozi desetletja so se vzporedno z naraščanjem pomena razvoja programskih sistemov z že razvitimi komponentami postopoma izoblikovale tudi številne definicije pojmov, ki so se nanašale na ponovno uporabo, proces ponovne uporabe, ponovno uporabne komponente, njihovo upravljanje, njihove knjižnice ipd. Ker pa so bile definicije navadno tesno povezane z nekim konkretnim pristopom, metodologijo, konkretnim avtorjem ali skupino avtorjev, lahko danes v literaturi še vedno zasledimo množico izrazov za en in isti pojem, včasih pa celo izraz, ki se pojmuje na različne načine, kar pomeni, da standardizacije izrazoslovja na tem znanstvenem področju, razen redkih poskusov, še ni. Da bi bilo možno vsebino tega dela razumeti nedvoumno, so v nadaljevanju podani opisi in definicije teh pojmov, tako kakor so obravnavani in videni v tem delu.

2.1.1 Ponovna uporaba

V danes zelo številnih prispevkih s področja ponovne uporabe, komponentnega razvoja programskih sistemov, sestavljanja sistemov iz vnaprej izdelanih komponent, sodobnih pristopov h gradnji spletnih aplikacij, pristopov k izboljšanju učinkovitosti programskega razvoja ipd., je mogoče zaslediti različne definicije ponovne uporabe, od katerih navajam samo najznačilnejše. McGregor in Sykes [29] sta ponovno uporabo definirala tako: *Ponovna uporaba je uporaba kateregakoli izdelka, ki je bil predhodno že uporabljen.* Software Productivity Consortium [49] je ponovno uporabo definiral kot *uporabo nekega izdelka*[2] *pri reševanju različnih problemov ali različicah istega problema.* Jacobson [19] podobno definira

[2] V originalu uporabljen izraz *asset.*

ponovno uporabo kot *nadaljnjo ali ponavljajočo se uporabo nekega izdelka*[3]. *Praviloma so programski izdelki načrtovani za uporabo pri gradnji sistemov zunaj njihovega originalnega konteksta.* Jakkola [20] obravnava ponovno uporabo kot *implementacijo nove programske opreme z uporabo oziroma izkoriščanjem obstoječih izdelkov*[4]. Zelo zanimivo in v primerjavi z zgornjimi definicijami alternativno definicijo pa podaja Bassett [4], ki pravi, da je ponovna uporaba *proces prilagajanja posplošene komponente različnim kontekstom uporabe.* Ta se od večine razlikuje po tem, da šteje za ponovno uporabo samo tako uporabo, kjer pride do kakršnekoli spremembe ali prilagoditve komponente (siva in bela ponovna uporaba), ne pa ponov(lje)ne uporabe, pri kateri se komponenta uporabi brez spreminjanja (črna ponovna uporaba).

Definicija ponovne uporabe, ki je v tem delu uporabljena kot izhodiščna, povzema navedene in številne druge:
Ponovna uporaba predstavlja pristop h gradnji oziroma sestavljanju programskih sistemov ali aplikacij iz ponovno uporabnih komponent, to je programskih izdelkov, ki so bili načrtno zgrajeni za večkratno ponovno uporabo v različnih kontekstih, pri čemer lahko te izdelke uporabimo nespremenjene ali jih prilagodimo konkretnemu kontekstu.

V delu je tudi privzeto, da so izrazi *programska oprema*, *programski sistem* in *aplikacija* sinonimi in predstavljajo tisti programski izdelek, ki nastane kot rezultat programskega procesa ali razvoja. V kontekstu ponovne uporabe pa gre za programski izdelek, načeloma sestavljen iz vnaprej pripravljenih programskih komponent, ki jih med seboj ustrezno povežemo v celoto in ga lahko zato poimenujemo izpeljano delo.

2.1.2 Ponovno uporabni izdelki ali ponovno uporabne komponente

Iz zgornje definicije ponovne uporabe bi lahko sklepali, da je ponovna uporabna komponenta v bistvu *katerikoli programski izdelek ali produkt katerekoli razvojne faze, ki je bil izdelan ali zgrajen z namenom, da bo ponovno uporabljen tudi v drugačnih kontekstih od tistega, v*

[3] V originalu uporabljen izraz *artifact.*
[4] V njegovih definicijah zasledimo izraza *objects of reuse* in *deliverables of a software (project).*

katerem je nastal. Zgrajen ali opremljen pa mora biti tako, da ga lahko poljubni ponovni uporabnik razume in uporabi brez avtorjeve pomoči.

V kontekstu ponovne uporabe se izraz *izdelek*[5] (ang. *asset, artifact, object of reuse, deliverable of software ipd.*) najpogosteje uporablja kot sinonim že skoraj ustaljenega izraza *ponovno uporabna komponenta* (ang. *reusable component*). Glede na to, da so izdelki navadno sestavljeni ali narejeni iz različnih komponent ali sestavnih delov[6], in da se pojem izdelka navadno uporablja v kontekstu nekega končnega izdelka, ki nastane kot rezultat bodisi umskega bodisi ročnega ali strojnega dela [148], se v disertaciji uporablja izraz ponovno uporabna komponenta ali velikokrat kar samo komponenta. Privzeto pa je, da se sistematična in popolna ponovna uporaba ne omejuje samo na ponovno uporabo kode (izvršljivih komponent)[7], ampak da je možno ponovno uporabiti vse vrste komponent, ne glede na njihovo zrnatost, spremenljivost, vidljivost ali vrsto. Lahko gre torej za dokumentacijo, načrte, modele, preskusne primere, plane, izvršljivo kodo, specifikacije, zahteve, arhitekture, podatke, ocene, uporabniške vmesnike itn. ali celo konstruktivne ideje in izkušnje v obliki vzorcev [11]. Izraz programska komponenta ali programski izdelek se torej nanaša na katerikoli izdelek, vmesni ali končni, ki nastane med razvojem programskega sistema ali znotraj programskega procesa.

Ponovno uporabno komponento vedno obravnavamo kot nabor, skupek ali paket elementov. Kateri elementi bodo sestavljali komponento, je odvisno od vrste komponente oziroma njenega namena in vsebine. Bistveno je, da so ti elementi izbrani tako, da čim bolje podajajo namen in funkcijo same komponente ter vse potrebno za uspešno, učinkovito in nesporno[8] ponovno uporabo komponente. Elementi so lahko fizično shranjeni skupaj, na istem mestu, ali pa ne. Glede na to, da so lahko posamezni elementi (modeli, preskusni primeri, zahteve, ipd.) vsebovani v različnih komponentah, menim, da je v izogib podvajanju ali redundantnosti in za povečanje obsega ponovne uporabe (tudi te elemente je možno ponovno uporabiti) smiselno

[5] V nekaterih slovenskih prispevkih, npr. [11], zasledimo namesto izraza izdelek izraz proizvod programskega procesa.
[6] V RAS [48] se izraz *asset* uporablja za ponovno uporabno komponento, ki pa je sestavljena iz različnih delov, imenovanih *artifacts* (dokumentacija, modeli, izvorna koda, zahteve, preskusni primeri, ipd.). Fizično vsak *artifact* predstavlja datoteko.
[7] Tako ponovno uporabno komponento definira komponentna tehnologija.
[8] Nesporno predvsem v pravnem smislu.

tudi sestavne dele obravnavati kot ponovno uporabne komponente in jih med seboj povezovati s kazalci.

Da bi obdržali pomen komponente kot osnovnega gradnika, v delu privzemam, da programski sistemi kot celote ali aplikacije, čeprav so ravno tako programski izdelki, ne predstavljajo ponovno uporabnih komponent. Resda pri ponovni uporabi načeloma nismo omejeni z velikostjo komponente, vendar tako velikih komponent ni smiselno ponovno uporabiti, saj njihovo prilagajanje novim zahtevam terja preveč časa in stroškov, zato njihova ponovna uporaba pravzaprav ni več smiselna.

Po [48] so zrnatost[9] (ang. *granularity*), spremenljivost (ang. *variability*), vidljivost ali vidnost (ang. *visibility*) [10] in razčlenjenost[11] (ang. *articulation*) najpomembnejše lastnosti ali dimenzije, ki opisujejo in določajo ponovno uporabno komponento. Stopnja ponovne uporabnosti neke komponente je v veliki meri odvisna od specifikacije in dokumentacije, ki komponento spremljata. Zelo pomembno je, da je komponenta opremljena na tak način, da jo je mogoče hitro najti, ugotoviti njeno funkcionalnost, ugotoviti, kje in kako jo je mogoče ponovno uporabiti ter kakšni so bili rezultati predhodnih ponovnih uporab te komponente, kako je z lastništvom komponente, kako je s pravicami intelektualne lastnine nad komponento ipd. (glej poglavje 5.2). Opremljenost komponente z vsemi temi informacijami povečuje

[9] *Zrnatost* opisuje, koliko posebnih problemov lahko z neko komponento rešimo, oziroma koliko alternativnih rešitev neka komponenta omogoča. Najpreprostejša komponenta rešuje le en, navadno dobro opredeljen problem. Z naraščanjem zrnatosti narašča število problemov, ki jih lahko rešimo s komponento, oziroma se povečuje število različnih rešitev istega problema. S stopnjevanjem zrnatosti komponente naraščata tudi njena velikost in kompleksnost. V večini literature zasledimo za ti skrajnosti pojma grobe in fine zrnatosti.

[10] *Spremenljivost* in *vidljivost* ali *vidnost* komponente se nanašata na možnost izvedbe sprememb in prilagoditev nad komponento in na to, kako se izvedbene podrobnosti oziroma notranjost komponente odražajo navzven. Eno skrajnost predstavljajo komponente, ki jih ni mogoče na noben način bistveno spremeniti, ker njihova notranjost ni vidna (enkapsulacija) in se jih ne da prilagoditi drugače kakor prek vmesnika. To so t.i. črne komponente (ang. *black-box assets*). Druga skrajnost so bele komponente (ang. *white-box assets*), katerih vidljivost je popolna in jih je mogoče poljubno spreminjati ter prilagajati. Take komponente navadno spremljajo tudi specifikacija zahtev, modeli in druge datoteke. Med belimi in črnimi komponentami so še prozorne (ang. *clear-box assets*) in sive komponente (ang. *gray-box assets*). Izvedbene lastnosti prozorne komponente so vidne samo prek določenih delov kode ali dokumentacije, kar omogoča boljše razumevanje delovanja komponente ter njeno učinkovitejšo ponovno uporabo, a jih ni mogoče spreminjati. Pri sivih komponentah je spreminjanje omogočeno samo nad podmnožico elementov, ki to komponento sestavljajo oziroma tvorijo. Spreminjanje je možno prek parametrov komponente.
Lastnosti vidljivosti in spremenljivosti sta med seboj zelo povezani in odvisni, sta pa tudi ključni pri definiranju različnih strategij ponovne uporabe.

[11] *Razčlenjenost* opisuje stopnjo popolnosti zagotavljanja (podajanja) rešitve problema posameznih delov ponovno uporabne komponente. Če posamezni deli samo specificirajo rešitev in je ne podajajo, je stopnja razčlenjenosti nizka. Ko posamezni deli rešitev tudi implementirajo in jo ustrezno dokumentirajo, je stopnja razčlenjenosti visoka.

njeno kakovost, zato sta večja zaupanje razvijalcev v komponento in stopnja njene ponovne uporabnosti.

Poleg ustreznega načrtovanja ponovno uporabne komponente in njene dokumentacije je za resnično ponovno uporabnost neke komponente treba to komponento najprej nekajkrat ponovno uporabiti, po možnosti v različnih kontekstih. Zato je zbiranje in shranjevanje informacij o ponovnih uporabah (izkušenj pri ponovnih uporabah komponente) bistveno za nadaljnje izboljšave komponente, njeno nadaljnjo ponovno uporabo in za kakovost knjižnice.

2.1.3 Knjižnice ponovno uporabnih komponent in komponentna tržišča

Ponovna uporaba brez ponovno uporabnih komponent ni mogoča. Za razpoložljive komponente morajo razvijalci sistemov vedeti, torej morajo obstajati katalogi, iz katerih je mogoče izbirati komponente, ki jih potrebujemo. Izbiranje pa je omejeno in poteka "na pamet", če poleg komponente nimamo na voljo dovolj informacij o tem, kaj in kako komponenta dela ter čemu je namenjena. Zato je treba komponente na ustrezen način povezati, jih opremiti z informacijami in organizirati. V ta namen so se začele graditi knjižnice ponovno uporabnih komponent ali komponentne knjižnice (ang. *component library* ali *reuse library* ali *library of reusable components* ali *asset library* ipd.), za katere se pogosto uporabljata tudi izraza repozitorij (ang. *repository*) in bazen (ang. *pool*) ponovno uporabnih komponent.

Osnova za organizacijo knjižnice in za postavitev njene strukture bi morali biti modeli domene[12]. Lahko bi torej rekli, da je komponentna knjižnica rezultat analize domene. Dober nabor komponent ali zbirk komponent pa sam po sebi še ne zagotavlja tudi visoke stopnje ponovne uporabe teh komponent. Knjižnica mora biti zgrajena tako, da se jo lahko uporabniki hitro naučijo uporabljati. Eden od načinov je gotovo strukturiranje knjižnice po strukturi domene ter standardizacija poimenovanj komponent po konceptih domene.
Komponentna knjižnica mora svojim uporabnikom postreči tudi z natančno definiranimi pravili delovanja, katalogom, postopki klasificiranja oziroma razvrščanja in certificiranja,

[12] Organizacija oziroma povezovanje zbirk komponent v strukturo mora odražati medsebojne odnose med komponentami, saj so take zbirke veliko bolj ponovno uporabne od zbirk s poljubno strukturo.

orodji in mehanizmi, ki omogočajo hitro in preprosto iskanje komponent, vizualizacijo njihovega delovanja ali pregledovanje njihove vsebine, sledenje komponentam in njihovim različicam, poročanje uporabnikov knjižnice, in drugimi storitvami, ki lajšajo in spodbujajo ponovno uporabo.

Pred leti, ko so se pojavili prvi zametki komponentne tehnologije, je bilo skoraj nemogoče pričakovati, da bi na tržišču, torej izven razvojne skupine, uspeli najti ustrezno komponento, ki bi jo bilo mogoče kar ponovno uporabiti, ne da bi vanjo posegali. Vzrok za to je predvsem pomanjkanje standardiziranih vmesnikov. V zadnjem desetletju pa se veliko pozornosti posveča ravno standardizaciji ponovno uporabnih arhitektur[13]. S temi arhitekturami je ponovna uporaba izdelkov med razvojnimi skupinami postala mogoča, in to ne samo znotraj ene domene ali aplikacijskega področja, ampak med različnimi domenami in razvojnimi okolji. Ta izmenjava se je danes že tako razmahnila, da lahko govorimo o tržiščih programskih komponent. Močan razlog za uspeh te tehnologije je tudi v dejstvu, da so pristopi komponentne tehnologije osnova tudi za gradnjo spletnih storitev.

Knjižnice se torej vse pogosteje odpirajo navzven in se povezujejo v komponentna tržišča (ang. *component marketplace*), ki predstavljajo osrednji vir informacij za proces produkcije ali izdelave komponent ter za proces uporabe komponent. Nastajanje komponentnih tržišč je logična posledica razvoja informacijskih tehnologij ter vse večje potrebe po pripravljenih komponentah, ki se dajo preprosto integrirati v razvijajoči se sistem.

2.1.4 Tehnologije ponovne uporabe

Ponovna uporaba ni posebna metodologija ali tehnologija razvoja, pač pa je pristop h gradnji programskih sistemov, ki so ga številne tehnologije privzele kot osnovo. V sodobni literaturi sta gotovo najbolj znani in najbolj uporabljani t.i. komponentna tehnologija CBSD[14], ki je nastala kot "podaljšek" ali nadgradnja objektno-usmerjenega pristopa in tehnologija razvoja

[13] CORBA (*Component Object Request Broker Architecture*), Java EJB (*Enterprise JavaBeans*) ali J2EE (*Java2 Enterprise Edition*), Microsoft .NET ali Microsoft COM (*Component Object Model*), Microsoft Transaction Server/COM+, Borland Delphi VCL/CLX, ki so postali industrijski standardi.
[14] *CBSD - Component Based Software Development*.

na temelju ponovno uporabnih izdelkov ABSD[15]. Osnova obeh je ponovna uporaba znotraj dobro definiranega in dokumentiranega razvojnega procesa. Kot osnovno razliko med njima bi lahko izpostavili različno pojmovanje tistega, kar je mogoče ponovno uporabiti, torej definicijo ponovno uporabne komponente. Komponentna tehnologija CBSD se omejuje na objektni razvoj in privzema, da je ponovno uporabna komponenta bodisi v naprej pripravljena izvršljiva enota (izvršljiva koda), torej črna komponenta, bodisi ponovno uporabno ogrodje (ang. *framework*), ki zagotavlja arhitekturo, znotraj katere se komponente – objekti "zlagajo" v sistem ali aplikacijo. Tehnologija razvoja na temelju ponovno uporabnih izdelkov pa dovoljuje oziroma podpira ponovno uporabo tudi belih, prozornih in sivih komponent in se pri tem ne omejuje samo na izvršljive komponente, ampak na vse možne vrste programskih komponent. Pravzaprav ne govori o ponovno uporabnih komponentah, ampak o izdelkih.

Leta 1997 je Bassett [4] definiral FBSD tehnologijo[16], v kateri definira ponovno uporabo ogrodij v neobjektnem okolju. Ogrodja, kot jih definira Bassett, so splošni razredi, opremljeni z mehanizmi za aktivno prilagajanje, za katere pravi, da so bolj učinkoviti kot standardni mehanizmi dedovanja v okolju objektno usmerjenih jezikov.

V pričujočem delu je s pojmom *razvoj na temelju ponovne uporabe* oziroma gradnja/razvoj/sestavljanje sistemov ali aplikacij iz ponovno uporabnih komponent (ali krajše komponentni razvoj) opisan razvoj programskih sistemov ali izdelkov (ne nujno objektno usmerjen) na temelju sestavljanja tistih ponovno uporabnih komponent, ki se dajo ponovno uporabiti brez prilagajanja ali z manjšimi prilagoditvami (črna in siva ponovna uporaba). Le izjemoma oziroma le tedaj, ko je možno te spremembe dokaj preprosto izvesti (bela ponovna uporaba), naj bi se izvedla ponovna uporaba komponent z večjimi spremembami. Komponentni razvoj pa ne zajema samo procesov neposredne ponovne uporabe, ampak tudi procese ustvarjanja in upravljanja ter vzdrževanja ponovno uporabnih komponent. Ko so v pričujočem delu omenjene *komponentne tehnologije*, so torej s tem mišljene *tehnologije, ki predvidevajo razvoj sistemov ali aplikacij na temelju ponovne uporabe, ne glede na izbrano metodologijo razvoja in/ali izbrani življenjski cikel*.

[15] *ABSD – Asset Based Software Development*.
[16] *FBSD - Frame Based Software Development*.

2.1.5 Proces ponovne uporabe

Bistvo vseh prednosti, ki jih je mogoče s ponovno uporabo doseči, leži torej v dobro organiziranem procesu, znotraj katerega poskrbimo za razvoj povsem novih ponovno uporabnih komponent, odkrivanje komponent, ki se dajo preoblikovati v ponovno uporabne, odkrivanje možnosti za ponovno uporabo in ponovno uporabo vsega, kar je mogoče ponovno uporabiti v vseh razvojnih fazah ter za ustrezno upravljanje ponovno uporabnih komponent.

V zadnjih letih je bilo v podporo ponovni uporabi razvitih tudi veliko orodij, metrik, metod in pristopov, ki omogočajo, lajšajo in spodbujajo ponovno uporabo na vseh nivojih in v vseh fazah razvojnega procesa. Motivacija za to je zelo močna, saj lahko z organizirano in sistematično ponovno uporabo dosežemo bistvene prednosti in v veliki meri odpravimo pereče probleme razvoja programskih sistemov zadnjega desetletja[17].
Proces vzpostavitve sistematične ponovne uporabe, ki temelji na dobro organiziranem in dobro definiranem procesu ponovne uporabe ter temeljitem planu ponovne uporabe, ki ga organizacija postavi glede na svojo stopnjo razvoja, znanja, sposobnosti, kadre, orodja, poslovne cilje, strategije itd., je dolgotrajen in zahteva visoka začetna vlaganja. Uspešna implementacija ponovne uporabe zahteva postavitev ustrezne infrastrukture za ponovno uporabo. Posebno kritičen vidik te infrastrukture pa predstavlja proces ponovne uporabe.

Najbolj splošno lahko proces ponovne uporabe razčlenimo na štiri osnovne aktivnosti ali podprocese, katerih poimenovanja so v različnih virih različna (podroben opis v 4. poglavju):

1. Produkcija ali razvoj ponovno uporabnih komponent / inženirstvo domene / razvoj za ponovno uporabo: vnaprejšnja izdelava ali reinženiring (retrogradnja) komponent za ponovno uporabo. Glavni aktivnosti v ta namen sta analiza in inženirstvo domene.

   Analiza domene je proces, znotraj katerega se identificirajo, zbirajo, organizirajo, analizirajo in predstavljajo podobnosti ter razlike med obstoječimi sistemi neke aplikacijske domene. Na temelju te analize se definirajo programske arhitekture ali ogrodja, ki podpirajo gradnjo novih sistemov ali aplikacij te domene.

[17] Prednosti se nanašajo predvsem na zmanjševanje stroškov, povezanih z razvojem, zmanjševanje stroškov vzdrževanja obstoječih sistemov, skrajševanje razvojnih časov, časov plasiranja sistemov na tržišče, povečevanje produktivnosti, kakovosti in konsistentnosti sistemov, zmanjševanje tveganj, izboljšanje funkcionalnosti in/ali učinkovitosti sistemov, če naštejem samo najbistvenejše [seznam A].

Inženirstvo domene zajema gradnjo ponovno uporabnih komponent in izdelkov, metod, orodij in spremljajoče dokumentacije, namenjenih reševanju problemov med razvojem sistema oziroma podsistema z uporabo znanja o modelih domene in programskih arhitekturah.

2. Uporaba ponovno uporabnih komponent / razvoj aplikacij iz ponovno uporabnih komponent / inženirstvo aplikacij / razvoj s ponovno uporabo: nastopi pri gradnji novih sistemov iz obstoječih komponent ali pri vzdrževanju oziroma posodabljanju obstoječih sistemov s preizkušenimi in kakovostnimi ponovno uporabnimi komponentami.
3. Vzdrževanje ali posredovanje ponovno uporabnih izdelkov / upravljanje komponent / upravljanje knjižnice ponovno uporabnih komponent.
4. Proces upravljanja infrastrukture za ponovno uporabo / podporni procesi ponovni uporabi: so osnova celotnemu procesu ponovne uporabe. Vključujejo določitev pravil ponovne uporabe, vlog v procesu ponovne uporabe in ciljev ponovne uporabe, načrtovanje dogovorov in standardov, potrditev dodajanja, brisanja in spreminjanja komponent knjižnice, naročanje ponovno uporabnih komponent, koordiniranje časovnih aktivnosti in virov ter usklajevanje ciljev ponovne uporabe s poslovnimi cilji organizacije.

2.1.6 Udeleženci v procesu ponovne uporabe

Med udeležence v procesu ponovne uporabe lahko prištevamo vse tiste, ki kakorkoli sodelujejo v razvojnem procesu. Konceptualno bi jih lahko razdelili v tri podskupine: tiste, ki sodelujejo na strani produkcije ponovno uporabnih komponent, torej *razvijalci in dobavitelji ponovno uporabnih komponent*, tiste, ki sodelujejo na strani uporabe komponent, torej ponovni uporabniki oziroma *razvijalci aplikacij* ali programskih sistemov, ki sem jih poimenovala *sestavljavci aplikacij*, ter tiste, ki skrbijo za same komponente in njihovo distribucijo oziroma posredovanje s strani produkcije na stran uporabe, to so *upravitelji komponent* oziroma knjižnično osebje. Posredno sodijo med udeležence v procesu ponovne uporabe tudi *uporabniki aplikacij ali programskih sistemov*, ki so bili zgrajeni iz ponovno uporabnih komponent.

Velikokrat, posebno v manjših razvojnih organizacijah, se vloge posameznih udeležencev prepletajo: razvijalci komponent so istočasno tudi razvijalci aplikacij, morebiti pa tudi sami

skrbijo za interno knjižnico ponovno uporabnih komponent. Pogosto se poenotita tudi vlogi razvijalca komponent ter njihovega posrednika.

V pričujočem delu so posamezne vloge obravnavane z naslednjo razlago:

- *razvijalci komponent* so (navadno) inženirji domene, ki dobro poznajo potrebe komponentnega tržišča znotraj domene ter gradijo ponovno uporabne komponente. Te so lahko razvite na novo ali izpeljane iz drugih komponent ali pridobljene iz obstoječih aplikacij domene;
- *dobavitelji (posredniki) komponent* so tisti, ki skrbijo za trženje ponovno uporabnih komponent. Lahko so zaposleni v isti organizaciji kakor razvijalci teh komponent, lahko pa igrajo podobno vlogo kot založniki v svetu tiskanega gradiva. Predstavljali naj bi vmesni člen med razvijalci komponent in knjižnico oziroma knjižnicami;
- *sestavljavci aplikacij ali ponovni uporabniki* so strokovnjaki, ki razvijajo aplikacije iz ponovno uporabnih komponent. Te lahko pridobijo neposredno od razvijalcev oziroma dobaviteljev komponent, bolje pa je, če jih poiščejo v ustrezni domenski knjižnici (večja izbira, zagotovljena določena kakovost ipd.);
- *upravitelj komponent* je tisti, ki sodeluje pri organizaciji in delovanju knjižnice ponovno uporabnih komponent.

2.1.7 Problemi, povezani s ponovno uporabo

Kljub številnim prednostim in izboljšavam, ki jih je mogoče doseči s sistematično ponovno uporabo, pa lahko v literaturi s tega področja (tako v znanstvenih raziskavah kakor izsledkih iz prakse) zasledimo navedenih kar nekaj problemov, povezanih s ponovno uporabo. Problemi so netehnične ali tehnične narave, velikokrat pa je njihova narava tudi dvojna. Gre za upravljavske, ekonomske in pravne probleme na eni ter čiste tehnične in tehnološke na drugi strani. V tem delu je poseben poudarek namenjen pravnim problemom, za katerih reševanje je potrebnih tudi kar nekaj vzporednih ali predhodnih tehničnih rešitev. So pa v delu omenjeni tudi vsi preostali, kajti ravno tako kakor pravni omejujejo ponovno uporabo.

Nekaj osnovnih problemov je opisanih že v [58], kot najpogosteje omenjene pa velja izpostaviti naslednje:

- Načrtovanje ponovno uporabne komponente je zahtevno in dolgotrajno, kar še dodatno povečuje že tako visoke stroške vzpostavitve procesa sistematične ponovne uporabe. Posledično je treba kar nekaj časa čakati na povrnitev vloženih sredstev.
- Nujno potreben element v procesu ponovne uporabe je knjižnica ponovno uporabnih komponent. Iskanje ustreznih komponent ni vedno preprosto, še posebno, ko je v knjižnici veliko komponent. Problemi se pojavijo tudi pri izbiranju najustreznejše komponente ter njeni poznejši prilagoditvi, če komponento ne spremlja dovolj kakovostnih informacij, na temelju katerih se lahko ponovni uporabnik odloča ali izvede prilagoditve. Zato mora knjižnica poleg drugega ponujati tudi sredstva za hitro iskanje, sredstva za iskanje podobnih komponent, pa tudi sredstva, ki omogočajo prikaz delovanja komponente ali njene funkcionalnosti.
- Nezaupanje razvijalcev v komponente, ki so jih razvili drugi, je lahko ključen problem. Največkrat se nanaša na strah, povezan z morebitnim nedelovanjem ali nepravilnim delovanjem ponovno uporabljene komponente.
- Poleg funkcionalnih zahtev za komponento in/ali sistem je treba pri sistematični ponovni uporabi obravnavati tudi nefunkcionalne zahteve, ki se nanašajo na kakovost komponente in/ali sistema. Med slednjimi so poleg običajnih, kot so učinkovitost, združljivost, prenosljivost, zanesljivost, preprosta uporaba ter prihranek časa pri uporabi, za ponovno uporabo še posebno pomembni prilagodljivost, razširljivost, ponovna uporabnost in robustnost [57], ki pa jih je zelo težko doseči in meriti.
- Četudi je komponenta načrtovana za ponovno uporabo, se njena ponovna uporabnost izkaže šele po nekajkratnih uporabah. V tem času pa se lahko odkrije tudi njeno nepravilno delovanje, ki ga je treba odpraviti, skupaj z vsemi morebitnimi posledicami, ki so medtem nastale.
- Ker je med ponovno uporabo dovoljeno komponente tudi spreminjati, se lahko kaj hitro pojavi "poplava" različic neke komponente, zato morajo biti vzpostavljeni mehanizmi za obvladovanje različic.
- Vzdrževanje sledi o ponovnih uporabnikih in posameznih ponovnih uporabah je pomembno s pravnega in tehničnega zornega kota.

2.1.8 Predpogoji za uspešno ponovno uporabo

Glede na pomembnost ponovne uporabe pri sodobnem razvoju programskih sistemov na eni strani in glede na kompleksnost samega procesa mora biti za dosego sistematične ponovne uporabe izpolnjenih kopica predpogojev, ki so rabili tudi kot osnova oziroma smerokaz pri oblikovanju v tem delu predlaganih metod, modelov in postopkov:

- ponovno uporabo je treba planirati, s čimer zmanjšamo tveganja, ki so povezana z uvajanjem ponovne uporabe; izvedbeni plan programa ponovne uporabe mora vključevati proces ponovne uporabe, katalog komponent, infrastrukturne in metodološke spremembe, ki bodo omogočale doseganje v programu zastavljenih ciljev;
- pred pričetkom izvajanja programa ponovne uporabe mora organizacija doseči določen nivo komponentnega razvoja[18] ter identificirati svoje možnosti (viri, osebje, aktivnosti);
- natančno je treba definirati sam proces ponovne uporabe, ki mora vključevati modeliranje domene in arhitekture za domeno; analiza domene mora biti osnova za identifikacijo tistega, kar je v domeni že ponovno uporabnega, in tistega, kar še ni;
- izbirati je treba "čiste" metodologije, kajti tiste, ki združujejo in prepletajo aktivnosti produkcije in uporabe komponent, so zelo kompleksne in komplicirane;
- v čim večji meri je treba ponovno uporabljati izkušnje[19];
- kot osnovni sestavni del infrastrukture za ponovno uporabo je treba izbrati ustrezna orodja, ki bodo lajšala in podpirala ponovno uporabo (glej priloga A), ter metrike, s katerimi je mogoče ovrednotiti posamezne funkcionalne in nefunkcionalne zahteve;
- razvojne metode morajo biti usmerjene v razvoj komponent z dobro definiranimi standardnimi vmesniki, prek katerih je mogoče komponente prilagajati;
- komponentne knjižnice morajo podpirati prikazovanje uporabe komponent z različnih zornih kotov, prikazovanje delovanja komponent, zbiranje in shranjevanje relevantnih informacij o komponenti, ki lajšajo njihovo iskanje, izbor in uporabo;
- vzpostaviti je treba mehanizme za zaščito pravic intelektualne lastnine;
- potrebna je standardizacija postopkov in opravil razvoja s in za ponovno uporabo, standardizacija delovanja komponentne knjižnice ter njenega povezovanja navzven z

[18] 5. nivo CMM (ang. *Capability Maturity Model*) - zrelostno zmožnostni model ali RMM (ang. *Reuse Maturity Model*) - zrelostno ponovno uporabni model [49].
[19] Izkušnje in znanje je mogoče vgraditi v komponente in spremljajočo dokumentacijo, s čimer se bistveno povečata kakovost ter ponovna uporabnost komponent, pojavi pa se problem varovanja intelektualne lastnine.

drugimi knjižnicami, uporabniškega vmesnika ter storitev knjižnice, izrazoslovja, vsebine ponovno uporabne komponente.

2.2 STRATEGIJE PONOVNE UPORABE

Vsaka organizacija, ki se ukvarja z razvojem programskih sistemov in želi to početi na temelju ponovne uporabe, se mora odločiti za strategijo, ki ji bo sledila. Strategija je seveda odvisna od stopnje razvoja procesov in orodij, nivoja ponovne uporabe, od zrelostnega nivoja, ki ga organizacija dosega, od finančnih zmožnosti organizacije, od zrelosti domene, za katero razvija programske sisteme, in drugih faktorjev. Načeloma začnejo organizacije izvajati ponovno uporabo sprva priložnostno, pozneje pa vedno več vlagajo v namensko gradnjo ponovno uporabnih komponent. Najprej jih shranjujejo neorganizirano, ko pa se ustvari dovolj velika kritična masa teh komponent, zgradijo interno ali notranjo knjižnico za lastne potrebe. Ob ugotovljeni potrebi po neki komponenti bodo razvijalci sistemov torej najprej preverili, ali lahko ponovno uporabijo kako domačo komponento. V kolikor take komponente v interni knjižnici ni, poskušajo bodisi najti vnaprej pripravljene komponente v kateri od domenskih komponentnih knjižnic, ki svoje komponente trži, ali pa se odločijo za razvoj komponente na novo. Na izbor možnosti vplivajo predvsem stroški, ki nastanejo pri posamezni obliki ponovne uporabe. Poleg podrobnejšega opisa teh stroškov so v nadaljevanju tega podpoglavja opisane tudi posamezne oblike ponovne uporabe, narejena je primerjava med njimi, podan in opisan pa je tudi model, na temelju katerega se lahko organizacija v danem trenutku odloča za vrsto ali strategijo ponovne uporabe. Kot osnova pri izbiri načina za ponovno uporabo neke komponente je privzeta filozofija *Reuse Before you Buy Before you Build*[20] (ponovno uporabi ali kupi, preden izdelaš sam).

2.2.1 Vrste ponovno uporabnih komponent in ponovnih uporab

Ponovno uporabne komponente bi lahko razvrstili v dve osnovni kategoriji. To sta kategorija belih in kategorija črnih ponovno uporabnih komponent. Bele komponente so v bistvu ali deli izvorne kode ali druge vrste komponent, ki se jih da tudi spreminjati in na tak način prilagajati novim ali spremenjenim zahtevam. Črne komponente so navadno v prevedeni ali binarni kodi,

[20] Blagovna znamka ComponentSource odprtega tržišča črnih (ang. off-the-shelf) komponent.

če gre za računalniške programe ali dele računalniških programov, zato jih ni mogoče neposredno spreminjati. Informacije o njihovi funkcionalnosti lahko razvijalci pridobijo iz priložene dokumentacije in prek javnega vmesnika komponente. Razširjanje ali spreminjanje njihove funkcionalnosti je možno bodisi prek vmesnika bodisi z ovijanjem komponente. V začetku 90-ih se je pojavil še en tip komponente, to so komponente COTS (ang. *Commercial Off-The-Shelf Components*) ali tržne komponente (ang. *Open Market Components*), ki jih je mogoče kupiti v komponentnih trgovinah. Te komponente slonijo na standardnih komponentnih arhitekturah[21]. Ker za ponovno uporabo takih komponent zadošča nakup same komponente, brez dodatne podpore ali drugih vrst storitev zanjo, se je teh komponent prijel (navzel) tudi izraz komponente "plug-and-play".

Upoštevajoč vrste ponovno uporabnih komponent in izsledke številnih prispevkov s področja ponovne uporabe [Seznam A] je mogoče ključne strategije ponovne uporabe razdeliti v tri večje skupine: belo ponovno uporabo, črno ponovno uporabo komponent iz interne knjižnice ter črno ponovno uporabo komponent s tržišča. V zadnjih letih postaja vse popularnejša oblika ponovne uporabe komponent, ki ji pravimo ponovna uporaba v obliki črne škatle (ang. *black-box reuse*) ali črna ponovna uporaba. Gre za način ponovne uporabe, pri kateri komponente uporabimo, take kakor so, ne da bi jih kakorkoli spreminjali ali prilagajali. Razvijalcem aplikacij zadostuje poznavanje funkcionalnosti programske komponente in načina njenega sodelovanja z okoljem, ni jim pa treba posegati v logiko in/ali kodo komponente.

Viri komponent, ki jih lahko ponovno uporabimo, so bodisi interne knjižnice (znotraj projekta ali znotraj organizacije), lahko so to komponentna tržišča črnih (*off-the-shelf*) komponent, lahko pa so tudi knjižnice komponent, ki sodijo v kategorijo javnodomenskih izdelkov ali proste programske opreme.

2.2.1.1 Bela ponovna uporaba

Bela ponovna uporaba omogoča razvijalcem prilagoditev komponent trenutnim zahtevam s spreminjanjem samih komponent (njihove izvorne kode ali teksta oziroma grafike, če ne gre

[21] Glej opombo 10.

za izvršljive komponente). Taka ponovna uporaba predstavlja zelo prilagodljiv način pridobivanja programskih komponent znotraj razvojnih projektov, saj obstoječe komponente priredimo novim zahtevam. To hkrati tudi maksimalno poveča možnosti ponovne uporabe neke komponente. Še vedno pa ostaja nerešen ključni problem, ki ga srečamo pri takih vrstah ponovne uporabe – spreminjanje programske kode, ki od razvijalca zahteva zelo visoko stopnjo poznavanja izvedbenih podrobnosti komponente. Tako poznavanje pa je možno le, če je bila komponenta razvita "doma" ali pa če komponento spremlja izjemno kakovostna razvojna dokumentacija, kar pa se v praksi navadno ne dogaja. Da stroški ponovne uporabe komponente z njenim prilagajanjem ne bi presegli stroškov razvoja nove komponente, je možna in smiselna le ponovna uporaba doma razvitih komponent[22]. Slabosti bele ponovne uporabe se skrivajo tudi v dejstvu, da naletimo ob vzpostavitvi procesa ponovne uporabe na zelo velike začetne stroške, povezane s polnjenjem knjižnice ponovno uporabnih komponent [42], s tem pa so povezani tudi problemi klasifikacije in iskanja teh komponent znotraj knjižnice ali med knjižnicami. Zelo pogosto je treba za izvajanje bele ponovne uporabe spremeniti tudi celoten razvojni proces in organizacijsko strukturo razvojnega oddelka [14]. Velik vpliv na stroške pri beli ponovni uporabi ima iskanje ustrezne komponente. Cilj iskanja je seveda najti komponento, ki bo čim bliže zahtevam in potrebam, na katere naletimo med razvojem. Navadno je rezultat iskanja množica komponent, iz katere je treba izbrati najustreznejšo. Iskanje pa lahko postane zelo problematično zaradi izjemnega števila komponent, ki se kar naenkrat znajdejo v knjižnici. Vsako prilagajanje in spreminjanje obstoječe komponente ima namreč za posledico nastanek nove ponovno uporabne komponente ali različico neke obstoječe komponente. Knjižnico torej hitro preplavijo komponente, ki so si tudi po funkcionalnosti zelo podobne.

2.2.1.2 Črna ponovna uporaba

Večino naštetih težav, ki jih prinaša bela ponovna uporaba komponent, je mogoče premostiti s črno ponovno uporabo, saj spreminjanje in prilagajanje najdenih oziroma izbranih komponent pri črni ponovni uporabi nista dovoljena. Zmanjšanje stroškov iskanja in prilagajanja sta sicer prednosti črne ponovne uporabe, kljub temu pa ima taka ponovna uporaba tudi določene

[22] Lahko je komponenta tudi razvita doma, pa ostanejo težave z nepoznavanjem te komponente enako velike, če jo uporabi nekdo, ki sploh ni sodeloval pri njenem razvoju ali znotraj razvojnega projekta, v katerem je nastala. Zato je treba tudi pri domačih komponentah poskrbeti za čimbolj kakovostno dokumentacijo.

pomanjkljivosti. V dosedanji praksi je bila ponovna uporaba komponent brez spreminjanja redka, kar je seveda botrovalo drastičnemu zmanjšanju stopnje ponovne uporabe ter posledično porajanju dvomov o smiselnosti in izvedljivosti same ponovne uporabe. Zato so kot kompromis nastale črne komponente, ki jih je mogoče ponovno uporabiti v celoti, brez spreminjanja in prilagajanja njihove "notranjosti", omejeno prilagajanje pa je mogoče le prek njihovih vnaprej definiranih parametrov oziroma vmesnika. S takim načinom prilagajanja je sicer mogoče povečati prilagodljivost komponent, a se stopnja ponovne uporabe s tem ne poveča bistveno in zaostaja za stopnjo bele ponovne uporabe.

2.2.1.3 Črna ponovna uporaba komponent s tržišča

Pridobivanje komponent s tržišč lahko stopnjo ponovne uporabe bistveno poveča. Razvijalci sistemov namreč iščejo med veliko večjim številom komponent, zato se možnost, da bodo med njimi našli ravno tako, ki popolnoma ali kar najbolje ustreza njihovim potrebam, poveča. Poveča pa se tudi verjetnost, da bo neka komponenta s tržišča večkrat (ponovno) uporabljena, kar poveča stopnjo ponovne uporabnosti.

2.2.2 Primerjava strategij ponovne uporabe

Da za neko razvojno okolje objektivno ocenimo najboljšo možno strategijo ali kombinacijo strategij, je treba poznati prednosti in slabosti posameznih strategij, predvsem pa stroške, ki posamezno strategijo spremljajo ali s katerimi se znotraj posamezne strategije srečujemo. Višina stroškov lahko v veliki meri vpliva na izbiro strategije ponovne uporabe.

2.2.2.1 Kriteriji primerjave strategij ponovne uporabe

Ponovna uporaba v katerikoli obliki je smiselna, le če so stroški ponovne uporabe ustrezne komponente nižji od stroškov razvoja te komponente na novo. Pred odločitvijo o ponovni uporabi neke komponente je torej treba znati oceniti *stroške ponovne uporabe*, ki pa so odvisni tudi od načina oziroma strategije ponovne uporabe. Kot ključna parametra pri ocenjevanju stroškov ponovne uporabe postavim strošek prilagajanja komponente ter strošek pridobivanja oziroma iskanja komponente.

2.2.2.1.1 Strošek prilagajanja

Ponovna uporaba je v bistvu poskus prilagoditve nekega izdelka, ki že obstaja, novim zahtevam. Da bi razvijalci to dosegli, morajo premostiti t.i. idejno razdaljo med dejanskimi lastnostmi in funkcionalnostmi komponente ter želenimi oziroma zahtevanimi. Večja je *prilagoditvena razdalja*, višji je strošek ponovne uporabe izdelka. Za natančnejšo analizo stroškov prilagajanja je treba preučiti različne elemente in vidike premoščanja vrzeli med specifikacijami ponovno uporabne komponente in nove komponente.

Pri ponovni uporabi pogosto naletimo na primer, ko ponovno uporabljeno komponento prenesemo v okolje, drugačno od tistega, v katerem in za katerega je bila komponenta razvita. Zato pride do razlik med ciljnim izvedbenim okoljem komponente in obstoječo komponento. Ker gre za razlike med kontekstom uporabe originalne komponente in kontekstom uporabe prilagojene komponente, poimenujemo ta element *kontekstna razdalja*. Velike razlike med kontekstoma zahtevajo veliko prilagajanja, to pa povzroča tudi visoke stroške prilagajanja.

Drugi pomemben element se nanaša na razlike med ciljno aplikacijsko domeno (domeno, znotraj katere bo komponenta ponovno uporabljena) in trenutno domeno (domena, za katero je bila komponenta razvita), zato jo poimenujemo *domenska razdalja*. V njej je zajeto tudi prilagajanje funkcionalnosti programske komponente za uporabo v določeni aplikacijski domeni. Najmanjše stroške prilagajanja imamo torej pri ponovni uporabi znotraj iste domene. Domensko razdaljo lahko predstavimo tudi s količino napora, ki ga zahteva spreminjanje funkcionalnosti obstoječe komponente, da bi jo prilagodili natančno določenim trenutnim zahtevam in potrebam, kontekstno razdaljo pa s količino napora, ki ga zahteva pretvorba obstoječe komponente v želeno izvedbeno okolje (druga platforma in/ali programski jezik).

Stroški prilagajanja bodo torej največji pri ponovni uporabi bele komponente zunaj domene, v kateri je bila razvita, najmanjši pa, ko ponovno uporabimo črno komponento znotraj domene, v kateri je bila ta komponenta razvita.

2.2.2.1.2 Strošek pridobivanja komponent

Strošek iskanja in cena komponente sta ključna stroška, ki skupaj oblikujeta strošek pridobivanja neke komponente. Med ponovno uporabo nekega programskega izdelka je treba največ napora vložiti v iskanje komponent, ki bodo čim bolje ustrezale specifikacijam potrebne komponente. Ne glede na vir komponent[23] mora razvijalec v procesu iskanja vedno izvesti isti postopek, in sicer pregledati klasifikacijske sheme, izvesti iskanje (zelo pogosto prek ključnih besed iz specifikacije) in med ponujenimi oziroma najdenimi kandidatnimi komponentami izbrati najprimernejšo (navadno je treba funkcionalnost ugotavljati iz dokumentacije ali primerov rabe, torej ročno). Kolikšen napor je za to potreben, je odvisno od števila vseh komponent v knjižnici in števila najdenih kandidatnih komponent. Seveda je s tem pogojen tudi strošek iskanja komponente, ki raste z velikostjo knjižnice. Ta strošek pa bo še večji pri iskanju komponente na tržišču.

Komponente iz domače knjižnice navadno ni treba plačati ali zanjo pridobiti licence. Ker so komponentne knjižnice zgrajene zaradi prodaje komponent, torej za ustvarjanje profita, se tu pojavijo zahteve po mehanizmih za zaračunavanje licenčnin, sklepanju licenčnih dogovorov ipd. Komponente, ki jih pridobimo s tržišč, bi zaradi potencialne večkratne prodaje (t.j. ponovne uporabe) morale biti cenejše od doma razvitih komponent. Zaradi še vedno nezadostne zrelosti komponentnih tržišč pa so stroški pridobitve komponent s tržišča vseeno večji od stroškov ponovne uporabe doma razvitih komponent. Stroške pridobivanja komponent s tržišča je možno zmanjšati z izbiranjem takih komponent, pri katerih so prilagoditvene razdalje majhne, saj so posledično tudi stroški prilagajanja manjši. Torej je pred izborom komponente iz množice kandidatnih komponent, ki ustrezajo zahtevam po funkcionalnosti, treba oceniti še prilagoditvene razdalje (izbrati "najbližjo" zahtevam). Če je za ugotavljanje funkcionalnosti komponente na voljo ustrezno orodje, se strošek ocenjevanja prilagoditvene razdalje zmanjša.

V tabeli 1 so zbrane osnovne lastnosti posameznih obstoječih strategij ponovne uporabe, ki hkrati predstavljajo osnovo za izvedbo primerjave med njimi.

[23] Možni viri ponovno uporabnih komponent so domače knjižnice in komponentna tržišča.

Lastnosti	*Bela ponovna uporaba*	*Črna ponovna uporaba (komponentni razvoj)*	
		Razvoj domačih komponent	*Pridobivanje komponent s tržišča*
Definicija	Uporaba že obstoječih domačih programskih izdelkov pri gradnji novih sistemov, kjer je spreminjanje komponent (tudi programske kode) dovoljeno.	Uporaba že obstoječih domačih programskih izdelkov pri gradnji novih sistemov, kjer spreminjanje ni dovoljeno, prilagajanje komponent pa je možno samo z vnaprej definiranimi parametri.	
Stopnja ponovne uporabe	Stopnja ponovne uporabe je zaradi možnosti spreminjanja obstoječih komponent (za prilagoditev novim zahtevam) visoka.	Pogosto je stopnja ponovne uporabe nižja, ker morajo komponente natančno sovpadati z zahtevami.	Stopnja ponovne uporabe je zmerna, saj je malo verjetno, da na tržišču najdemo komponento, ki bo popolnoma ustrezala našim zahtevam.
Prilagodljivost uporabe komponent	Večja prilagodljivost pri ponovni uporabi, saj lahko razvijalci izbirajo znotraj širše množice potencialno ponovno uporabnih komponent.	Prilagodljivost je omejena, saj so možnosti prilagajanja odvisne od tega, kako dobro je vmesnik komponente parametriziran.	
Cilj ponovne uporabe	Lokalna ponovna uporaba, navadno omejena na razvojno skupino na isti lokaciji. Medorganizacijska delitev in izmenjava belih komponent nista običajni.	Ponovna uporaba je lokalna, vendar ni omejena na razvojne skupine na isti lokaciji. Medorganizacijska delitev in izmenjava komponent sta lažji kakor pri beli ponovni uporabi.	Komponente lahko pridobimo od prodajalcev. Na tržišču jih lahko kupujemo in prodajamo.
Razvojni stroški komponente	Nižji (vnaprejšnji) razvojni stroški.	Razvojni stroški so višji zaradi parametriziranja komponentinega vmesnika.	Visoki razvojni stroški zaradi parametrizacije.
Stroški ponovne uporabe komponent	Zmerni stroški pridobivanja, visoki stroški prilagajanja.	Nizki stroški pridobivanja in nizki stroški prilagajanja.	Visoki stroški pridobivanja in nizki stroški prilagajanja.
Kakovost	Kakovost komponente je organizaciji znana in jo je laže nadzorovati.	Kakovost komponente je znana in jo je laže nadzorovati.	Kakovost komponente ni nujno znana, zato obstaja določeno tveganje (uporaba popolnoma ali delno nekakovostnih programskih izdelkov).
Čas povrnitve stroškov	Ponovna uporaba postane ekonomsko učinkovita in opravičena šele, ko je v knjižnici dovolj velika kritična masa komponent. Zaradi vlaganja v ponovno uporabne komponente je zahtevana zmerna stopnja ponovne uporabe.		Vnaprejšnjega vlaganja v gradnjo knjižnice ni.
Problemi z upravljanjem knjižnice	Eksponentna rast velikosti knjižnice in nadzor nad različicami postaneta zelo velika problema.	Linearna rast velikosti knjižnice. Nadzor nad različicami je pomemben, vendar lažji, saj ne obstaja več izvodov iste komponente.	Omejena potreba po domači knjižnici, saj se tržišča obvezujejo, da bodo komponente katalogizirala in vzdrževala.

Tabela 1: Primerjava strategij ponovne uporabe.

Začetni razvojni stroški črne komponente so vedno višji od začetnih razvojnih stroškov bele komponente. Razlog tiči v tem, da je treba črno komponento zgraditi tako, da jo bo možno prilagajati konkretnim zahtevam prek parametrov vmesnika (parametrizacije), kar pomeni dodaten napor pri načrtovanju komponente. Ti dodatni stroški naj bi se amortizirali z večkratno ponovno uporabo črne komponente. Ekonomsko upravičenost ponovne uporabe neke črne komponente je torej mogoče doseči le z visoko stopnjo ponovne uporabe te komponente. Slednjo pa lahko dosežemo tako, da poleg ponovne uporabe komponente v hiši poskušamo komponento prodati na tržišču, če le to obstaja. S posredovanjem črne komponente na tržišče (iz hiše) je možno odpraviti nekatere bistvene pomanjkljivosti klasične črne ponovne uporabe, vendar večina današnjih tržišč komponent, posebno domensko specifičnih, kljub vsemu še ni dovolj zrela, saj je šele v fazi doseganja kritične mase

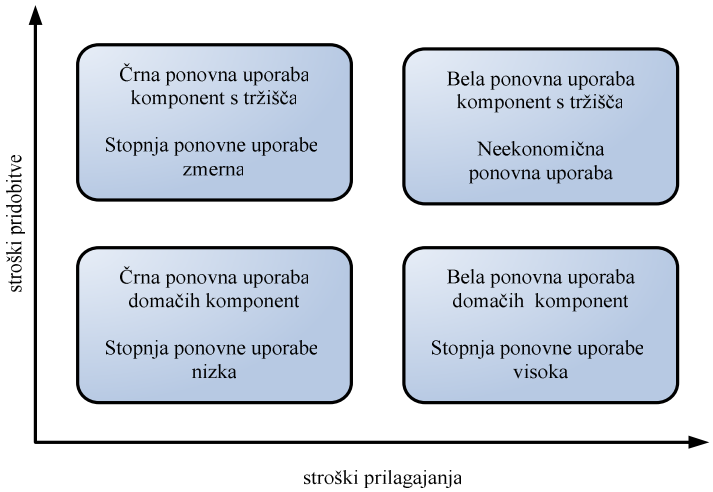
komponent. Ena od posledic nezrelosti komponentnih tržišč pa je visok strošek pridobivanja komponente, ki je posledica visokih stroškov iskanja komponente ter visokih cen komponent. Stroški iskanja komponente so visoki predvsem zato, ker ne obstajajo poenoteni mehanizmi iskanja po različnih tržiščih, pa tudi poenotene, standardizirane klasifikacijske sheme ne. Cene komponent pa so (pre)visoke predvsem zaradi neučinkovitosti komponentnih tržišč, kar je posledica premajhnega števila potencialnih kupcev komponent in zelo neučinkovitega posredovanja tržišč. Slabost komponentnih tržišč, ki trenutno obstajajo[24], je, da izvajajo samo distribucijo komponent, niso pa učinkoviti posredniki.

Na sliki 1 so prikazane lastnosti posameznih strategij ponovne uporabe, razporejene glede na strošek prilagajanja in strošek pridobivanja komponente.

Bela ponovna uporaba komponent s tržišča je v večini primerov stroškovno zelo nezanimiva, saj so stroški prilagajanja, tako kakor pri vsaki beli ponovni uporabi, relativno visoki. Ker pa je treba za komponente s tržišča tudi plačati (licenčnine), so stroški pridobivanja komponent (v primerjavi s komponentami iz interne knjižnice) visoki. Stroškovno se izkaže kot najbolj zanimiva možnost črne ponovne uporabe domačih komponent, velikokrat pa je pri taki ponovni uporabi problem številčno premajhen in vsebinsko omejen nabor komponent, ki so ciljno zelo ozko usmerjene (te komponente izhajajo iz predhodnih projektov). Zato je stopnja pri taki ponovni uporabi nizka.

Slika 1 hkrati predstavlja tudi ogrodje, na temelju katerega je mogoče razporejati oziroma določiti lastnosti strategijam ponovne uporabe glede na ti dve dimenziji stroškov, prikazala pa naj bi tudi stopnje ponovne uporabe pri posameznih strategijah. Poleg tega je s slike jasno razvidno, da je v primeru visokih stroškov prilagajanja in visokih stroškov pridobivanja komponent ponovna uporaba nesmiselna. V tem primeru se kot najboljša rešitev izkaže gradnja komponente na novo.

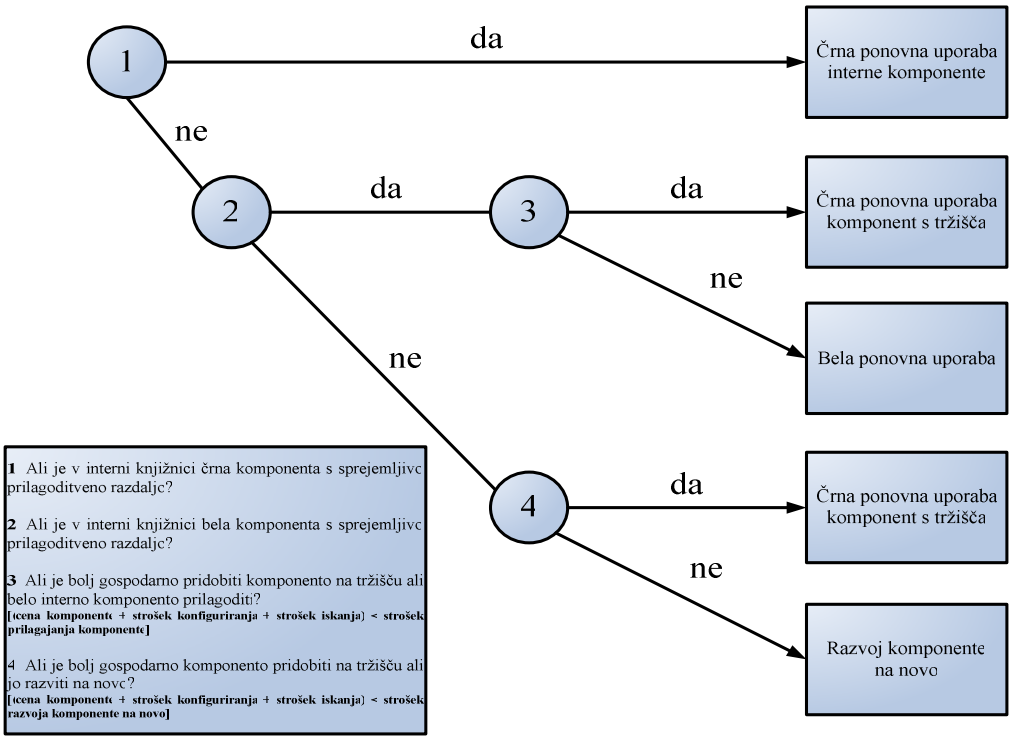
[24] V zadnjih letih se je razvilo le nekaj takih tržišč, ki dejansko prodajajo komponente zelo različne zrnatosti, od objektov za grafične uporabniške vmesnike do visoko zrnatih komponent z vgrajeno poslovno logiko. V zadnjem času so se začele pojavljati tudi tendence in poskusi razvoja domenskih komponentnih tržišč (npr. finančno področje, zavarovalništvo). Primeri nekaterih bolj znanih komponentnih tržišč: ComponentSource, združen s Flashline – komponente COTS za standardne komponentne arhitekture, NIST in NETLIB – znani zbirki matematične programske opreme, člankov in podatkovnih baz, SourceForge – tržišče odprtokodne programske opreme, Fortran90 SR, CMU AI Repository, Celestrack WWW SR in druga.



Slika 1: Strategije ponovne uporabe glede na stroške.

2.2.3 Izbira strategije ponovne uporabe

Glede na dane možnosti se mora torej razvojna skupina odločiti, katero strategijo bo izbrala. Odločitveno drevo (na komponentnem nivoju) na sliki 2 predstavlja model, ki je lahko pri sprejemanju take odločitve v pomoč.



Slika 2: Odločitveno drevo za izbiro načina ponovne uporabe komponent.

Model predstavlja različne strategije ponovne uporabe in vprašanja, na katera morajo razvijalci odgovoriti, preden se odločijo za katero od strategij. V drevesu sta zajeta tako strošek pridobivanja komponente kakor strošek prilagajanja komponente. V njem je podana tudi enačba za izračun stroškov, ki je razvijalcem v pomoč pri odgovarjanju na vprašanja. Iz enačbe je razvidno, da je cena komponente zelo pomemben faktor pri določanju, v kolikšnem obsegu je (še) smiselno v iskanje komponente vključiti komponentna tržišča.
Prvo vprašanje razvijalca se mora nanašati na to, ali so komponente s sprejemljivo prilagoditveno razdaljo v domači knjižnici. V tem primeru ni plačil, licenčnin in drugih stroškov. Pri izbiranju strategije za določanje oziroma ocenjevanje prilagoditvene razdalje je treba postaviti neko mejo sprejemljivosti, ta meja pa se pri črni ponovni uporabi lahko razlikuje od tiste pri beli ponovni uporabi. Domenska razdalja med črno komponento in trenutnimi zahtevami naj bi bila sprejemljiva, če je možno komponento trenutnim zahtevam prilagoditi že z nastavitvijo ustreznih vrednosti parametrom, ki jih komponenta zagotavlja. Kontekstna razdalja mora biti pri črni ponovni uporabi blizu nič. Pri beli ponovni uporabi pa bo prilagoditvena razdalja sprejemljiva, le če bo strošek prilagajanja komponente trenutnim zahtevam manjši od stroška razvoja komponente na novo. Ker bela ponovna uporaba dovoljuje spreminjanje domene in konteksta, je sprejemljiva meja za kontekstno razdaljo pri tej obliki ponovne uporabe veliko višja kakor pri črni ponovni uporabi. Če je v interni komponentni knjižnici na voljo primerna črna komponenta, je seveda uporaba te komponente primernejša od katerekoli druge možnosti, saj so stroški oblikovanja črne komponente nižji od stroškov prilagajanja bele komponente ali razvoja komponente na novo (vozel 1 na sliki 2). Če pa take črne komponente v interni knjižnici ni, se morajo razvijalci odločati med pridobivanjem črne komponente s tržišča ali bele znotraj hiše. Slednje lahko pomeni le belo ponovno uporabo ali razvoj komponente na novo (vozel 2 na sliki 2). V obeh primerih je odločitev odvisna od cene komponente na tržišču (vozla 3 in 4 na sliki 2).

Odločitveni model prikazuje, da lahko komponentna tržišča s tem, ko omogočajo boljše sovpadanje potreb razvijalcev z razpoložljivimi komponentami, zmanjšajo potrebo po prilagajanju komponent. Pri tem se torej zmanjšata domenska in kontekstna razdalja, kar povzroči tudi znižanje cene komponente. Pričakovati je, da se bo z nadaljnjim razvojem komponentne tehnologije in mehanizmov komponentnih knjižnic ter razvojem ustreznih orodij cena iskanja komponent zelo znižala, torej naj bi bili stroški iskanja skoraj

zanemarljivi. Dostop do posrednikov, ki zbirajo komponente iz različnih virov, standardizirajo obrazce za katalogiziranje komponent, standardizirajo iskalne indekse in drugo potrebno, bo prav gotovo olajšal iskanje komponent in drastično zmanjšal stroške iskanja.

Torej, izbira ponovne uporabe črnih komponent, pridobljenih iz komponentnih tržišč, je boljša od bele ponovne uporabe vse dotlej, dokler so na voljo komponente s kratko prilagoditveno razdaljo in njihova cena razvijalcem omogoča ustvariti prihranke v primerjavi z belo ponovno uporabo.

2.3 TVEGANJA IN IZZIVI UDELEŽENCEV V KOMPONENTNIH TEHNOLOGIJAH

Eden od predpogojev za prevzem ali sprejem ponovne uporabe kot filozofije razvoja je tudi poznavanje izzivov in tveganj, ki jih na ponovni uporabi temelječe tehnologije ali komponentne tehnologije ter ponovna uporaba komponent prinašajo. V literaturi s področja komponentnih tehnologij so navedene številne pomembne razvojne prednosti in izzivi, ki jih te tehnologije prinašajo, težko pa v njej zasledimo opisana tveganja, povezana z uporabo teh tehnologij v vsakodnevnem razvojnem procesu. Nekatera najbistvenejša tveganja in vloge posameznih udeležencev ali uporabnikov komponentne tehnologije so predstavljena v nadaljevanju tega poglavja, hkrati pa so rabila kot osnovna izhodišča za izdelavo modela knjižnice ponovno uporabnih komponent, ki je podan v petem poglavju.

Med ključne uporabnike komponentnih tehnologij sodijo razvijalci komponent (inženirji domene), sestavljavci ali razvijalci aplikacij (inženirji aplikacij) in stranke – naročniki in/ali uporabniki aplikacij. Velikokrat se razvijalci aplikacij znajdejo istočasno v vlogi razvijalcev komponent in sestavljavcev aplikacij, kar je nenazadnje bistvo procesa ponovne uporabe. Med razvojem naj bi najprej poiskali in uporabili komponente, ki so že razvite (bodisi v interni knjižnici ali na komponentnih tržiščih), manjkajoče pa razvili na novo. Pri razvoju komponent na novo pa jih je smiselno razvijati z namenom njihove bodoče ponovne uporabe. Pri vse hitrejšem razvoju komponentnih tržišč in razmahu komponentnih tehnologij imajo vedno pomembnejšo vlogo posredniki oziroma dobavitelji in upravljavci komponent, ki so nek vezni

ali izmenjalni člen med razvijalci komponent in njihovimi neposrednimi uporabniki[25]. Njihova primarna naloga je zagotavljanje tržišča za izmenjavo ponovno uporabnih komponent.

2.3.1 Razvijalci komponent

S tveganji in izzivi komponentne tehnologije se razvijalci komponent ali inženirji domene srečujejo pri razvoju komponent, upravljanju in vodenju razvojnih projektov ter trženju komponent. Razvijalci bodisi proizvajajo komponente za potrebe tržišča (komponente za množično trženje[26]) ali za specifične potrebe strank – sestavljavcev aplikacij. V prvem primeru morajo odkriti tista poslovna področja ali domene, znotraj katerih in za katera bo razvijanje komponent smiselno in ekonomsko upravičeno. Za razvijalce komponent predstavlja pokrivanje neke domene s komponentami, ki domeni vsebinsko in količinsko ustrezajo, velik izziv. Dodatni problem poraja dejstvo, da se domene skozi čas spreminjajo, torej se morajo razvijalci med razvojem komponent za spreminjajočo domeno nenehno prilagajati tem spremembam, ki hkrati predstavljajo tudi nove potrebe in zahteve za komponente. Če razvoj komponent ni skrbno planiran, se hitro lahko zgodi, da se v razvijalčevi komponentni knjižnici znajde velika množica komponent, med katerimi je tudi precej zastarelih in neuporabnih. Če poskušajo razvijalci ta problem reševati z ovijanjem obstoječih komponent, zahteva to zelo veliko časa, komponente pa lahko hitro izgubijo prožnost, prilagodljivost in kakovost, s tem pa se tudi stopnja ponovne uporabe takih komponent drastično zniža.

Pri obvladovanju domene morajo razvijalci znati domeno pravilno oziroma čim optimalneje razdrobiti na povezano množico komponent, kajti te komponente nosijo v sebi znanje o domeni. Njihovo število, zrnatost, velikost in medsebojne odvisnosti pa igrajo odločilno vlogo pri razvoju aplikacij domene iz komponent. Zelo pomembno je tudi, da se razvijalci prepričajo, da posamezne komponente dejansko odslikujejo znanje o domeni in poslovne procese te domene. Veliko prednost in izziv za razvijalca predstavlja tudi razvoj ustreznega vmesnika komponente (določanje parametrov), ki je ključen za ponovno uporabo same

[25] To so predvsem sestavljavci aplikacij, posredno pa tudi uporabniki aplikacij.
[26] Ang. *mass-market components.*

komponente. V vmesniku mora biti določeno, kako in kaj komponenta dela, kateri so vhodni parametri, kakšni so pričakovani rezultati delovanja, kakšni so postopki rokovanja z izjemami in podobno. Razvoj je otežen, ker si s konvencionalnimi pristopi razvijalci ne morejo pomagati, novih orodij za gradnjo komponent ni veliko, predvsem pa so začetni stroški nabave takih orodij zelo visoki, potrebnega pa je tudi veliko znanja. Še en pomemben faktor je izbira ustrezne vmesne prevajalne opreme (ang. *middleware*[27]), ki mora biti usklajena z zahtevami, bodisi posamezne stranke bodisi množičnega tržišča.

Pri vodenju projektov komponentnega razvoja mora razvijalec spremljati in opazovati vsako komponento od nastanka do njene prve ponovne uporabe. Zaradi popravljanja napak in izboljševanja komponent je lahko nadzor nad različicami iste komponente problematičen (težaven), zato je vzpostavitev mehanizmov spremljanja različic zelo pomembna, saj morajo biti komponente, uporabljene v več aplikacijah, med seboj usklajene. Razvijalci morajo o izdaji nove različice komponente obvestiti sestavljavce sistemov. Kako pogosto bodo izdajali nove različice, kako bodo o tem obveščali svoje stranke in podobno, določijo razvijalci sami.
Razvoj kakovostnih komponent zahteva izčrpno preskušanje. Nad vsako komponento je treba v času njenega razvoja izvesti validacijo in verifikacijo. Glede na večstransko uporabo komponent v različnih aplikacijah, je postopek njihovega preskušanja zahtevnejši kakor pri klasičnih aplikacijah, saj mora razvijalec vsako komponento preskušati kot enoto, za katero pa ne ve natanko, v kakšnem kontekstu bo uporabljena in kdo jo bo uporabil. Komponentni razvoj programske opreme zahteva tudi nov nabor specifičnih metrik, saj klasična merila kritičnih faktorjev kot sta npr. zanesljivost in produktivnost, kot meri nista najbolj primerni za komponente [45].

Pred zagonom projekta komponentnega razvoja je treba opraviti analizo stroškov in prihodkov, ki nam pomaga pri odločitvi, ali neko naročilo stranke oziroma sestavljavca aplikacij sprejeti ali se raje usmeriti v razvoj komponent za množično prodajo. V strošku gradnje komponente so upoštevani stroški analize domene, identifikacije, razvoja in preskušanja komponente. Ključni prihodek s komponento predstavlja njena prodaja. Pri oblikovanju cene komponente za specifično stranko je odločilna ocena razvojnih stroškov te

[27] Vmesna prevajalna oprema je programska oprema za upravljanje komunikacije ali posredovanje med aplikacijo in omrežjem. Sodi med komunikacijske elemente, ki omogočajo delovanje koncepta porazdeljenosti.

komponente, pri oblikovanju cene komponente za množično trženje pa je treba upoštevati tudi bodoče povpraševanje tržišča, konkurenco ipd., kar pa je v začetnih fazah razvoja komponente zelo težko predvideti. Ker kakovost komponente neposredno vpliva tudi na kakovost iz komponent sestavljene aplikacije, je odgovornost razvijalca komponente toliko večja. Ko pa se število na trgu ponujenih komponent različnih razvijalcev veča, je za vzpostavitev zaupanja uporabnikov teh komponent poleg certificiranja komponent zelo pomembno tudi certificiranje razvijalcev[28]. Za potencialnega kupca komponente (sestavljavca aplikacij) je zelo pomembno, da ga prepričamo tudi v avtentičnost komponente, saj ga tako zaščitimo pred uporabo oziroma nakupom ponarejenih komponent ter s tem pred vpletenostjo v kakršnekoli kršitve pravic intelektualne lastnine pri morebitni uporabi ponaredka. Vzpostaviti je torej treba ustrezne nadzorne mehanizme, ki bodo ponarejevalcem preprečevali povratno inženirstvo in reinženirstvo nad komponentami. Dodatni varnostni prijemi so potrebni tudi za to, da se komponente zaščitijo pred korupcijo, virusi, vohunskimi programi in hekerji.

Razvijalci navadno posredujejo komponente za množično trženje na trg prek izbranih posrednikov (distributerji programske opreme). Najpogosteje so to spletni ("on-line") prodajalci ali posredniki. Med njimi se razvijalci odločajo na temelju učinkovitosti orodij, ki jih ponujajo njihove spletne knjižnice (orodja za klasificiranje, brskanje, iskanje in izbor ustrezne komponente iz knjižnice), opremljenosti knjižnic z mehanizmi za spremljanje različic, preprečevanje kršitve pravic intelektualne lastnine, varnostnih mehanizmov idr. Ne glede na to, da razvijalec zaupa trženje svojih komponent posredniku, se mora soočiti še z enim zelo pomembnim tržnim problemom, to je izbira takega načina plačila komponente[29], da bo zanj profit optimalen oziroma čim večji. Za sodelovanje med razvijalci in tistimi, ki tržijo njihove izdelke, je ustrezna licenčna pogodba bistvenega pomena. Iz nje je, med drugim, razvidna tudi najprimernejša uporaba komponente in razvijalčeva odgovornost v primeru, ko komponenta ne bo delovala tako, kot je navedeno v njeni specifikaciji. V primeru specifične stranke se razvijalec in sestavljavec aplikacij dogovorita o lastništvu komponent in iz njih izpeljanih izdelkov. V primeru množičnega trženja pa se razvijalci soočijo z nekaterimi

[28] Današnji prodajalci komponent (večinoma poteka prodaja prek interneta) že ponujajo možnost certificiranja komponent, poleg pa še kopico dokumentacije, katere namen je prepričati potencialno stranko v to, da so kot razvijalci vredni zaupanja.
[29] Z razvojem komponent so se razvili različni načini plačevanja komponent oziroma njihove uporabe: takojšnji nakup vsega za takojšnje plačilo, plačilo po uporabi, časovno omejena uporaba, omejitev števila uporab ipd.

enakimi problemi intelektualne lastnine kakor razvijalci klasičnih aplikacij. Ravno problemi pravne narave lahko v veliki meri vplivajo ali so celo odločilni za nakup oziroma uporabo neke komponente.

2.3.2 Razvijalci ali sestavljavci aplikacij

Glavne aktivnosti, s katerimi se srečujejo razvijalci aplikacij, so sestavljanje komponent v aplikacije ter upravljanje projektov komponentne gradnje aplikacij. Pri svojem delu so v veliki meri odvisni od delovanja in ponudbe komponentnih tržišč. Večina aktivnosti razvijalcev aplikacij je povezanih z iskanjem in izbiranjem komponent. Ta proces je iterativen. Uspešnost iskanja potrebnih komponent je odvisna od klasifikacijskih in iskalnih mehanizmov, ki jih ponujajo posamezni spletni prodajalci komponent. Tako kakor razvijalci komponent se tudi razvijalci aplikacij pri svojem delu srečujejo z zelo velikim številom različnih komponentnih knjižnic, v katerih morajo razpoznati in locirati komponente, ki jih potrebujejo. Vedno obstaja nevarnost, da izbrana komponenta ne deluje v skladu s svojo specifikacijo ali da njeno sodelovanje z drugimi komponentami ni uspešno, zato mora uporabnik komponente dodatno usklajevati. Zgodi se tudi lahko, da razvijalci aplikacij na prostem trgu ne uspejo najti komponent, ki jih potrebujejo. V tem primeru lahko bodisi poiščejo razvijalca, ki jim bo razvil komponento glede na specifične potrebe, bodisi poskušajo prepričati svojo stranko (naročnika aplikacije), da svoje zahteve prilagodi trenutno razpoložljivim komponentam. Razvijalec aplikacije mora tako skupaj s svojo stranko (naročnikom aplikacije) izvesti stroškovno analizo obeh primerov, saj bi naročilo specifičnih komponent povečalo stroške in s tem ceno aplikacije, sprememba zahtev in uporaba na trgu razpoložljivih nadomestnih komponent pa bi lahko to ceno tudi zmanjšali.

Preden razvijalci oddajo komponente v knjižnico, jih morajo preskusiti[30] ter zagotoviti, da komponente ne vsebujejo napak. Resnejše in kakovostnejše komponentne knjižnice morajo same poskrbeti, da preverijo kakovost in ustreznost posameznih komponent, preden jih sprejmejo v knjižnico. Razvijalec aplikacij se o izvedenih preskusih in ocenah lahko prepriča v spremljajoči dokumentaciji h komponenti. Če pa dokumentaciji ali komponentni knjižnici ne zaupa, mora vsako komponento iz knjižnice pred uporabo preskusiti sam in s komponento

[30] Preskušanje enote.

predložiti tudi ustrezna dokazila oziroma rezultate preskušanj. V vsakem primeru mora procesu sestavljanja komponent v celoto slediti še preskus integracije ali preskus celote. Ta mora biti še posebno skrbno izveden, saj so viri komponent različni, delovati pa morajo uglašeno v skladu z uporabnikovimi zahtevami. Preskušanje integracije je pomembno tudi zato, ker je možno, da razvijalci aplikacij neko komponento uporabijo v drugačne namene od tistih, za katere je bila razvita.

Na videz sta si življenjski razvojni cikel klasičnih aplikacij[31] in življenjski razvojni cikel na temelju komponent zgrajenih aplikacij zelo podobna, vendar gre pri slednjem v resnici za izrazito različna opravila in postopke, ki jih je treba izvesti dodatno. Specifikacije komponent nastanejo npr. v času načrtovanja aplikacije, medtem ko predstavljata bistveni del podrobnega načrtovanja aplikacije iskanje in identifikacija komponent. Razvijalci morajo slediti nadaljnjemu razvoju, nadgradnjam in različicam ter morebiti odstranitvam posameznih komponent, uporabljenih v njihovih aplikacijah, iz knjižnic. Glede na to, da že sami razvijalci aplikacij pogosto izdelajo in tržijo kar nekaj različic iste aplikacije, je sledenje različicam ponovno uporabljenih komponent še toliko bolj pomembno.

Komponentni razvoj aplikacij zahteva tudi drugačne vloge razvojnega osebja, z drugačnimi sposobnostmi in spretnostmi od tistih, ki se jih zahteva pri klasičnem razvoju. Analitiki morajo npr. biti še posebno (izurjeni v analizi) vešči v analizi sovpadanja uporabniških zahtev z razpoložljivimi komponentami iz knjižnice, obvladovanju različnih iskalnih tehnik in podobno. Razvijalci aplikacij potrebujejo za oceno učinkovitosti procesa sestavljanja komponent in njegovih rezultatov posebne metrike, potrebujejo pa tudi posebna orodja, ki olajšajo sestavljanje komponent (npr. vizualna orodja, ki omogočajo sestavljanje komponent kar na namizju razvijalčevega računalnika). Imeti morajo tudi dober vpogled v sam proces razvoja aplikacije iz komponent.

Iz komponent sestavljena aplikacija bo podedovala vse morebitne slabosti posameznih komponent, iz katerih je zgrajena. Uporabniki bi lahko zato zahtevali, da jim razvijalci aplikacije posredujejo podrobnosti o komponentah, ki bodo njihovo aplikacijo sestavljale, saj komponentam ne zaupajo. Zato je pomembno, da bi se področje certificiranja komponent

[31] Mišljen je razvoj aplikacij, ki ne temelji na ponovni uporabi (ne glede na izbrano metodologijo razvoja).

standardiziralo, sicer se bodo razvijalci aplikacij vedno bolj omejevali le na manjšo množico skrbno izbranih razvijalcev komponent, katerim bodo zaupali, ne bodo pa posegali po drugih komponentah, ker jim dodatno certificiranje teh komponent zaradi nezaupanja vanje poveča celotne stroške ponovne uporabe in razvoja. To pa je v nasprotju s strategijo komponentnih tržišč in zmanjšuje prednosti trgovanja s komponentami na odprtem trgu, kar predstavlja bistvo komponentnih tehnologij. Zanašanje razvijalcev aplikacij samo na zunanje vire komponent postavlja razvijalce aplikacij pred pomembno tveganje, saj imajo v tem primeru zelo omejen nadzor nad komponentami; nad tipi razpoložljivih komponent, nad časovnim razporedom izdelave nadaljnjih različic komponent, nad zagotovili, da bodo nove različice združljive s starimi. Razpoložljivost razvijalca aplikacije za njenega uporabnika pomeni, da je razvijalec aplikacij pripravljen sprotno odkrivati in odpravljati napake ter po potrebi aplikacijo izboljševati. Če pa je razvijalec aplikacije odvisen od razvijalca komponent, ki jih množično trži in s katerim ni nujno, da bo dolgo sodeloval, predstavlja to zanj precejšnja tveganja. Zato je še toliko bolj pomembno, da se vsi ti odnosi in odvisnosti ustrezno opredelijo v licenčnih pogodbah oziroma pogodbah o medsebojnem sodelovanju.

2.3.3 Uporabniki aplikacij

Uporabniki aplikacij so stranke, ki aplikacije naročajo zaradi specifičnih potreb in zahtev poslovnega okolja, v katerem delujejo. Ob tem seveda pričakujejo, da bo aplikacija tem potrebam in zahtevam zadostila. Zaradi pomanjkanja ustreznih oziroma najprimernejših komponent se kaj hitro lahko zgodi, da razvijalci izdelajo aplikacijo iz tistih komponent, ki so na razpolago, kar lahko pomeni, da aplikacija ne ustreza postavljenim zahtevam stranke. Lahko se tudi zgodi, da izbrane komponente niso dovolj kakovostne ali da morda ne delujejo po specifikacijah, na podlagi katerih so bile izbrane. Če bi taka aplikacija ovirala stranko pri izvajanju njenih ključnih poslovnih aktivnosti, bi to za stranko predstavljalo veliko tveganje.

Tudi upravljanje množice aplikacij, zgrajenih iz komponent, predstavlja za uporabnika določeno tveganje. Uporabnik za izvajanje svojih poslovnih operacij oziroma aktivnosti navadno že uporablja neko množico aplikacij, pričakuje pa in zahteva, da bo nova aplikacija, v tem primeru sestavljena iz komponent, sposobna sodelovati z obstoječimi. Če naročena aplikacija ne deluje tako, kakor je bilo v zahtevah postavljeno, lahko za pomanjkljivo ali

nepravilno delovanje razvijalci komponent in razvijalci aplikacij krivijo drug drugega, uporabnik pa ne more izkoriščati aplikacije, katere razvoj je plačal. Tovrstne probleme oziroma njihove rešitve je zato smiselno vključiti v pogodbe o medsebojnem sodelovanju.

Težavne odločitve, ki jih morajo uporabniki aplikacij sprejemati, so povezane tudi z izborom razvijalca nove aplikacije. Potencialne kandidate je treba najprej identificirati, jih oceniti (ovrednotiti), nazadnje pa izbrati najprimernejšega. Stranke bodo navdušene, če bodo lahko pri izboru razvijalcev komponent ter samih komponent vplivale na razvijalca aplikacije. Na ta način lahko razvijalce komponent tudi prepričajo v to, da v komponento vgradijo njihovo poslovno logiko in prakso. Vendar za take komponente uporabnik, ki jih je naročil, zaradi ščitenja poslovnih skrivnosti, ne bo dovolil ponovne uporabe. Vsekakor je koristno, da se v pogodbi med razvijalcem aplikacije in njenim naročnikom ta prepoved eksplicitno navede. Pri prehodu iz tradicionalnega razvoja na komponentni razvoj je zelo pomembno, da stranke natančno ocenijo oziroma razumejo, za katere projekte je komponentni razvoj primernejši. Gotovo je tveganje manjše, če se stranke odločijo, da bodo komponentno razvile najprej rutinske in nestrateške aplikacije, šele pozneje pa prešle na komponentni razvoj strateških aplikacij. Velika bojazen strank se lahko poraja tudi ob razmišljanju, da s tem, ko bodo njihove aplikacije zgrajene iz komponent, ki jih je mogoče kupiti na prostem trgu, ne morejo dosegati kompetitivne prednosti pred konkurenco, saj ima tudi ta dostop do istih komponent. Resda je kakovost aplikacije v veliki meri odvisna od komponent, ki jo sestavljajo, vendar je bistvo doseganja kompetitivne prednosti odvisno predvsem od sposobnosti kombiniranja na trgu razpoložljivih komponent s komponentami, ki so namensko razvite na zahtevo stranke.

2.3.4 Medsebojne odvisnosti med udeleženci

Ker so razvijalci aplikacij v veliki meri in vedno bolj odvisni od razvijalcev komponent, stranke oziroma uporabniki aplikacij pa od aplikacij, ki so jih razvili razvijalci aplikacij, se tveganja in izzivi prenašajo iz enega udeleženca komponentne tehnologije na drugega. V tabeli 2 so zbrana ključna tveganja in izzivi posameznih udeležencev, prikazan pa je tudi kaskadni učinek med posameznimi udeleženci, npr:

- modeliranje domene je osnova, na kateri se gradijo ponovno uporabne komponente, ki naj bi zadovoljevale zahteve uporabnikov aplikacij znotraj te domene;

- izdelava novih različic komponent vpliva tudi na opravila, ki jih izvajajo razvijalci aplikacij, saj morajo stalno spremljati nove različice komponent, ki so jih ponovno uporabili, ter ustrezno spreminjati aplikacije, ki te komponente vsebujejo;
- preskušanje komponent olajša preskušanje celotne aplikacije in vpliva tudi na večjo kakovost aplikacije ter s tem večje zadovoljstvo uporabnika;
- kakovost komponent, ki se preverja s certificiranjem in preverjanjem verodostojnosti, v veliki meri vpliva na kakovost aplikacije;
- veliko število knjižničnih komponent lahko poraja problem neskladnosti med knjižnicami.

Zato je toliko bolj pomembno, da se ta tveganja zmanjšajo z ustrezno postavljenimi pravili knjižnice in/ali komponentnih tržišč ter zasnovo procesov, povezanih s ponovno uporabo, ravno tako pa tudi z ustreznimi pogodbami, ki uravnavajo odnose med udeleženci v procesu ponovne uporabe. Slednje so natančneje opisane v šestem poglavju.

Razvijalec komponent	Sestavljavec aplikacij	Uporabnik aplikacij
Razvoj komponent	**Sestavljanje aplikacije**	**Uporaba aplikacije**
Modeliranje domene →	Zadovoljitev zahtev →	Zadovoljene zahteve
Lastnosti komponent (npr. velikost) →	Neskladne knjižnice →	Kakovost
Vodenje projekta	**Vodenje projekta**	**Upravljanje aplikacije**
Različice komponent →	Različice aplikacij	Omejen nadzor
Preskušanje komponent →	Preskušanje aplikacij	Novi odnosi
Orodja in metodologije →	Kakovost/certificiranje/avtentičnost	Sovpadanje z obstoječimi sistemi
Nove metrike	Nove metrike	Identifikacija primernih sestavljavcev
Novo osebje	Novo osebje	Identifikacija projektov
Trženje	**Trženje**	**Strateške prednosti**
Množina komponentnih knjižnic →	Odnosi med prodajalci →	Doseganje kompetitivne prednosti
Kakovost/certificiranje/avtentičnost →	Lastništvo/licenciranje →	Lastništvo/licenciranje
Specifična stranka ali množično trženje		
Lastništvo/licenciranje		

Legenda: tveganje izziv tveganje in izziv

Tabela 2: Medsebojni vplivi med udeleženci v komponentnih tehnologijah.

2.3.5 Posredniki ali upravitelji komponent

Osnovna naloga posrednikov komponent je zagotavljanje tržišča za izmenjavo ponovno uporabnih komponent. Na eni strani prevzemajo komponente od razvijalcev ali dobaviteljev komponent, jih vključujejo na tržišče, komponentno knjižnico ali mrežo komponentnih knjižnic ter na drugi strani prodajajo sestavljavcem aplikacij. Slednji pa lahko posrednikom podajajo svoje potrebe, ki jih ti posredujejo naprej razvijalcem komponent.

Če v verigo udeležencev vključimo posrednike komponent, potem gredo vse transakcije med razvijalci komponent in njihovimi uporabniki prek njih. Posredniki so odgovorni tudi za zagotavljanje kakovosti in varnosti komponent, torej se lahko certificiranje izvaja izključno v njihovi domeni in se s tem v veliki meri razbremenijo razvijalci komponent in sestavljavci aplikacij. Osnovni predpogoj za sodelovanje pa je seveda zaupanje, ki ga posredniki komponent lahko pridobijo ravno s posredovanjem kakovostnih in "čistih" komponent, kakovostnih storitev, z jasnimi pravili delovanja in preverjanja komponent. Natančneje sem njihovo vlogo opisala v petem poglavju.

Posredniška vloga se nanaša na premostitev vrzeli med tem, kar tržišče (t.j. sestavljavci in uporabniki aplikacij) želi in potrebuje ter tem, kar dobavitelji (t.j. razvijalci komponent) proizvajajo.

Slabost obstoječih komponentnih tržišč vidim v tem, da izvajajo samo distribucijo (fizični pretok) komponent, niso pa učinkoviti posredniki. Zato menimo, da je vloga posrednikov še toliko bolj pomembna.

3 PRAVNI VIDIKI PROGRAMSKE OPREME IN NJENE PONOVNE UPORABE

3.1 PRAVNO VARSTVO PROGRAMSKE OPREME

Za socialni in ekonomski razvoj vsake države je ena najpomembnejših industrija programske opreme ali, nekoliko širše, industrija računalniških ali informacijsko komunikacijskih tehnologij. Ta industrija je nenehen vir inovacij in se zelo naglo razvija. Njenim predstavnikom je od vedno v interesu zavarovati svoje izdelke pred zlorabami, krajami, uničenji, neupravičenimi ekonomskimi izkoriščanji in podobnim – še posebej danes, v obdobju globalne ekonomije in velikih zaslužkov s programsko opremo.

Ker sodijo računalniški programi in programska oprema nasploh med intelektualne stvaritve, so varovani s pravno ureditvijo intelektualne lastnine. V primerjavi z drugimi oblikami intelektualne lastnine ima programska oprema kar nekaj posebnosti: predstavlja enega glavnih dejavnikov tehnološkega razvoja; je izjemno draga in zahtevna naložba; navadno gre pri programski opremi za kompleksne stvaritve, ki vključujejo kar skupine strokovnjakov; njena najpomembnejša prednost in hkrati njena slabost pa je v tem, da jo je mogoče na izjemno preprost in poceni način razmnoževati in razpečevati. Zato so pravice intelektualne lastnine na programski opremi med najbolj kršenimi in tudi najbolj ogroženimi.

Poglavitna naloga ali cilj pravnega varstva računalniških programov je preprečevanje ter omejevanje piratstva oziroma varovanje pravic njihovih avtorjev in/ali lastnikov. Piratstvo se nanaša na način reprodukcije in distribucije računalniških programov, pri čemer so kršene avtorske in druge pravice, saj tretjim osebam omogoča uporabo nelegalnih kopij programa.

Pravice intelektualne lastnine na računalniških programih se varujejo na različne načine[32]: z avtorskim pravom, s patentnim pravom, pravom poslovne skrivnosti, konkurenčnim pravom, pravom blagovne znamke in pogodbenim pravom, žal pa ne obstaja svojevrstna pravna ureditev, ki bi v popolnosti ali vsaj v večjem delu urejala pravno varstvo programske opreme v celoti, upoštevajoč posebnosti novih informacijsko komunikacijskih tehnologij, posebnosti novih pristopov k razvoju programske opreme in vseh posebnosti, ki jih prinaša internet. Katera od oblik varstva je za dani računalniški program najprimernejša, je odvisno od programa in njegovega namena. Na veliko odločitev, povezanih z razvojem, distribucijo, vzdrževanjem in izboljševanjem programske opreme, vplivajo omejitve, ki izhajajo ravno iz prava intelektualne lastnine, zato je za razvijalce zelo koristno, če te omejitve poznajo.

Tako kot pri programiranju tudi v pravu za dani problem ne obstaja samo ena – najboljša rešitev. Zato se v praksi za čim popolnejšo varstvo programske opreme (intelektualnih pravic na programski opremi) navadno uporablja kombinacija pravnih ureditev. Te ureditve se dopolnjujejo in velja t.i. načelo kumulativnega varstva. Pri kombiniranju oblik varovanja pa je potrebna previdnost, saj se nekatere oblike izključujejo in kaj hitro lahko pride do konfliktov.

Pravne ureditve s področja intelektualne lastnine so se skozi zgodovino stalno spreminjale in prilagajale, da bi zadostile potrebam in zahtevam, ki so jih porajali številni novi tehnološki dosežki na področju informacijske in digitalne elektronske tehnologije. Ko je bilo reševanje problemov s področja programske opreme ali računalniških tehnologij z uporabo klasičnih, že uveljavljenih načel pravnega varstva, ali z njihovim prilagajanjem oteženo oziroma onemogočeno, so se oblikovali novi, *sui generis* ali svojevrstni pravni pristopi ali koncepti[33]. Ti so bili oblikovani tako, da so obstoječe ureditve samo dopolnili, ne pa nadomestili ali zamenjali. Navadno so pravni koncepti *sui generis* zlitje obstoječih pravnih konceptov s specifičnimi pravili, ki naslavljajo določen problem ali ožje področje [78].

Slovenija je že pred vstopom v EU zakonodajo s področja intelektualne lastnine prilagodila evropskim zahtevam. Področje varstva računalniških programov ureja v okviru avtorskega

[32] Glej priloga A.
[33] Za varstvo topologije integriranih vezij (Zakon o varstvu topografije polprevodniških vezij) [140] in za varstvo podatkovnih baz (Direktiva o varstvu podatkovnih baz) [114] takšni svojevrstni pravni ureditvi znotraj EU že obstajata.

prava (ZASP[34]), kakor večina preostalih držav, zato je tej prevladujoči obliki varovanja pravic intelektualne lastnine na računalniških programih namenjen pretežni del priloge A.

Glede na izjemen in ekonomsko vedno večji gospodarski pomen računalniških programov ter glede na probleme, ki jih porajajo nove oblike shranjevanja in distribucije programske opreme, bo tako v evropski in slovenski kakor v svetovni pravni praksi treba ponovno[35] resno razmisliti o oblikovanju mednarodne pravne ureditve za področje programske opreme in informacijsko komunikacijskih tehnologij.

3.1.1 Pravo intelektualne lastnine

Pojem intelektualne lastnine se nanaša na vrsto lastnine, ki izvira iz človekovega intelekta oziroma razuma [86]. Ko je ta lastnina opredmetena v blagu in/ali storitvah, jo lahko imetnik pravice ali oseba, ki jo je ta pooblastil, komercialno izkoriščata.

Pravo intelektualne lastnine je skup pravnih pravil, ki urejajo pravice na intelektualnih stvaritvah. Je novejša samostojna pravna disciplina, ki vključuje pravo industrijske lastnine in avtorsko pravo ter hibridne pravne sisteme *sui generis*, ki se kot posledica tehnološkega razvoja pojavljajo vse pogosteje. Pravo industrijske lastnine ureja izume, vzorce in industrijske modele, blagovne in storitvene znamke ter označbe porekla blaga, avtorsko pravo pa ureja avtorske in sorodne pravice na književnih, umetniških in znanstvenih delih. S pravnima sistemoma *sui generis* s področja računalniških tehnologij pa se uravnavata področje topografije polprevodniških računalniških vezij in podatkovnih baz[36].

Ker je programska oprema kot produkt človekove intelektualne ustvarjalnosti tudi intelektualna stvaritev, sodi med predmete varstva in pravic v okviru intelektualne lastnine.

[34] ZASP – Zakon o avtorski in sorodnih pravicah, Uradni list RS, št. 21-958/1995 in 43-1927/2004 [133].
[35] Tendence po svojevrstni pravni ureditvi varovanja računalniških programov so že dolgo navzoče. Celo WIPO – World Intellectual Property Organization (slovensko SOIL – Svetovna organizacija za intelektualno lastnino) je leta 1970 predlagala tako svojevrstno ureditev, pozneje pa od tega odstopila in kot osnovo za Direktivo o pravnem varstvu računalniških programov (ang. *Council Directive on the legal protection of computer programs*), sprejeto 14.5.1991, vzela model avtorskih pravic. Obširno lahko o prednostih in slabostih take ureditve beremo tudi v [78, 79].
[36] Glej opombo 24.

Pravo intelektualne lastnine torej predstavlja ogrodje ali okvir, znotraj katerega se ustvarjajo, dodeljujejo in uveljavljajo pravice na programski opremi (računalniških programih).

Intelektualna lastnina na programski opremi oziroma računalniških programih je urejena v številnih pravnih aktih, mednarodnih in nacionalnih.

Med najpomembnejše mednarodne konvencije sodijo: Pariška konvencija za varstvo industrijske lastnine (sprejeta 1883)[37] in Bernska konvencija za varstvo književnih in umetniških del (sprejeta 1886)[38], ki predstavljata temelj kasneje sprejetim konvencijam in sporazumom, Madridski aranžma o mednarodnem registriranju znamk (sprejet 1995)[39], Sporazum o trgovinskih vidikih pravic intelektualne lastnine – Sporazum TRIPS (sprejet 1994)[40], Svetovna (univerzalna) konvencija o avtorski pravici ali Pogodba svetovne organizacije za intelektualno lastnino o avtorski pravici (sprejeta 1996)[41] ter druge[42].

Med najpomembnejše direktive EU sodijo: Direktiva o pravnem varstvu računalniških programov (sprejeta 1991)[43], Pogodba o sodelovanju na področju patentov (sprejeta 1992)[44], Direktiva o varstvu avtorskih in sorodnih pravic (sprejeta 1993)[45], Direktiva o pravnem varstvu baz podatkov (sprejeta 1996)[46], Direktiva o informacijski družbi in harmonizaciji določenih vidikov avtorskih in sorodnih pravic v informacijski družbi (sprejeta 2001)[47], Direktiva o uveljavitvi pravic intelektualne lastnine (sprejeta 2004)[48].

Najpomembnejši nacionalni zakoni, predpisi in uredbe s področja intelektualne lastnine: Zakon o avtorski in sorodnih pravicah (1995), Zakon o industrijski lastnini (1992), Zakon o pravicah industrijske lastnine iz delovnega razmerja (1995), posredno pa še z Zakonom o

[37] Ang. *Paris Convention for the Protection of Industrial Property.*
[38] Ang. *Berne Convention for the Protection of Literary and Artistic Works.*
[39] Ang. *Madrid Agreement Concerning the International Registration of Marks.*
[40] Ang. *Agreement on Trade-Related Aspects of Intellectual Property Rights.*
[41] Ang. *WIPO Copyright Treaty.*
[42] Ang. Npr. Pogodba o pravu blagovne znamke (sprejeta 1994), Pogodba o patentnem pravu (sprejeta 2000).
[43] Ang. *Council Directive on the legal protection of computer programs.*
[44] Ang. *European Patent Convention.*
[45] Ang. *Council Directive of protection of copyright and certain related rights.*
[46] Ang. *Council Directive on the legal protection of databases.*
[47] Ang. *Council Directive on the harmonisation of certain aspects of copyright and related rights in the information society.*
[48] Ang. *Council Directive on the enforcement of intellectual property rights.*

varstvu topografije polprevodniških vezij (1995), Zakonom o varstvu konkurence (1993), Zakonom o gospodarskih družbah (1993), Obligacijskim zakonikom (2001) idr. z vsemi spremembami.

Osnovni namen prava intelektualne lastnine je vzpodbujanje razvoja inovacij, ki lahko prispevajo k napredku znanja in znanosti ter širjenju inovativnih del za splošno uporabo[49]. To pravo daje ustvarjalcem izključne pravice, s katerimi lahko nadzorujejo izkoriščanje varovanega dela v komercialne namene.[50] Pravice trajajo določeno časovno obdobje (odvisno od vrste zaščite), po preteku tega časa pa izdelek preide v javno domeno ali javno last (ang. *public domain*) in takrat ga je možno prosto uporabljati ali kopirati, torej postane javna dobrina[51].

Pravice intelektualne lastnine ne smejo biti premočne, ker preprečujejo prenos znanja, napredka itn., ne smejo pa biti prešibke, ker sicer dela oziroma stvaritve niso ustrezno varovane, kar ima za posledico padec stimulacije za ustvarjanje novega znanja in novega napredka.

3.1.2 Poenotenje pojmov programska oprema in računalniški program

V vseh doslej omenjenih pravnih virih lahko zasledimo nekonsistentno uporabo terminov računalniški program in programska oprema oziroma njuno neupravičeno poenotenje. Problem namreč predstavlja dejstvo, da je programska oprema veliko širši pojem od računalniškega programa, vse obstoječe oblike pravnega varstva pa se po vsebini nanašajo na računalniške programe.

Izraza računalniški program (ang. *computer program*) in programska oprema (ang. *software*, *computer software*) se v pogovornem jeziku pogosto enačita, vendar vsebinsko to enačenje ni

[49] S pojmom intelektualne lastnine se opišejo pravice, ki se nanašajo na intelektualno aktivnost na industrijsko-gospodarskem področju, znanosti, književnosti in umetnosti – 2. člen Konvencije o ustanovitvi svetovne organizacije za intelektualno lastnino (WIPO), Stockholm, 14.7.1967. WIPO je za svoj osnovni cilj postavila pospeševanje varstva intelektualne lastnine v svetu [74], kar se odraža v vseh konvencijah, ki jih sprejema.
[50] Podeljene pravice so v bistvu negativne: to so pravice, ki drugim preprečujejo, da naredijo določene stvari oziroma da izkoriščajo neko delo ali v njem izražene ideje brez dovoljenja imetnika pravic. [86]
[51] Javne dobrine so dela, ki jih ni mogoče zaščititi znotraj sistema intelektualne lastnine (ali jim je zaščita že potekla), če pa nad tem delom izvedemo izboljšave, lahko zaščitimo samo slednje.

upravičeno. Programska oprema poleg računalniških programov[52] obsega tudi spremljajočo dokumentacijo (programsko ali razvojno[53] in uporabniško[54]), pripravljalno gradivo za program ter podatkovne baze, med programsko opremo pa sodijo tudi vsi "vmesni" izdelki, ki nastajajo v procesu razvoja nekega programskega sistema ali aplikacije[55] [93, 108]. Po Trampužu et al. [104] je pojem programske opreme zbirni pojem, katerega sestavni deli so računalniški program, programska in uporabniška dokumentacija, in obsega vse, kar je potrebno za uporabo strojne opreme.

Programsko opremo delimo po več merilih, najpogosteje glede na namembnost: če gre za programsko opremo, ki določa način delovanja računalniškega sistema in to delovanje krmili, potem gre za *sistemsko programsko opremo*, če je njena naloga reševanje specifičnih problemov konkretnega uporabnika, pa gre za *aplikacijsko ali uporabniško programsko opremo*.

S stališča licenčnih pogodb se programska oprema deli na *standardno* (paketno), namenjeno široki porabi, dosegljivo v računalniških prodajalnah ali na spletnih straneh proizvajalcev (ang. *off-the-shelf software*), narejeno za vnaprej nedoločeno število uporabnikov, in na *individualno*, narejeno posebej po željah in potrebah določenega uporabnika, navadno na podlagi pogodbe o razvoju ali naročilu programske opreme (ang. *custom developed software*). Programsko opremo oziroma računalniške programe lahko legalno pridobimo na tri načine: z razvojem v lastni režiji, po naročilu ali z nakupom paketne programske opreme[56] oziroma licence za njeno uporabo. Pravice, povezane z uporabo pridobljene programske opreme, pa tudi pravni problemi, so v veliki meri odvisne od načina pridobitve ter vrste programske opreme.

[52] Računalniški program je niz ali skupek navodil (ukazov), ki jih računalnik razume in na njihovi podlagi izvaja operacije, ki so potrebne za rešitev neke naloge pri samodejni obdelavi podatkov.
[53] Gradivo, ki predstavlja osnovo, na kateri je izdelan program, navadno v obliki seznama nalog, opisa programa, diagramov ipd.
[54] Vsebuje navodila in napotke za uporabo programa. Lahko je v tiskani ali v elektronski obliki.
[55] To so lahko specifikacije zahtev, znanje o domeni, procesi, metode, preskusni primeri in preskusni plani, modeli, vzorci, ogrodja, izvedbe razredov in primerki objektov (pri objektno usmerjenem razvoju) ipd.
[56] Po [88] je v ZDA več kot tretjina prihodkov industrije programske opreme iz naslova programske opreme po naročilu, ne paketne.

Različne obstoječe oblike varstva intelektualne lastnine na računalniških programih, to so avtorskopravno in patentnopravno varovanje, poslovna skrivnost, blagovna in storitvena znamka, varovanje pred nelojalno konkurenco ter pogodbeno pravo so podrobno opisani v prilogi A. Še posebno podrobno sta opisani avtorskopravno varovanje, ker je ta oblika najpogostejša, in pogodbeno pravo oziroma licenčne pogodbe, ki predstavljajo dodatek k posameznim oblikam varstva.

3.1.3 Sui generis (svojevrstno) varstvo računalniške tehnologij

Sui generis oblika varstva za programsko opremo se vzpostavi s kombiniranjem pravnih načel, veljavnih za različna področja intelektualne lastnine. Razlogi za takšno kombiniranje so navadno v dejstvu, da veljavna pravna načela v nekem trenutku, ob dejanskem stanju industrije ter tehnike in znanosti na nekem področju, ne zadoščajo več, torej ne ponujajo več ustreznega (popolnega) varstva. Znano je, da nobena obstoječa pravna ureditev varstva intelektualnih pravic na programski opremi ni v celoti primerna [78, 79], saj vsaka varuje le določene vidike in ga je zato za popolno varstvo že sedaj treba kombinirati z drugimi oblikami varstva intelektualne lastnine. Dejstvo je tudi, da je bil razvoj prava intelektualne lastnine ves čas omejen na širjenje seznama izdelkov, prepovedi in nedovoljenih dejanj ter izjem, ki se po neki obstoječi ureditvi varujejo, ni pa bilo konkretnih poskusov prenove sistema intelektualne lastnine v celoti. Iz tega naslova so se pojavljale tudi razne tendence po razvoju svojevrstne pravne ureditve za področje računalniških programov in programske opreme nasploh, ki so ob današnjih napredkih tehnologije in digitalizaciji informacije še toliko večje.

V številnih študijah, ki so bile v ta namen narejene, so bile izpostavljene slabosti in prednosti posamezne oblike varstva intelektualne lastnine. Nekateri avtorji ostajajo prepričani (npr. Lowenheim) [78], da je avtorskopravno varstvo, ki je v večini pravnih sistemov prevladujoča oblika varovanja računalniških programov (vsaj teoretično, saj le malo programov doseže zahtevano stopnjo originalnosti), še vedno najprimernejša oblika in tudi ponuja zadostno varstvo. Prednost avtorskopravnega varstva je tudi v tem, da je mednarodno priznana in uveljavljena oblika varstva računalniških programov.

Druga, veliko številčnejša skupina avtorjev (Davidson, Galbi, Karjala, Merges, Menell, Lemley, Samuelson, Stern) [78, 79] pa meni, da ima avtorskopravno varstvo kar nekaj pomanjkljivosti. Kot neprimerno navajajo trajanje avtorskopravnega in tudi patentnopravnega varstva, ki je za področje računalniških programov ob hitrem spreminjanju industrije programske opreme absolutno predolgo, ravno tako pa so postopki za pridobivanje patenta dolgotrajni in dragi. Dejstvo, da z avtorskimi pravicami ni mogoče varovati ideje, ki je v delu ali z delom izražena, ampak samo način, ki je bil pri stvaritvi dela uporabljen za izrazitev te ideje, poraja v praksi glede pravic na računalniškem programu številne probleme oziroma odprta številna vprašanja [78, str. 33 – 147]. Računalniški programi so opredeljeni kot avtorska dela, ne pa kot izdelki (ob pojavitvi računalniških programov se je avtorsko pravico, ne glede na kompleksnost problema, preprosto razširilo na področje računalniških programov in je do danes tudi tako ostalo), ureditev avtorskih pravic in upravičenj, ki iz njih izhajajo, v pogodbah s področja izdelave in vzdrževanja aplikacijskih sistemov je pogosto pomanjkljiva ali glede na naravo posla celo neprimerna, kar vodi k neoptimalnim rešitvam in dodatnim problemom, velik problem varstva pravic intelektualne lastnine, posebno avtorskih, v povezavi z internetom, pa še dodatno poraja dvome in vprašanja o ustreznosti avtorskopravnega varovanja programske opreme. Prekrivanje obstoječih oblik varstva ustvarja na tržišču zmedo pri distribuciji programov, ker ni vedno jasno, katere pravice veljajo za določen program. Problem predstavlja tudi dejstvo, da avtorskopravno varovanih del ni mogoče modificirati, jih prevajati, vprašljivo pa je tudi njihovo kombiniranje z drugimi deli.

V nekaterih evropskih državah [81] so prišli do zaključka, da bi lahko obstajala možnost t.i. dvojnega varstva oziroma dveh tipov varstva pravic intelektualne lastnine na programski opremi: popolno avtorskopravno varstvo za programe, ki dosegajo kriterije originalnosti in omejeno (zmanjšano) varstvo za preostale programe, to pomeni, da bi se ustvarile neke sorodne pravice za računalniške programe.

3.1.4 Digitalna intelektualna lastnina

Danes je večina avtorskih del tudi digitaliziranih ali celo samo v digitalni obliki. Oblike produkcije, shranjevanja in distribuiranja programske opreme in informacije nasploh so se z razširjeno uporabo internetnega omrežja bistveno spremenile. Kopiranje in (ponovna)

distribucija digitalne informacije sta poceni in preprosta. Tehnologija na eni strani omogoča izboljšanje dostopa do znanja in prenosa tega znanja, po drugi strani pa nepooblaščeno ustvarjanje neomejenega števila visoko kakovostnih kopij varovanega dela ter skoraj hipno in geografsko neomejeno distribucijo, dela pa lahko na spletu objavi vsak, ki ima do njega dostop. V povezavi z dostopanjem in pridobivanjem informacij oziroma izdelkov prek interneta pa se pojavlja tudi vprašanje zasebnosti, saj tehnična infrastruktura interneta omogoča preprosto in poceni zbiranje ter procesiranje velikih količin podatkov o uporabnikih in njihovih navadah[57]. Ta dejstva prav gotovo močno vplivajo na spremenjeno vlogo "klasičnih" avtorskih pravic in pravic intelektualne lastnine nasploh v digitalnem okolju. Tradicionalna izključna avtorska pravica o prepovedi oziroma omejitvi uporabe varovanega dela je v digitalnem okolju brezpredmetna in jo je nemogoče uveljavljati. Zato je treba vzpostaviti mehanizme, ki bodo še vedno omogočali neposredni dostop do varovanih del v digitalni obliki, uporaba teh del pa se bo ustrezno zaračunavala. Torej gre za neke vrste prehod od paradigme izključnosti k paradigmi kompenzacije.

Eno najpomembnejših vprašanj, na katero morajo imetniki avtorskih pravic odgovoriti, je, ali želijo z varstvom svojih pravic povečati število pooblaščenih dostopov do svojih del (po Gervaisu t.i. pozitivno licenciranje) ali zmanjšati število nepooblaščenih dostopov oziroma ponovnih uporab teh del (t.i. negativno licenciranje) [72]. Negativno licenciranje je mogoče doseči z uporabo tehničnih ukrepov za omejevanje in nadzor dostopov do varovanega dela[58], pozitivno licenciranje pa je usmerjeno k oblikovanju pogodbenih določil in pogojev na uporabniku čim bolj prijazen in sprejemljiv način ter čim bliže njihovim zahtevam in potrebam[59].

V kolikor bo pravno varstvo digitalne intelektualne lastnine neučinkovito, se bodo imetniki avtorskih pravic vse bolj zatekali k tehnološkim oblikam zaščite svojih del. Ker je za vsak tehnološki ukrep mogoče najti tudi tehnološki protiukrep, pa varstvo pravic samo s

[57] V EU je bila leta 1995 sprejeta Direktiva o varstvu pred nepooblaščenim zbiranjem, obdelovanjem in ponovno uporabo osebnih podatkov, ki bi uporabnike digitalnih virov obvarovala pred takim nepooblaščenim zbiranjem podatkov.
[58] Z uporabo digitalnih kontejnerjev oziroma elektronskih ovojnic, raznih metod šifriranja, zbiranja informacij o uporabnikih, omejevanje dostopov (sistemi DRM – *Digital Rights Management*), tehnične samopomoči (ang. *Technical Self-Help*) ipd.
[59] Uporabniku pred nakupom omogočiti pregled pogojev in dovoljenih načinov uporabe, po nakupu pa morebitno povečanje obsega pridobljenih pravic.

tehnološkega vidika ne bo popolno. Zato bo treba najti tudi ustrezne pravne mehanizme. Internetnega prava danes še nimamo, čeprav se spremembe na področju pravnega varstva digitalne intelektualne lastnine premikajo v to smer.[60] Že veliko število podpisnic Svetovne konvencije o avtorski pravici (ang. *WIPO Copyright Act*) je dovolj resen pokazatelj, da obstaja enotno mnenje o tem, da je treba varstvo digitalnih del urediti na mednarodnem nivoju. Kot rezultat je v ZDA leta 1998 že prišlo do spremembe njihovega Copyright Acta v t.i. DMCA – Digital Millenium Copyright Act, ki upošteva že precej problemov avtorskih pravic v internetu, v EU pa do sprejema Direktive o informacijski družbi, ki ureja predvsem pravice reproduciranja, distribucije ter priobčitve javnosti, pa tudi pravno varstvo naprav, ki preprečujejo kopiranje in sistem upravljanja pravic. Z vse bolj razširjenim elektronskim poslovanjem, ki je vzpostavilo zahteve in potrebe po elektronskem sklepanju pogodb ter plačevanju z elektronskim denarjem, vse lažjim in vedno bolj množičnim dostopom do digitalnih vsebin prek interneta ter globalizacijo tržišč je pričakovati tudi nove ali dopolnilne pravne ukrepe, ki bodo v sodobnem digitalnem okolju ustrezno urejali pravice intelektualne lastnine in njeno poslanstvo.

3.1.5 (Ne)ustreznost obstoječega pravnega varstva programske opreme

Nobena od obstoječih pravnih ureditev pravic intelektualne lastnine na področju računalniške tehnologije torej ne ustreza povsem kontekstu programske opreme in ne zadostuje za njeno popolno pravno varstvo. Že v uvodnem delu tega poglavja smo izpostavili problem neupravičenosti poenotenja pojmov računalniški program in programska oprema ter razhajanja osnovnih definicij med računalniško in pravno stroko, v nadaljevanju pa bodo dodatno izpostavljeni številni problemi in pravna vprašanja, povezani s sodobnimi načini razvoja programske opreme, konceptom programskih komponent in njihove ponovne uporabe ter knjižnic teh komponent, ki še dodatno govorijo v prid svojevrstne pravne ureditve pravic intelektualne lastnine na programski opremi.

Nove oblike trgovanja z informacijami in izdelki v digitalni obliki ter nova sredstva za njihovo distribucijo porajajo tudi zahteve in potrebe po novih pravilih, tudi pravnih. Za

[60] Dobra primera sta prav gotovo monografski publikaciji Zakon o elektronskem poslovanju in elektronskem podpisu s komentarjem z leta 2002 [85] ter Internet in pravo z leta 2003 [80].

zagotovitev ustreznega varstva pravic na teh informacijah in izdelkih je nujno potrebna sinergija med pravom in tehnologijo. Osnovno vodilo, ki ga je treba pri oblikovanju tehnoloških in pravnih ukrepov upoštevati, je, da je treba zagotoviti pravo ravnovesje med varstvom pravic intelektualne lastnine in zagotavljanjem dostopa do znanja in produktov znanja. V tej smeri bi tudi morala biti oblikovana bodoča pravna ureditev za področje intelektualne lastnine na programski opremi.

3.2 PRAVNI VIDIKI PONOVNE UPORABE

3.2.1 Pregled problematike

Osnovno vodilo za ponovno uporabo katerekoli programske komponente in pogosto tudi osnovno merilo smiselnosti ponovne uporabe neke komponente je preprosto – morebitno prilagajanje komponente specifičnim zahtevam okolja, v katerem naj bi se jo ponovno uporabilo, mora biti cenejše od razvoja te komponente na novo. Med stroške, ki nastajajo pri ponovni uporabi, pa ne sodijo samo stroški dejanskega prilagajanja oziroma spreminjanja komponente in njene integracije v novo okolje, ampak tudi stroški iskanja in izbire najustreznejše komponente, med katerimi predstavljajo stroški ugotavljanja lastništva komponente in pravic imetnika oziroma avtorja komponente pomemben del. Pravice so določene z vrsto pravnega varstva, moč varstva pa je odvisna od oblike distribuirane programske opreme[61]. Večino ponovno uporabnih komponent je prav gotovo mogoče avtorskopravno varovati, možno pa je seveda uporabiti tudi druge sisteme pravnega varstva, kot so patenti[62] in poslovna skrivnost[63] ali kar kombinacija obeh. Oblika pravnega varstva je seveda odvisna od vrste komponente in varstva, ki ga želimo doseči.

[61] Programska oprema, ki ni namenjena ponovni uporabi, se distribuira kot binarna koda, ki ji je priložena uporabniška dokumentacija, medtem ko se ponovno uporabne komponente distribuirajo predvsem kot specifikacije, načrti in izvorna koda in so zaradi tega bistveno bolj "ranljive".

[62] Podeljevanje patentov za računalniške programe niti pri nas niti v svetu še ni postala praksa, četudi se število takih patentov v zadnjih letih povečuje. Na vsak način pa patentnopravno ni varovan računalniški program kot tak, ampak v njem vgrajeni postopki in metode oziroma ideje. Zelo hitre in pogoste spremembe na področju razvojnih tehnologij programske opreme porajajo dvom o smiselnosti patentnopravnega varovanja ponovno uporabnih komponent. Postopek pridobivanja patenta je namreč zelo dolg in drag in če komponenta res ni strateško dovolj pomembna in ne predstavlja tržno zelo zanimive investicije, potem je po mojem mnenju ni smiselno patentirati. V času, ko teče postopek pridobivanja patenta, naj komponenta ne bi bila dosegljiva nikomur. Tako se lahko pripeti, da takrat, ko se za komponento podeli patent, ta po vsebinski ali tehnološki plati, ali pa po obeh, ni več zanimiva ali je že zastarela. Glede na to, da ob prijavi patenta avtor nima nobenega zagotovila, da bo patent podeljen, se lahko odloči, da posreduje komponento v knjižnico, ob poznejši podelitvi

Glede na to, da je ponovno uporabna programska oprema namenjena (in s tem namenom tudi grajena) večkratnim uporabam različnih uporabnikov (razvijalcev aplikacij) v različnih kontekstih, katerim jo je mogoče na preprost način prilagoditi, poraja tak koncept številne pravne probleme, ki jih ni mogoče preprosto rešiti z obstoječo pravno ureditvijo varovanja intelektualnih pravic na programski opremi, saj se koncept "računalniškega programa"[64], njegove namembnosti, načina uporabe itn. bistveno razlikuje od koncepta ponovno uporabnih programskih komponent. Večine ponovno uporabnih komponent pač ne moremo označiti kot računalniške programe (npr. dokumentacija, modeli, načrti, plani, specifikacije zahtev, licence, ocene, znanje o domeni) in jih zato tudi ne moremo avtorskopravno varovati kot računalniške programe ali kot njihove sestavne dele, pa tudi s patentom ne, saj ne gre za izume. Lahko pa nekatere izmed njih avtorskopravno varujemo kot pisana dela.

Problemov s ponovno uporabo in kršenjem intelektualnih pravic v praksi ni samo v najbolj osnovnem primeru ponovne uporabe, to je pri interni ponovni uporabi, torej znotraj neke organizacije, ki se ukvarja z razvojem programske opreme. V tem primeru organizacija svoje komponente hrani v interni knjižnici, dostopni samo njenim razvijalcem, lastnik te knjižnice in imetnik vseh avtorskih pravic pa je organizacija sama[65]. Problemi se začno pojavljati, ko ponovna uporaba preraste okvire organizacije. Takrat govorimo o eksterni ali zunanji ponovni uporabi, kjer razvijalci ponovno uporabne komponente pridobivajo bodisi iz neke druge knjižnice, tržišča komponent, iz javne domene, po naročilu ali kako drugače. Pri ponovni uporabi vedno nastane izpeljano ali derivativno delo, zato ponovna uporaba komponente, ki je sama po sebi deležna pravnega varstva ali je del varovane programske opreme, privede do

patenta, ko je bila komponenta že nekajkrat ponovno uporabljena, pa uveljavlja pravice iz naslova patenta "za nazaj". To lahko porodi številne probleme. Nekaj takih primerov je opisanih tudi v [78]. Za vsako uporabo s patentom varovane komponente je treba od imetnika patenta pridobiti dovoljenje za njegovo uporabo. Po 106. členu ZIL lahko nosilec patenta s pogodbo prenese v celoti ali delno svoje pravice na drugega. Licenčna pogodba mora biti vpisana v register, sicer nima pravnega učinka nasproti tretjim osebam (po 110. členu ZIL [136]).

[63] Če je programska oprema varovana kot poslovna skrivnost, potem se ne sme pojavljati v javnosti, če pa želimo pravico uporabe te programske opreme prenesti na druge, morajo ti podpisati dogovor o varovanju poslovne skrivnosti, še preden pridejo v stik z njo. Tu se pojavi vprašanje pregledovanja komponente in seznanitve z njeno vsebino pred odločitvijo ponovnega uporabnika za njeno uporabo oziroma nakup, kar naj bi bila ena osnovnih funkcij ali storitev, ki jih sodobna komponentna knjižnica ponuja. Še pred tem mora knjižnica komponento, ki je posredovana v knjižnico, ustrezno preizkusiti, oceniti in opremiti po standardih knjižnice, pri čemer seveda mora komponento (s)poznati. Po mojem mnenju vključevanje komponent, varovanih kot poslovno skrivnost, v knjižnico, ni smiselno.

[64] Glej 3.1.2.

[65] 112. člen ZASP [133].

kršitve pravic[66]. Ravno tako lahko vsakršno delo, ki nastane s spreminjanjem in prilagajanjem programskih komponent, četudi v namene vzdrževanja in izboljševanja, lahko štejemo kot izpeljano delo iz prvotne komponente, pri tem pa lahko nastopijo težave s kršenjem pravic na prvotnih komponentah.

Vsekakor je težav pri ponovni uporabi komponent brez spreminjanja oziroma prilagajanja manj. Če je ponovno uporabljena komponenta dejansko tudi originalno delo imetnika avtorskih pravic, moramo od njega predhodno pridobiti dovoljenje oziroma kupiti pravico za njeno spreminjanje. V ta namen se sklene ustrezna licenčna pogodba. Obstaja seveda možnost, da je tudi ta oseba komponento predhodno ponovno uporabila, torej jo je pridobila bodisi iz javne domene bodisi je avtor te komponente nekdo drug, kar je pri ponovni uporabi povsem običajno. Tu se postavi kopica vprašanj v zvezi z lastništvom nad posameznimi deli izpeljanega dela oziroma izpeljanih del, v katerih so komponente ponovno uporabljene ali so celo sestavljene iz več ponovno uporabljenih komponent različnih avtorjev, vprašanja v zvezi z varovanjem izpeljanega dela kot individualne intelektualne stvaritve ponovnega uporabnika, s pravicami avtorja, ko za neko komponento podeli eno ali več licenc za njeno ponovno uporabo ipd.

Tudi pri ponovni uporabi s predhodno prilagoditvijo komponente (siva in bela ponovna uporaba), kar predstavlja najpogostejši primer ali način ponovne uporabe, se pojavlja vprašanje lastninske pravice na komponenti, ki je bila pri ponovni uporabi spremenjena (ali je lastništvo odvisno od količine narejenih sprememb, namena ponovne uporabe, pomembnosti komponente ipd.), vprašanje lastništva in nadaljnje ponovne uporabe spremenjene komponente, lastništva končnega izdelka, v katerem je vsebovana tudi spremenjena komponenta itn.

Podobna vprašanja v zvezi z lastništvom in kršitvami intelektualnih pravic se porajajo tudi v povezavi s knjižnico, v kateri so shranjene ponovno uporabne komponente. V knjižnicah so

[66] Po 113. členu ZASP [133] sodijo prevajanje, prilagajanje in prirejanje ter drugo predelovanje računalniških programov ali njihovih rezultatov med izključne pravice avtorja. To pa predstavlja aktivnosti, ki so sestavni (bistveni) del procesa ponovne uporabe neke programske komponente, ko gre za ponovno uporabo s prilagajanjem.

navadno zbrane komponente različnih pravnih kategorij iz različnih virov[67]. Za učinkovito ponovno uporabo je potrebna učinkovita knjižnica ponovno uporabnih komponent. K njeni učinkovitosti prav gotovo lahko veliko prispevajo tudi postopki, ki so kakorkoli povezani z označevanjem pravic na komponentah, sledenjem uporabi in spremembam komponent, sprejemanjem komponent v knjižnico ter dodeljevanjem pravic na komponentah njihovim uporabnikom oziroma svojim članom. Z intelektualnimi pravicami na komponentah knjižnice so tesno povezana tudi vprašanja odgovornosti, jamstev in odškodnin.

Dejstvo je, da želijo tako avtorji/razvijalci komponent na eni strani kakor komponentne knjižnice na drugi s ponovno uporabo oziroma posredovanjem komponent predvsem zaslužiti. Pri ponovni uporabi gre torej za uporabo komponent za trženje, pri čemer so komponente najpogosteje sestavni del nekega drugega izdelka. Ker to dejstvo še dodatno povečuje omenjene probleme, ga je treba pri analizi situacij, ki lahko pri ponovni uporabi privedejo do pravnih problemov, posebej upoštevati.

Na splošno bi torej lahko pravne probleme, povezane s ponovno uporabo, razdelili v tri skupine in sicer na tiste, ki so kakorkoli povezani z dejstvom, da pri ponovni uporabi vedno nastane izpeljano delo, tiste, ki so povezani z lastništvom komponent, ter na probleme, povezane z zbiranjem, ocenjevanjem, katalogiziranjem, upravljanjem in licenciranjem komponent, ki so shranjene v knjižnici. Ti problemi se navezujejo na osnovne dejavnosti in aktivnosti procesa ponovne uporabe, zato je treba njihovi analizi in iskanju njihovih rešitev posvetiti dovolj prostora, sicer bo ponovna uporaba še naprej omejena na ponovno uporabo standardnih komponent (črna ponovna uporaba), ki pri razvoju aplikacij iz posameznih specifičnih domen ne predstavljajo bistvenega dela končnega izdelka in zato ne moremo govoriti o razširjeni, sistematični ponovni uporabi ter od nje pričakovati napovedovanih prednosti in pridobitev.

V naslednjih podpoglavjih so nekateri najočitnejši in najpomembnejši problemi samo izpostavljeni, njihove rešitve oziroma ukrepe, ki bi pripomogli k njihovemu reševanju, pa smo umestili v naslednja poglavja, kjer so podrobno opisani sam proces ponovne uporabe z vsemi

[67] Komponente, ki vsebujejo za organizacijo strateško znanje, komponentne s posebno komercialno vrednostjo, splošne komponente, za domeno specifične komponente, javnodomenske komponente itn.

aktivnostmi ter knjižnice ponovno uporabnih komponent. Licencam in pogodbam, kot najmočnejšim pravnim sredstvom za preprečevanje kršitev pravic na programski opremi, tudi ponovno uporabni, smo namenili samostojno poglavje.

3.2.2 Problemi izpeljanih del

Po avtorskem pravu je izpeljano delo[68] opredeljeno kot delo ali stvaritev, ki izvira iz drugega dela ali gradiva. Izpeljano delo nastane kot rezultat neke predelave avtorskega dela[69], pri čemer predelava lahko pomeni neko dejavnost v smislu predelovanja ali pa rezultat te dejavnosti[70]. Izpeljano delo, ki je individualna intelektualna stvaritev, se obravnava kot samostojno avtorsko delo[71].

Ne glede na uporabljeni tehnološki model razvoja programske opreme na temelju ponovne uporabe se vedno srečamo s pojavom, ko se pri gradnji novega programskega izdelka, ki je bodisi samostojna aplikacija ali večja, kompleksnejša ponovno uporabna komponenta, uporabi komponente, ki že obstajajo, kar pomeni izpeljavo nekega dela iz drugega. Veliko teh komponent je morda tudi last organizacije, vedno pogosteje pa razvijalci posegajo tudi po tujih komponentah, zbranih v, najpogosteje spletnih, knjižnicah. Pred uporabo tujih komponent je treba od njihovih avtorjev pridobiti dovoljenje za njihovo uporabo. Ravno tako morajo avtorji privoliti v vsako pomembnejše spreminjanje teh komponent – "... prevod, prilagoditev, priredbo ali kakšno drugačno predelavo ..."[72], saj sodijo reproduciranje, predelave in distribucija med izključne pravice avtorja[73]. Lahko pa avtor te izključne pravice delno ali v celoti prenese na tretje osebe z licenčno pogodbo.

Izpeljava del je tako bistveni sestavni del, lahko bi rekli kar nujni del procesa ponovne uporabe neke komponente. Prvotno delo (komponenta, ki je ponovno uporabljena) je lahko vgrajeno v izpeljano delo v spremenjeni ali nespremenjeni obliki, lahko pa je to delo do take

[68] V samem ZASP [133] in komentarju k ZASP [104] zasledimo za izpeljana dela tudi pojme odvisna, izvedena ali derivativna dela.
[69] 7. člen ZASP [133].
[70] V [104] je predelava definirana kot "ustvarjalna sprememba, poustvaritev ali preustvaritev prvotnega dela".
[71] 7. člen ZASP [133].
[72] 2. točka 113. člena ZASP [133].
[73] 113. člen ZASP [133].

mere predelano, da postane nov programski izdelek (nova različica komponente). Programski izdelki, ki so rezultat razvoja na temelju ponovne uporabe, so navadno oziroma vedno sestavljeni ali izpeljani iz večjega števila ponovno uporabljenih komponent različnega izvora[74]. Ker so ta izpeljana dela najpogosteje namenjena nadaljnjemu trženju, je treba za vsa prvotna dela, ki so bila uporabljena, pridobiti ustrezna eksplicitna dovoljenja imetnikov pravic na njih, kajti (ponovna) uporaba brez njihovega predhodnega dovoljenja ali soglasja pomeni po avtorskem pravu kršitev dveh izključnih pravic avtorja: pravico reproduciranja in pravico predelave[75]. Do kršenja pravic pri ponovni uporabi ne pride samo v primeru, ko je avtor izpeljanega dela hkrati tudi avtor prvotnega dela.

Izpeljana dela pa ne nastajajo samo pri neposredni gradnji ali razvoju novih programskih izdelkov iz obstoječih. Že s prevajanjem kodnih komponent iz enega v drug programski jezik ali spreminjanjem oziroma prilagajanjem kode za potrebe izvajanja na drugi strojni opremi proizvedemo izpeljana dela. Tudi vse oblike računalniško ustvarjenih del se lahko obravnavajo kot izpeljana dela. Še bolj kritična dejanja v smislu izdelave izpeljanih del pa so izvajanje vzdrževalnih del nad komponentami, torej popravljanje oziroma odpravljanje napak in pomanjkljivosti, nadgrajevanje funkcionalnosti komponent ali izboljševanje njihovih lastnosti. Zelo pogosto se namreč pripeti, da navkljub skrbnemu in natančnemu preskušanju določene programske komponente v času njenega razvoja na napako naletimo šele, ko to komponento integriramo v večji sistem, torej neposredno pri njeni ponovni uporabi, čeprav smo jo uporabili brez prilagajanja. Velikokrat je treba funkcionalnost za ponovno uporabo pridobljenih komponent nadgraditi ali izboljšati zaradi vpeljave tehnoloških sprememb v razvoj sistemov. Skratka, tudi tako zaželeni in v primerjavi s strojno opremo opevani lastnosti programske opreme, kot sta njena vzdrževalnost in prilagodljivost, lahko povzročata kopico težav. Namreč, vsa omenjena dejanja lahko avtor in/ali lastnik programskih komponent prenese na njihove uporabnike z licenco, vendar so zaradi številnih izjem in nejasnosti ali nedorečenosti, ki izhajajo iz pravnih aktov, te licence zelo kompleksne, dogovarjanje o pogojih uporabe in obsegu pravic, ki se prenašajo na licencojemalca, pa dolgotrajno. Če upoštevamo, da lahko kompleksen aplikacijski sistem sestavlja tudi nekaj tisoč ponovno

[74] Od različnih avtorjev.
[75] Pravica predelave pomeni po 33. členu ZASP [133] izključno materialno avtorsko pravico, da se neko prvotno delo prevede, priredi, spremeni ali kako drugače predela (1. točka člena), nanaša pa se tudi na primere, ko se nespremenjeno prvotno delo vgradi v novo delo (2. točka člena). Pri slednjih gre v bistvu za neke vrste reprodukcijo nekega dela v novem avtorskem delu.

uporabljenih komponent in da je treba za vsako skleniti licenčno pogodbo, hitro ugotovimo, da postane ponovna uporaba prej zaviralni faktor v programskem razvoju, ki je tudi finančno zelo zahteven, kakor pa spodbujevalni. Licencam se ni mogoče izogniti. Vendar bi bilo mogoče že s prilagoditvijo obstoječe zakonodaje, ki bi upoštevala tudi načela in aktivnosti ponovne uporabe, te licence bistveno poenostaviti, število izjem in posebnosti, ki bi jih bilo treba v njih specificirati, pa omejiti.

3.2.2.1 Problem lastništva izpeljanega dela

Naslednji problem, ki se tudi pojavlja pri izpeljanih delih, se nanaša na lastništvo. Gre za lastništvo izpeljanega dela in lastništvo pravic avtorja prvotnega dela v izpeljanem delu. Če se v licencah za ponovno uporabne komponente lastništvo in pravice na izpeljanem delu ne opredelijo, nastanejo pri posredovanju teh komponent v knjižnico ali pozneje iz knjižnice ponovnim uporabnikom, težave.

Izpeljano delo je kot rezultat predelave samostojno, novo avtorsko delo[76]. Avtor izpeljanega dela na svojem delu uživa vse pravice avtorja, v prometu pa jih prenaša skupaj z avtorjem prvotnega dela, ki ima pravico nadaljnjega izkoriščanja novega dela[77], saj je njegovo prvotno delo povzeto v novem delu, oziroma je njegov sestavni del in skupaj predstavljata nedeljivo celoto. Avtor lahko prenese pravice predelave na tretje osebe z licenčno pogodbo, oziroma jo mora v primeru ponovno uporabnih komponent prenesti na uporabnike teh komponent, sicer ponovna uporaba nima smisla, saj v vsakem primeru pomeni kršitev avtorskih pravic. Prenos pravic v bistvu pomeni prodajo teh pravic. Cena mora biti dovolj visoka, da se bo prodajalcu (licencodajalcu) splačalo pravice prodati, poleg tega pa dovolj nizka, da se bo licencojemalcu splačalo te pravice pridobiti.

V kolikšnem obsegu se pravice prenašajo, je odvisno od primera do primera. Vsekakor pa mora ponovni uporabnik pridobiti nek osnovni nabor pravic na komponentah in spremljajoči dokumentaciji, da si s tem ustvari možnosti uporabe, vzdrževanja in nadgrajevanja ponovno uporabljene komponente ter se tako izogne vsem kršitvam avtorskih pravic. Če bo aktivnosti

[76] 7. člen ZASP [133].
[77] 2. točka 7. člena ZASP [133].

vzdrževanja za ponovnega uporabnika izvajala tretja oseba, si mora ponovni uporabnik izboriti tudi pravico podeljevanja podlicenc za te aktivnosti. Na drugi strani pa mora licencodajalec zaščititi svoje interese. Nenazadnje je komponento izdelal za prodajo in gotovo mu ni v interesu prepustiti oziroma prenesti vseh pravic, posebno materialnih, na njene uporabnike. Zato licencodajalci navadno podelijo le omejene pravice[78].

Včasih je za vzdrževanje in nadgradnjo licenciranih komponent potreben tudi dostop do programskih orodij, ki jih je razvijalec uporabil pri razvoju teh komponent. Vendar je zaradi prevelikega razvijalčevega interesa na eni ter industrije in orodij na drugi strani verjetnost, da bo uporabnikom komponent pravica dostopa do teh orodij podeljena, zanemarljivo majhna. Postavi pa se tudi vprašanje ponovne uporabnosti tako "odvisnih" komponent.

Neposredno povezan z izdelavo izpeljanih del je tudi *problem nadaljnjega licenciranja izpeljanega dela*[79], ob predpostavki, da gre zopet za ponovno uporabno komponento. Čeprav poskrbimo za ustrezne oznake avtorstva posameznih ponovno uporabnih komponent, ki smo jih vključili v izpeljano delo, se postavi vprašanje, ali lahko izpeljano delo licencira dalje njegov avtor sam ali potrebuje za to privolitev vseh "soavtorjev" ali avtorjev prvotnih komponent, iz katerih je izpeljana komponenta sestavljena. Še večji problem se pojavi, če so bile pridobljene pravice za te komponente različne ali če so avtorji prvotnih komponent različno omejili avtorja izpeljanega dela pri podeljevanju pravic predelave in nadaljnje distribucije izpeljanega dela.

Pogost pojav pri ponovni uporabi je izdelava *različice neke komponente*. V bistvu ta nastane takrat, ko funkcionalnost neke komponente v taki meri spremenimo ali nadgradimo, da ta postane nova komponenta, novo avtorsko delo. Tako lahko kaj hitro naletimo na situacijo, ko imamo v isti komponentni knjižnici nekaj različic ene in iste komponente. Različice so tudi izpeljana dela, pri njihovi izdelavi pa vedno pride do znatnejših modifikacij[80] prvotne

[78] Omejitev uporabe ponovno uporabnih komponent se lahko nanaša na uporabo samo pri izdelavi konkretne aplikacije, na število uporab, zahtevano dodatno plačilo za "posebne" uporabe, prepoved trženja izpeljanega dela, v katerem je neka komponenta uporabljena ipd.
[79] 78. člen ZASP [133] govori o nadaljnjem prenosu in pravi, da "imetnik, na katerega je bila prenesena materialna avtorska pravica ali druga pravica avtorja, ne more brez dovoljenja avtorja te pravice prenesti naprej na tretje osebe, če ni s pogodbo drugače določeno".
[80] Modifikacije morajo biti vsebinsko dovolj velike, sicer se komponenta ne upošteva kot različica, ampak samo kot prilagojena komponenta. Glavna razlika med različico in prilagojeno komponento je ta, da se različica

komponente. Če smo te modifikacije izvedli med ponovno uporabo komponente, ki ni naša last, potem teh modifikacij ne smemo dati v ponovno uporabo drugim, če ne pridobimo ustreznega soglasja imetnika pravic na prvotni komponenti. Razmišljanje v tej smeri nas kaj hitro privede do dileme, ali bi potemtakem lahko tudi imetniki avtorskih pravic na programskih orodjih[81] zahtevali kakršnekoli avtorske pravice na programskih izdelkih, ki bi bili razviti s temi orodji. Tudi v tem primeru bi lahko govorili o vrsti izpeljanih del. Izpeljana dela v avtorskem pravu pač niso dovolj natančno opredeljena in dopuščajo tudi tako razlago, ki pa je s praktičnega vidika nesprejemljiva.

Tudi v povezavi z *dokumentacijo* se lahko pojavijo pravni problemi. Dokumentacija namreč predstavlja pomemben sestavni del vsake ponovno uporabne komponente. Del dokumentacije predstavlja t.i. razvojna dokumentacija, ki je v bistvu pripravljalno gradivo za izdelavo komponente (ki je lahko tudi računalniški program), del pa je namenjen predvsem lajšanju ponovne uporabe komponente, saj so v njem zbrane glavne lastnosti komponente, primeri uporabe, posebnosti, podatki o avtorstvu itn.[82], ki so ključnega pomena pri odločanju ponovnega uporabnika o morebitni uporabi komponente. Pripravljalno gradivo je po avtorskem pravu varovano skupaj z računalniškim programom[83] kot avtorsko delo, postavi pa se vprašanje, ali lahko tudi preostali del dokumentacije obravnavamo kot avtorsko delo in je torej tudi avtorskopravno varovano[84]. Posebnost tega dela dokumentacije je v tem (vsaj moralo bi biti tako), da se stalno spreminja in nadgrajuje. V primerih, ko avtor sam neposredno distribuira in daje v ponovno uporabo komponente, kar je zelo redek primer, avtorstvo te dokumentacije ni sporno, saj jo dopolnjuje kvečjemu avtor sam na podlagi poročil uporabnikov in lastnih izkušenj z njeno uporabo. V primerih pa, ko avtor posreduje komponento v knjižnico, pa je navadno za spreminjanje te dokumentacije oziroma dodajanje

obravnava kot samostojna, nova komponenta, ki se navadno tudi vključi v knjižnico, prilagojena komponenta pa obstaja samo v tistem konkretnem izpeljanem delu, za katerega je bila prvotna komponenta prilagojena.

[81] Npr. prevajalnikih, operacijskih sistemih, CAD/CAM sistemih ipd.

[82] Glej poglavje 5.2, v katerem je definirana vsebina ponovno uporabne komponente.

[83] 111. člen ZASP [133].

[84] V določbah ZASP, ki se nanašajo na avtorskopravno varovanje računalniških programov (111. – 117. člen), nikjer ne zasledimo obravnave programske dokumentacije, ki nastane po tem, ko je bil računalniški program že napisan, preizkušen in poskusno uporabljen, je pa nujni in obvezni sestavni del dokumentacije, ki spremlja nek računalniški program. V komentarju k 111. členu lahko preberemo, da sodi tovrstna dokumentacija skupaj z uporabniško dokumentacijo med programsko opremo. Sklepamo torej lahko, da je ne moremo obravnavati kot sestavni del računalniškega programa in torej kot taka ni neposredno varovana po 1. točki 6. člena ZASP, ki govori o tem, da sestavni deli avtorskega dela, ki so sami po sebi individualne intelektualne stvaritve, uživajo enako varstvo kot samo delo. [133]

pomembnih informacij v dokumentacijo na eni strani zadolženo osebje knjižnice, na drugi strani pa samodejni mehanizmi knjižnice[85].

Ponovni uporabnik med samo uporabo komponente lahko ugotovi, da je dokumentacija, ki jo spremlja, nepopolna in torej iz nje ni mogoče pridobiti vseh potrebnih informacij za izvedbo ustreznih ali potrebnih sprememb oziroma nadgradenj[86]. V kolikor licencodajalec ni pripravljen v nekem sprejemljivem času dokumentacije dopolniti ali je to pripravljen storiti samo za nerazumno visoko ceno, preostane licencojemalcu samo obratni inženiring, ki pa zahteva izdelavo kopije predmeta licence. Najprej se seveda postavi vprašanje, ali je izdelava kopije v namene obratnega inženirstva v takem primeru legalna. Po 115. členu ZASP reprodukcija v namene pridobivanja informacij o strukturi programa, idejah in načelih, na katerih je narejen, ni dovoljena in predstavlja kršitev avtorskih pravic.

3.2.3 Pravni problemi knjižnice ponovno uporabnih komponent

Knjižnica in vse z njo povezane aktivnosti predstavljajo osrednji del procesa ponovne uporabe. Za učinkovito ponovno uporabo je seveda pomembno, da knjižnica deluje in da je težav, povezanih s posredovanjem na eni ter uporabo komponent na drugi strani čim manj ali jih sploh ni. Ker lahko problemi pravne narave postanejo za knjižnico zelo velik problem, je treba v njeno delovanje vgraditi ustrezne mehanizme, ki bodo te probleme preprečili ali vsaj omejili. Končni cilj vseh ukrepov in postopkov za omejevanje ali reševanje pravnih problemov mora vedno biti lajšanje in spodbujanje ponovne uporabe.

Že z ustanovitvijo knjižnice ponovno uporabnih komponent se poraja cel kup obveznosti in odgovornosti do same knjižnice in do vseh, ki so kakorkoli vključeni v njeno delovanje, torej do dobaviteljev in razvijalcev komponent, uporabnikov komponent ali razvijalcev aplikacij ter nenazadnje tudi do uporabnikov aplikacij, zgrajenih iz ponovno uporabljenih komponent. Pri obravnavi knjižnice s pravnega vidika je treba prepoznati in identificirati možna tveganja ali

[85] Glej 5. poglavje.
[86] Načeloma naj bi se ob sprejemu komponente v knjižnico pregledovala tudi ustreznost dokumentacije, ki jo spremlja, ker pa se vseh primerov uporabe ne da vnaprej predvideti, ni mogoče ugotoviti, ali so v dokumentaciji zajete vse informacije, ki so lahko pri ponovni uporabi komponente potrebne. Zato naj bi knjižnice tudi imele vgrajene mehanizme povratnega sporočanja ponovnih uporabnikov, kjer bi se taki in podobni problemi lahko posredovali knjižnici in razvijalcem komponent.

izvore pravnih problemov. Veliko jih je povezanih z vprašanji intelektualne lastnine na komponentah, s pomanjkanjem komponent in informacij o knjižničnih komponentah ter uveljavljanjem pravic na komponentah. Če so našteta tveganja prevelika, odvračajo tako imetnike pravic na komponentah kakor uporabnike teh komponent od medsebojnega sodelovanja oziroma sodelovanja s knjižnico. Zato jih je treba zmanjšati ali razporediti, tako da ne bodo predstavljala prevelikih finančnih ali psiholoških bremen tistim, ki so vključeni v ponovno uporabo.

3.2.3.1 Sprejemanje komponent v knjižnico

V knjižnico se lahko posredujejo različne vrste komponent, ki so varovane z različnimi pravnimi mehanizmi (avtorskopravno, s patentom, kot poslovna skrivnost). Da ne bi prihajalo do kršitev pravic intelektualne lastnine na teh komponentah, se mora knjižnica ustrezno zaščititi. Glede na to, da mora nad komponentami, ki jih sprejema, izvesti določene postopke (certificiranje, klasificiranje, katalogiziranje, preskušanje, dokumentiranje ipd.) že pred njihovim sprejemom, in glede na vlogo, ki jo ima knjižnica pri nadaljnjem posredovanju teh komponent, mora priti do prenosa določenih pravic od avtorja ali dobavitelja komponente na knjižnico. Lahko se tudi zgodi, da knjižnica komponento zavrne, ker ne ustreza kriterijem kakovosti, ki jih je postavila za svoje komponente. Neustreznost komponente pa knjižnica ugotovi šele potem, ko je določene postopke (najmanj certificiranje in pregled funkcionalnosti) nad komponento že izvedla. Torej je nujno, da se pravice knjižnice pred sprejemom in po vključitvi komponente v knjižnico jasno opredelijo že ob tem, ko dobavitelj pošlje knjižnici zahtevek po sprejemu komponente v knjižnico.

Da bi se knjižnica izognila kršitvi pravic intelektualne lastnine že ob sprejemanju komponente, mora vzpostaviti določene postopke iskanja ali ugotavljanja izvora komponente. Nekaj informacij lahko pridobi iz javno dostopnih registrov (npr. urada za zaščito avtorskih pravic ali patentov), drugih javno razpoložljivih informacij ter poizvedovanj pri dobavitelju komponente. Ti postopki so lahko za knjižnico zelo zamudni in dragi, poleg tega pa tveganja še vedno ostanejo, saj nam razpoložljive javne informacije ne dajejo nobenih zagotovil, da komponenta ne bi bila varovana ali v postopku pridobivanja patenta. Zato predlagam, da bi knjižnica od dobaviteljev komponent zahtevala, da ob posredovanju zahtevka za sprejem

komponente v knjižnico podajo tudi vse informacije o izvoru in morebitnih oblikah varstev pravic intelektualne lastnine na komponentah. Hkrati mora dobavitelja tudi obvezati, da bo v primeru posredovanja netočnih ali nepopolnih informacij sam odgovoren za nastalo škodo. Smiselno je namreč predpostaviti, da je ravno dobavitelj komponent tisti, ki najbolje ve, ali je kršil pravice intelektualne lastnine nekoga drugega ali ne. Če je dobavitelj hkrati tudi razvijalec komponent, potem bo nedvomno pripravljen sprejeti tako odgovornost. Postavi pa se vprašanje, kako je s pripravljenostjo dajanja takih zagotovil in sprejemanja takih odgovornosti tistih dobaviteljev, ki so samo posredniki komponent različnih razvijalcev. Interes dobaviteljev lahko knjižnica poveča z ustrezno finančno stimulacijo. Vsekakor pa je zelo verjetno, da bodo dobavitelji zahtevali, da so njihove odgovornosti omejene, saj bi bilo sicer tveganje za njih preveliko in se zato ne bi odločali za posredovanje komponent v knjižnico, s čimer bi bila tudi ponovna uporaba prizadeta, knjižnice ponovno uporabnih komponent pa bi izgubile svoj smisel.

Veljavne pravne ureditve s področja varovanja pravic intelektualne lastnine dajejo lastnikom komponent izključne pravice do uporabe, reproduciranja in spreminjanja varovanega dela, kar pomeni, da morajo tako knjižnica kakor ponovni uporabniki za vsako tako dejanje pridobiti ustrezno dovoljenje imetnika pravic. To je očiten znak, da nobeden od obstoječih zakonov ni bil načrtovan tako, da bi lajšal ali podpiral ponovno uporabo. Zato mora knjižnica sama poskrbeti, da od imetnikov pravic na komponentah pridobi nek osnovni nabor pravic, ki ji omogoča izpolnjevanje njenega poslanstva. V ta nabor gotovo sodijo pravica do izdelave kopij varovanega dela, do izvedbe ustreznih modifikacij in izboljšav ter do izvajanja aktivnosti vzdrževanja komponent. Dejansko imetnik pravic nad komponento že v trenutku, ko komponento posreduje v knjižnico za ponovno uporabo, privoli v prenos omenjenih pravic, kajti sicer izdelava in posredovanje ponovno uporabne komponente nimata smisla. V kolikor gre za zelo specifične komponente, ki so tržno tudi izredno zanimive ali celo vključujejo določen *know-how*, ki ga je treba posebej zaščititi, lahko dobavitelj te pravice nekoliko omeji, vendar mora osnovni nabor pravic, ki omogočajo postopke sprejemanja komponente v knjižnico in nadaljnjo distribucijo teh komponent, vendarle biti prenesen na knjižnico. S tem se tudi nadzor nad nadaljnjo ponovno uporabo in morebitnim kršenjem pravic prenese iz dobavitelja komponente na knjižnico. Seveda pa dobavitelju ni v interesu prenesti na knjižnico vseh pravic, saj bi s tem sebi onemogočil svoje nadaljnje izkoriščanje

komponente, vprašljivo pa bi tudi bilo nadaljnje vzdrževanje teh komponent in odpravljanje morebitnih napak.

3.2.3.2 Opredelitev odgovornosti

Pri sklepanju pogodb med dobavitelji komponent in knjižnico se morata obe strani dogovoriti tudi za odgovornosti, ki so povezane z netočnimi ali nenatančnimi oziroma nepopolnimi informacijami o komponentah, ki se posredujejo v knjižnico. Dobavitelji komponent so odgovorni knjižnici in uporabnikom komponent, knjižnica je odgovorna vsem svojim članom, uporabniki komponent pa knjižnici, drugim uporabnikom in dobaviteljem. Gre torej za navzkrižne, prepletene odgovornosti. Vse informacije, ki jih dobavitelji posredujejo v knjižnico skupaj s komponentami, so v končni fazi namenjene uporabnikom komponent. Z njihovo pomočjo naj bi se seznanili s funkcionalnostjo komponente, njenimi tehničnimi karakteristikami, izkušnjami pri ponovni uporabi, primerih, v katerih se jo da uporabiti, in drugimi, za ponovno uporabo zelo bistvenimi stvarmi. Zato je nadvse pomembno, da so te informacije kakovostne. Določene standarde kakovosti lahko postavi že knjižnica in v postopku certificiranja, ko pregleda tudi ustreznost spremljajoče dokumentacije, komponento zaradi nekakovostne dokumentacije lahko tudi zavrne. Če je knjižnici posredovanih veliko komponent z nekakovostno dokumentacijo, ki jih knjižnica zavrne, pomeni to za knjižnico izgubo zelo veliko časa in nastanek nepovratnih stroškov. Torej bi lahko dobavitelje obravnavali kot odgovorne za škodo, ki jo utrpi knjižnica zaradi njihovega posredovanja nekakovostne dokumentacije. Morda bi se lahko tudi pripetilo, da bi knjižnica pomanjkljivosti v dokumentaciji v samem postopku sprejemanja komponente ne opazila ter komponento sprejela, pozneje pa bi zaradi njih utrpeli škodo ponovni uporabniki. Škoda se odraža predvsem v dodatnem času, ki je potreben za odpravo pomanjkljivosti in odpravo zaradi njih nastalih napak, kar pa nenazadnje pomeni tudi finančne stroške (npr. zamujanje rokov in plačila kazni). Dobaviteljem je torej v interesu posredovati kakovostne komponente s kakovostno spremljajočo dokumentacijo, kar se nenazadnje obrestuje s pogost(ejš)o ponovno uporabo komponente in večjim finančnim prilivom iz tega naslova. Zato je treba v pogodbi med dobaviteljem in knjižnico jasno opredeliti, v katerih primerih je odgovornost na dobaviteljevi strani in kakšne so posledice v primeru nastanka škode ali celo izgub na strani ponovnega uporabnika.

Knjižnica je velika podatkovna baza, v kateri so združene komponente različnih tipov ter kopica različnih informacij, ki jih knjižnica pridobiva iz vseh strani ter ponuja na voljo svojim članom – dobaviteljem ali razvijalcem komponent in razvijalcem aplikacij ali ponovnim uporabnikom. Knjižnica seveda nosi odgovornost, da bo posredovala točne, natančne in popolne informacije, kakovostne komponente, hkrati pa preprečila prosto posredovanje vseh varovanih komponent in informacij. Zaradi naštetih odgovornosti mora biti knjižnica skrbno načrtovana, veliko pozornosti mora biti posvečene preverjanju vstopnih informacij in pa dostopom oziroma omejitvam dostopov do teh informacij. Če knjižnica vsako kandidatno komponento podvrže tudi postopku ocenjevanja kakovosti oziroma doseganja kriterijev za sprejem v knjižnico in to informacijo ponudi svojim članom, postane odgovorna za morebitne napake ali pomanjkljivosti pri ocenjevanju. Tu je knjižnica odgovorna tako dobavitelju, ki mu z zavrnitvijo komponente ali z neupravičeno slabšo oceno kakovosti komponente povzroči škodo, kakor do ponovnih uporabnikov, ki so z neprimerno visoko oceno kakovosti komponente zavedeni.

Sodobna knjižnica mora biti sposobna sprejemati, ustrezno obdelati in shraniti povratne informacije ponovnih uporabnikov, ki se nanašajo na komponente. Take informacije so lahko izjemno koristne za knjižnico in za nadaljnje uporabe komponent ali za njihovo izboljšavo in nadgradnjo. Ravno zaradi pomembnosti teh informacij za knjižnico je smiselno poročanje uporabnikov ustrezno nagrajevati. Kaj hitro pa se lahko pripeti, da bodo ponovni uporabniki zaradi "zaslužka" s poročanjem posredovali neresnične, lahko celo škodljive informacije. S tem lahko uporabnik povzroči direktno škodo razvijalcu komponente, če se knjižnica odloči, da komponento odstrani iz knjižnice ali če od njega zahteva, da komponento ustrezno izboljša, direktno škodo lahko utrpi nek drug ponovni uporabnik, ki je bil zaveden zaradi netočnosti informacij v zvezi s komponento, škodo pa v vsakem primeru utrpi tudi knjižnica sama, ker ima z nagrajevanjem nekorektnih informacij stroške. Poleg odgovornosti končnih uporabnikov za korektno ponovno uporabo komponent obstajajo torej tudi odgovornosti korektnega povratnega poročanja knjižnici.

Odgovornosti zaradi okvarjenih komponent, to pomeni nepravilno delujočih, nedelujočih ali neustrezno delujočih komponent morajo tudi biti ustrezno obravnavane in vključene v pogodbo med dobaviteljem komponent in knjižnico. Z dobro opredeljenimi pravili knjižnice,

zahtevami ali standardi za kakovost komponent, vsebino in kakovost spremljajoče dokumentacije ter postopki in pravili ocenjevanja kakovosti ter pridobivanja informacij v zvezi z lastništvom pravic intelektualne lastnine in izvorom komponente se da v veliki meri vse obravnavane odgovornosti v veliki meri omejiti. Knjižnica mora tudi poskrbeti za podpise ustreznih pogodb in določitev vseh preostalih odgovornosti v njih, ravno tako pa jasno opredeliti, katere pravice se prenašajo z dobavitelja oziroma imetnika pravic na komponenti na knjižnico in katere s knjižnice na ponovne uporabnike.

3.2.3.3 Problemi uveljavljanja pravic

Velik izvor pravnih problemov v povezavi s knjižnico predstavlja problem uveljavljanja pravic. Bistvo knjižnice naj bi bilo, da svojim ponovnim uporabnikom prek učinkovitih iskalnih mehanizmov postreže s seznamom komponent, ki ustrezajo iskalnim pogojem, ter uporabnikom omogoči pregled funkcionalnosti in večine informacij, ki so s temi komponentami povezane. Uporabnik lahko iz množice pregledanih komponent izbere eno samo ali nobene. Seveda pa lahko med pregledovanjem pride do določenih spoznanj o komponentah, ki jih potem uporabi pri razvoju lastnih komponent, izdelanih za trženje, pri čemer gre za očitno kršitev pravic intelektualne lastnine. Postavi se seveda vprašanje, kako bo imetnik pravic nad neko komponento, ki je tako postala osnova za izdelavo druge, najverjetneje zelo podobne komponente, lahko uveljavljal svoje pravice in dokazoval "krajo". Ponovni uporabnik si lahko, če to pravila knjižnice omogočajo, neko komponento tudi sposodi. Ko jo preuči, jo vrne in izjavi, da komponente ne bo uporabil v svojem delu. Delno je po našem mnenju možno problem rešiti z ustreznim zaračunavanjem ogleda, izposoje in dejanske uporabe, vsekakor pa je treba mehanizem plačevanja dopolniti tudi z mehanizmom podpisovanja ustreznih licenc, v katerih so navedene odgovornosti in odškodnine v primeru kršitev licenčnih določb in pogojev. Natančneje so te rešitve obravnavane v 6. poglavju. Knjižnici je vsekakor v interesu, da nima težav s kršitvami pravic ne ona sama ne njeni člani, najsibodo to dobavitelji ali ponovni uporabniki. Zato si lahko kot cilj postavi, da bo sprejemala samo pravno čiste komponente, za katere predvidi, da pri njihovi ponovni uporabi ne bi smelo biti težav. Določene komponente lahko zavrne tudi zaradi nesprejemljivih omejitev, ki jih postavlja imetnik pravic na komponenti. Dober prijem se nam zdi, da bi knjižnica poleg katalogiziranja glede na klasifikacijsko shemo (glede na vsebinsko plat ali

vrsto komponente) vzporedno izvajala tudi razvrščanje v skupine ali razrede glede na možnosti ponovne uporabe v odvisnosti od vrste in moči pravnega varstva na komponenti oziroma možnosti nadaljnje distribucije komponent. Komponente se hkrati opremijo s pogoji za ponovno uporabo, tako da lahko ponovni uporabnik že med pregledovanjem komponente vidi, pod kakšnimi pogoji jo lahko uporabi (v smislu pozitivnega licenciranja)[87] in se tudi na podlagi tega odloča o njeni uporabi ali zavrnitvi. Predlagani sistem lahko knjižnica še dodatno nadgradi s sistemom dodeljevanja dostopov po nivojih, s katerim tudi ponovne uporabnike razvrsti v skupine in nekaterim omogoči dostop samo do posameznih skupin komponent.

3.2.3.4 Sledenje uporabam komponent in njihovim različicam

En velik del ali družina problemov pravne narave, povezanih s komponentnimi knjižnicami, se nanaša na sledenje nadaljnjim uporabam knjižničnih komponent ter izdelavam različic komponent. V nalogi predlagam, da se v knjižnico vgradi poseben mehanizem, s katerim bi se v ločeno podatkovno bazo, do katere bi imelo dostop samo osebje knjižnice, zapisovalo vsaj vse ponovne uporabe neke komponente, če že ne tudi informacij o tem, kdo si je katero komponento izposodil in vrnil, ne da bi jo uporabil v svojem izdelku. Tako bi lahko knjižnica v vsakem trenutku pregledala seznam ponovnih uporabnikov neke komponente, seznam komponent, ki jih je nek član ponovno uporabil, in podobno. Zelo koristna in pomembna informacija, ki jo je tudi smiselno hraniti v tej bazi, je informacija o projektih ali delih, v katerih je bila komponenta ponovno uporabljena[88] in seveda, ali je bila ponovno uporabljena s prilagajanjem ali kot črna komponenta. Iz te baze lahko knjižnica po potrebi črpa podatke o tem, koga mora obvestiti v primeru, ko se neka komponenta odstranjuje iz knjižnice, ali pa je bila zaradi napak popravljena oziroma nadgrajena, morebiti pa je bila na njeni osnovi celo izdelana nova različica neke komponente. Seveda lahko knjižnica nadzoruje oziroma spremlja samo prve ponovne uporabe, torej njenih članov, ne more pa slediti morebitnim nadaljnjim ponovnim uporabam, ko njeni člani neupravičeno podeljujejo podlicence za komponente, na katerih nimajo teh pravic. Če knjižnica v licenco za komponento vključi tudi določbe o prepovedi nadaljnje distribucije licencojemalca, se s tem znebi vseh odgovornosti, povezanih

[87] Glej poglavje 3.1.5.
[88] V licenci za komponento je lahko tudi izrecno določeno, da je licenca podeljena samo za enkratno uporabo. Tudi če je podeljena za več ponovnih uporab, je koristno shraniti informacijo o teh posameznih ponovnih uporabah. Vsaka "neprijavljena" ponovna uporaba, ki presega dovoljeno število ponovnih uporab, predstavlja kršitev.

s to vrsto kršitev. Seveda je treba poskrbeti za ustrezno varovanje zasebnosti vseh uporabnikov, dobaviteljev in ponovnih uporabnikov. Baza bi zato morala biti skrbno varovana, podatki, ki bi jih posredovala npr. posameznim dobaviteljem komponent ali imetnikom pravic na teh komponentah, pa bi morali biti ustrezno "prečiščeni" oziroma filtrirani. Zato je treba vzporedno uporabiti tudi sistem šifriranja uporabnikov, tako da identiteto posameznikov pozna samo knjižnica, ki pa je dolžna skrbeti za uporabnikovo zasebnost, saj se je za to nenazadnje obvezala že v pogodbi o včlanitvi.

Specifičnost izdelave različice neke komponente je v tem, da se navadno tudi različica ponovno uporabne komponente shrani v knjižnico. Torej, različica komponente je izpeljano delo iz neke ponovno uporabne komponente, ki je samo zase zopet ponovno uporabna komponenta. Če je avtor različice hkrati tudi avtor osnovne komponente, ni težav. Če različico izdela ponovni uporabnik, ki je z licenco za komponento pridobil tudi pravice do predelave te komponente, in to različico shrani v isto knjižnico, kjer je shranjena prvotna komponenta, tudi ni težav. Če pa poskuša različico, ki ni njegovo originalno delo, posredovati v neko drugo knjižnico kot avtorsko delo, pomeni, da so bile kršene pravice avtorja prvotne komponente.

3.2.3.5 Problem reproduciranja varovanih del

Med kršitve, ki izhajajo iz osnovnega delovanja knjižnice oziroma principa delovanja knjižnice, sodi tudi problem kopij varovanega dela. Navadno se po vseh uspešno prestalih vstopnih procedurah komponenta sprejme v knjižnico tako, da se "original" deponira na varno, za distribucijo oziroma ob vsakokratni ponovni uporabi pa se izdela kopija, ki se pošlje ponovnemu uporabniku. Glede na to, da se kopije delajo z namenom nadaljnjega trženja komponente, bi to brez posebnega dovoljenja nosilca pravic na komponenti predstavljajo kršitev njegove izključne pravice distribuiranja[89]. Torej, za vsako izdelano kopijo oziroma ob vsaki ponovni uporabi, ki se zaključi s podpisom licence, bi knjižnica morala od dobavitelja komponente oziroma imetnika pravic na tej komponenti pridobiti ustrezno dovoljenje, kar bi seveda predstavljajo veliko omejitev za učinkovitost knjižnice. Četudi se v licenci, ki jo za neko komponento dobavitelj podeli knjižnici, specificira možnost izdelave več kopij,

[89] 112. in 113. člen ZASP [133].

problema ne rešimo v celoti. Po eni strani ne knjižnica ne dobavitelj ne moreta vnaprej predvideti števila ponovnih uporab neke komponente, po drugi strani pa je z dobaviteljevega vidika preveč tvegano z licenco vnaprej omogočiti zelo veliko kopij. Problem bi bilo možno rešiti tako, da se dobavitelj in knjižnica dogovorita, da knjižnica lahko izdela poljubno število kopij, vendar mora dobavitelj za vsako ponovno uporabo, za katero je bila kopija narejena, dobiti povratno informacijo. Tej informaciji lahko sledi tudi ustrezen finančni priliv na konto dobavitelja, če je v licenci sklenjeno, da bodo ti prilivi vezani na posamezne ponovne uporabe. Drugi način za rešitev tega problema, ki je za knjižnico veliko manj obremenjujoč, pa je ta, da fizično komponenta sploh ni v knjižnici, ampak pri dobaviteljih, avtorjih ali na splošno imetnikih pravic na tej komponenti. Tako so v knjižnici samo opisi komponente, ko pa želi potencialni ponovni uporabnik videti delovanje komponente, si jo podrobneje ogledati ali komponento celo ponovno uporabiti, knjižnica pošlje ustrezno zahtevo po komponenti dobavitelju, ki poskrbi za izdelavo kopije in pošiljanje ponovnemu uporabniku. Da pri pošiljanju ne bi prišlo do težav z varnostnega vidika, da ne bi bili zanemarjeni ali izpuščeni postopki, ki jih knjižnica izvede ob neki ponovni uporabi (zapisi v posebno bazo ponovnih uporab, sklepanje standardiziranih licenc za komponente, proženje mehanizmov povratnega poročanja ipd.), je v tem primeru morda bolje, da dobavitelj na zahtevo pošlje kopijo komponente knjižnici, ta pa jo posreduje ponovnemu uporabniku. Določeno tveganje za knjižnico v tem primeru vendarle obstaja. Knjižnica namreč nima zagotovila, da bo kopija komponente, ki ji jo bo dobavitelj poslal na zahtevo, enaka originalu, ki ga je knjižnica sprejela. To pomeni, da bi morala preverjati identičnost med kopijo, ki jo je dobavitelj poslal ob prijavi in je bila sprejeta v knjižnico, ter kopijo, ki jo je poslal za ponovno uporabo. Glede na to, da knjižnica svojim članom oziroma ponovnim uporabnikom zagotavlja in mora zagotavljati varnost, kakovost in učinkovitost, je to zanjo v tem primeru tveganje gotovo precejšnje.

Tudi pri reproduciranju knjižnice bi lahko nastopile težave. Knjižnica ponovno uporabnih komponent je v osnovi podatkovna baza[90] in je kot taka posebej (avtorsko) varovana[91].

[90] Po 141a. členu ZASP [133] je podatkovna baza "zbirka neodvisnih del, podatkov ali drugega gradiva v kakršnikoli obliki, ki je sistematično ali metodično urejeno in posamično dostopno z elektronskimi ali drugimi sredstvi ...".
[91] 6. oddelek ZASP, členi od 141a do 141g [133].

Izdelovalec knjižnice ima med drugim izključno pravico reproduciranja[92] svoje podatkovne baze, kar v principu pomeni izdelavo kopij podatkovne baze v celoti ali posameznih delov. Če pa so posamezni deli, torej komponente, varovane po sistemu intelektualne lastnine[93], pa bi pri uveljavljanju te pravice lahko prišlo do nasprotujočih si interesov, ki izhajajo iz iste pravne ureditve[94]. Enako velja za pravico distribuiranja primerkov podatkovne baze, dajanja v najem in dajanja na voljo javnosti.

3.2.3.6 Drugi problemi digitalnih knjižnic

Samuelsonova [65] opozarja, da so tudi pravice indeksiranja dokumentov v (poljubni) digitalni knjižnici, ki ga izvajajo razvijalci knjižnic, vprašljive. Kot take navaja tudi povzemanje, filtriranje informacij iz dokumentov, ki jih posredujejo uporabniki knjižnice, dodajanje pripomb, zaznamkov ali celo preoblikovanje digitalnih dokumentov[95], medsebojno povezovanje knjižnic in izmenjava dokumentov ali celo komponent med knjižnicami. Internet s svojimi principi delovanja imetnike pravic na varovanih delih že dolgo sili v iskanje načinov, ki bi jim omogočali zaračunavanje uporabe njihovih del, kar jim omogoča nadaljnje vlaganje v produkcijo in distribucijo varovanega materiala. Treba pa je izbrati tak način, da za uporabnike ne bo odbijajoč. Že v poglavju 3.1.5 smo omenili, da je eno najpomembnejših vprašanj, na katero si morajo imetniki pravic odgovoriti, ali želijo z varstvom svojih pravic povečati število pooblaščenih dostopov oziroma ponovnih uporab varovanega dela ali zmanjšati število nepooblaščenih dostopov oziroma ponovnih uporab varovanega dela. Slednje je mogoče doseči z dodatnimi tehničnimi ukrepi oziroma t.i. sistemi za nadzor ali preprečitev dostopa do varovanih del ali njihove uporabe (npr. digitalni kontejnerji ali elektronske ovojnice, metode šifriranja, sistemi DRM ipd.).[96]

[92] 141c. člen ZASP [133].
[93] V 2. odstavku 141a. člena ZASP [133] sicer piše, da "Z uvrstitvijo gradiva v podatkovno bazo in z njegovo uporabo ne smejo biti prizadete pravice na tem gradivu."
[94] V tem primeru iz istega zakona.
[95] To so aktivnosti, tesno povezane s posodabljanjem dokumentacije, ki spremlja ponovno uporabne komponente.
[96] O pravni ureditvi vprašanj digitalnega okolja je več opisanega pod opombo 329 v prilogi A.

3.2.4 Nedefiniranost ponovne uporabe v obstoječih pravnih aktih

Ker ponovna uporaba in vse aktivnosti ter posebnosti v povezavi z njo v avtorskem pravu niso upoštevane kot posebna vrsta uporabe programske opreme[97], je uravnavanje avtorskih pravic na ponovno uporabnih komponentah ter vseh programskih izdelkih, ki nastanejo kot rezultat razvoja na temelju ponovne uporabe, za razvijalca ali avtorja komponent ter za njihove posrednike in uporabnike zelo zahtevno in velikokrat destimulativno, kar pomeni, da jih odvrača od ponovne uporabe. Izjema je le omemba izpeljanih del, ki jih lahko obravnavamo kot rezultat procesa ponovne uporabe. Iz problemov, ki smo jih v zvezi z izpeljanimi deli opisali v poglavju 3.2.2, pa je moč sklepati, da je tudi ta pojem obravnavan zelo površno, vsaj z vidika ponovne uporabe, saj ostaja problem prenosa pravic s ponovno uporabo rešljiv še vedno samo v domeni licenčnih pogodb, ki torej predstavljajo osnovo za prenos in definiranje pravic avtorja prvotnega in avtorja izpeljanega dela. Čeprav je sklepanje licenčnih dogovorov lahko zelo zamudno in drago, v primeru ponovne uporabe velikega števila komponent različnih avtorjev ali celo iz različnih knjižnic pa izjemno zapleteno, je zaenkrat to edini način, da se izognemo kršitvam avtorskih pravic. Licenčne pogodbe morajo biti take, da olajšajo in spodbujajo ponovno uporabo oziroma skladno obravnavajo vse oblike izpeljanih del in problemov, ki lahko pri njihovi izdelavi nastanejo. Ker so licenčne pogodbe tako pomemben faktor za pravno "čisto" ponovno uporabo, jim je namenjeno posebno poglavje tega dela, v katerem so podane tudi osnovne smernice za oblikovanje standardnih, ponovno uporabnih licenc za ponovno uporabo.

Čeprav je govora o pravnih problemih, se jim ni mogoče v celoti izogniti samo z ustrezno prilagoditvijo pravne ureditve s področja intelektualne lastnine, ampak je treba ustrezno kombinirati tehnične in pravne rešitve oziroma ukrepe. Za varovanje in uveljavljanje pravic je torej potrebna sinergija med pravom in tehnologijo. In to je bil tudi cilj, ki smo ga poskušali skozi naslednja poglavja doseči.

[97] Ponovna uporaba je zelo specifičen primer uporabe programske opreme, ki v veliki meri odstopa tudi od principov uporabe paketne programske opreme, namenjene masovnemu trženju ali po naročilu grajenih specifičnih programskih sistemov. Obstoječa zakonodaja je prirejena slednjim.

4 PROCES PONOVNE UPORABE

Ponovna uporaba spremeni življenjski cikel, ker predstavlja popolnoma drugačen način razvoja programske opreme. Tradicionalni pristop k razvoju programske opreme namreč narekuje in predpostavlja, da se vsak programski izdelek razvija od začetka, torej znova. Razvoj izdelkov znova pa je pri razvojnem ciklu na temelju ponovne uporabe skrajna izjema.

Da omogočimo ponovno uporabo, moramo procese, aktivnosti in opravila, ki se nanjo nanašajo, vključiti že v začetne faze razvojnega cikla in z njimi tudi cikel zaključiti. Napačno bi bilo z aktivnostmi ponovne uporabe zavlačevati vse do zaključka razvojnega cikla, saj bi tako zelo zmanjšali in omejili možnosti za ponovno uporabo preostalih vmesnih izdelkov (ponovna uporaba se omeji navadno na kodo), omejene pa bi bile tudi potencialne prednosti, ki jih je mogoče s ponovno uporabo doseči. Ponovna uporaba znotraj neke faze namreč ni omejena samo na to fazo, pač pa odpira možnosti ponovne uporabe tudi v poznejših fazah razvojnega cikla[98]. Če torej želimo od ponovne uporabe res pridobiti največ, kar se da, moramo z razmišljanjem in planiranjem ponovne uporabe frontalno prekriti celoten razvojni proces.

Implementacija ponovne uporabe zahteva formalizacijo izvajanja ponovne uporabe prek integriranja procesov, aktivnosti in opravil v razvojni življenjski cikel.[99] Če ponovna uporaba znotraj življenjskega cikla ni eksplicitno opredeljena, potem organizacija ne bo mogla v celoti izkoristiti prednosti, ki jih ponovna uporaba omogoča. Moč ponovne uporabe je zelo velika, če upoštevamo, da je večino programskih izdelkov mogoče zgraditi ali sestaviti iz posameznih komponent, ki so že na razpolago. Eden večjih problemov, na katerega naletijo organizacije, ki želijo izvajati ponovno uporabo, je dejstvo, da v njihovem razvojnem procesu dejansko manjkajo aktivnosti, ki bi se nanašale na ponovno uporabo in jo omogočale.[3, 4, 15, 41]

[98] Npr. ponovna uporaba zahtev pelje navadno v ponovno uporabo načrta in dalje v ponovno uporabo kode ter preskusnih elementov.
[99] Vsak proces lahko razbijemo na posamezne aktivnosti, posamezne aktivnosti pa na množico opravil.

Proces ponovne uporabe opisuje, kako zgraditi programske izdelke iz posameznih ponovno uporabnih komponent, kako ponovno uporabne komponente zgraditi ter kako jih upravljati[100]. Pri obravnavi sistematične ponovne uporabe je treba enakovredno obravnavati problem razvojnega cikla programskega izdelka in problem življenjskega cikla komponent, iz katerih se lahko ta izdelek zgradi.

Sistematična ponovna uporaba pa ne prinaša samo prednosti, ampak tudi določene stroške, ki predstavljajo tveganja in porajajo pomisleke o tem, da bi bilo sploh mogoče te prednosti doseči. Zato je pred odločitvijo o vzpostavitvi ponovne uporabe treba razmisliti o:

- stopnji relevantnosti ponovne uporabe za organizacijo[101];
- razpoložljivosti primernih orodij[102] in komponent, načrtovanih za ponovno uporabo;
- zrelosti organizacije za ponovno uporabo[103];
- pripravljenosti ljudi, ki so v organizaciji zaposleni, na spremembe v njihovem delu.

Organizacije, ki želijo uvesti sistematično ponovno uporabo, morajo svojo pripravljenost in sposobnost za sprejem vseh za to potrebnih aktivnosti in sprememb predhodno analizirati, kajti vzpostavitev procesa sistematične ponovne uporabe, poleg visoke stopnje zrelosti organizacije, zahteva tudi visoka vlaganja. Med analizo je treba natančno opredeliti cilje, stroške, tveganja in pričakovane prednosti ter pripraviti natančen časovni plan prehoda na sistematično ponovno uporabo.

Za uspešen prehod in uspešno uvedbo sistematične ponovne uporabe je torej treba:

- natančno definirati postopke pridobivanja, oskrbe (dobave), razvoja, upravljanja in vzdrževanja komponent;
- upravljati in izboljšati razvojni proces v skladu z zahtevami ponovne uporabe;

[100] V literaturi s področja ponovne uporabe pogosto zasledimo koncepte ustvarjanja ali gradnje komponent, upravljanje komponent in uporaba komponent kot sestavnih delov procesa ponovne uporabe. [1, 9, 10, 12, 13, 25, 30, 49, 50 idr.]

[101] Zelo majhni organizaciji oziroma taki organizaciji, ki ima z razvojem in vzdrževanjem programskih izdelkov zelo malo stroškov, ponovna uporaba najbrž ne bo prinesla tako velikih prednosti, da bi upravičila visoke začetne stroške, ki nastanejo z njeno vzpostavitvijo.

[102] Nabor klasičnih tipov orodij v podporo ponovni uporabi je opisan v Prilogi A.

[103] Organizacija mora imeti doseženo neko stopnjo zrelosti, tako tehnične kakor organizacijske, preden začne z uvajanjem sistematične ponovne uporabe in standardov. Dosegala naj bi vsaj 5. zrelostni nivo po zrelostno zmožnostnem modelu CMM (*Capability Maturity Model*), kar pomeni strukturiran proces razvoja, statistični nadzor kvalitete, stalno izboljševanje kvalitete. [46]

- zgraditi okolje za upravljanje in inženiring, ki sloni na ponovni uporabi, torej infrastrukturo za ponovno uporabo;
- doseči medsebojno razumevanje in sodelovanje vseh, ki so vključeni v razvojni proces na temelju ponovne uporabe;
- olajšati ponovno uporabo komponent pri razvoju programskih izdelkov ali sistemov;
- olajšati razvoj ponovno uporabnih komponent.

4.1 KLJUČNI ELEMENTI ZA PRILAGODITEV RAZVOJNEGA ŽIVLJENJSKEGA CIKLA

V literaturi s področja ponovne uporabe [seznam A] lahko zasledimo različne analize uspešnih, manj uspešnih in pa popolnoma neuspešnih poskusov vzpostavitve procesa ponovne uporabe. Iz njih lahko izluščimo tri za ponovno uporabo značilne elemente, ki jih je nujno treba vgraditi v razvojni proces na temelju ponovne uporabe, če želimo, da bo le ta uspešen. Ti so: proces je treba podpreti z razmišljanjem o ponovni uporabi, kar je predpogoj za sprejemanje kakršnihkoli odločitev; razvijalčeve pozornosti in dejavnosti je treba preusmeriti iz posameznega programskega izdelka na množico oziroma družino izdelkov; razvojni proces je treba obravnavati kot dva vzporedna, včasih tudi navidez ločena podprocesa – gradnje in uporabe ponovno uporabnih komponent.

4.1.1 Razmišljanja v smeri ponovne uporabe

Pri razvoju programskih izdelkov in sprejemanju razvojnih odločitev morajo razvijalci stalno razmišljati o vseh možnostih zvišanja stopnje ponovne uporabe. V ta namen bi bilo na primer treba:

1. *maksimalno izkoristiti obstoječe komponente bodisi pri razvoju novih izdelkov bodisi pri izboljševanju ali vzdrževanju obstoječih sistemov,* kar omogoča izkoriščanje v preteklosti narejenega dela in daje ponovni uporabi prednost pred razvojem sistemov na novo na podlagi trenutnih programskih zahtev.
2. *razpoznati možnosti za ponovno uporabo,* kar zahteva neprestano spremljanje vseh priložnosti za ponovno uporabo kateregakoli vmesnega ali končnega izdelka, ki nastane znotraj projekta (vsak izdelek potencialno ponovno uporaben).

3. *na ponovno uporabo v prihodnje se je treba pripraviti že med izvajanjem tekočega razvojnega procesa*: pri ponovni uporabi nekega izdelka je treba upoštevati tudi vpliv ponovne uporabe tega izdelka na ponovne uporabe v preostalih procesih življenjskega cikla.

Privzgojeno razmišljanje o ponovni uporabi pomeni, da se v vsakem trenutku zavedamo, kaj je že bilo zgrajenega, in to po možnosti pri gradnji novih ali izboljšanju starih izdelkov tudi ponovno uporabimo. Tak koncept razmišljanja spodbija v praksi zelo razširjen način razvijanja sistemov v stilu "kopiraj in spremeni" (prvotna, zelo razširjena oblika ponovne uporabe), ki ima za posledico veliko povečanje števila med seboj zelo podobnih komponent. Vzdrževanje teh komponent je zelo zahtevno, njihova ponovna uporabnost pa nizka, kar ponovne uporabe ne spodbuja.

4.1.2 Preusmeritev razvijalcev na družino izdelkov

Druga lastnost, po kateri se ponovna uporaba razlikuje od klasičnega razvoja, je tudi v množici specifičnih zahtev oziroma potreb, ki nastanejo znotraj življenjskega cikla kot posledica ponovne uporabe.

Razvijalci se pri postavljanju zahtev za razvijajoč izdelek ne smejo osredotočati samo na tekoči projekt, ampak morajo upoštevati tudi zahteve drugih sorodnih projektov in predvidene zahteve za bodoče projekte iste domene. Razmišljati morajo torej o družini izdelkov s sorodnimi zahtevami, lastnostmi, funkcionalnostmi itn., in to razmišljanje implementirati v eni komponenti ter nato to komponento ponovno uporabiti povsod in vedno, ko se te zahteve pojavijo oziroma ponovijo. Taka komponenta mora biti dovolj splošna, uskladiti jo je treba s standardi, uporabljena strategija načrtovanja pa mora upoštevati konsistentnost implementacij programskih izdelkov, kar vnaša v življenjski cikel določene specifične lastnosti.

Z analizama podobnosti in razlik množice obstoječih in bodočih programskih izdelkov je mogoče ugotoviti, ali je pri ustvarjanju in vzdrževanju programskih izdelkov iz te množice bolj koristno take ponovno uporabne komponente razviti ali jih pridobiti.[104]

[104] Ponovno uporabna komponenta mora namreč biti načrtovana tako, da jo je pri obvladovanju razlik v zahtevah, ki nastopijo pri ponovni uporabi, možno preprosto prilagoditi ali uporabiti celo brez prilagoditev.

Zahteve ponovne uporabe seveda vplivajo na celoten življenjski cikel in porajajo potrebo po določenih spremembah znotraj življenjskega cikla. Zato je treba posameznim fazam razvojnega cikla dodati za ponovno uporabo specifične procese, aktivnosti ter opravila.

4.1.3 Podprocesi razvojnega procesa na temelju ponovne uporabe

Razvoj na temelju ponovne uporabe spremeni tudi pogled razvijalcev na sam življenjski cikel, ki ga je treba obravnavati kot dvostranski. Na eni strani so aktivnosti, povezane z uporabo ponovno uporabnih komponent pri gradnji novih programskih sistemov ali aplikacij in tvorijo t.i. *razvoj s ponovno uporabo*, na drugi strani pa aktivnosti ustvarjanja, pridobivanja in reinženiringa ponovno uporabnih komponent ali izdelkov, ki sestavljajo t.i. *razvoj za ponovno uporabo*.

Medtem, ko lahko razvoj s ponovno uporabo dosežemo z vključitvijo nekaterih za ponovno uporabo specifičnih aktivnosti v klasični razvojni cikel, je razvoj za ponovno uporabo povsem nov proces, v celoti specifičen za ponovno uporabo. Poteka lahko povsem ločeno, se pravi, da se aktivnosti enega in drugega ne prepletajo neposredno, ali pa v simbiozi z razvojem s ponovno uporabo. Dejansko sta obe vrsti razvoja med seboj odvisni in druga drugo pogojujeta, od velikosti organizacije ter prakse ponovne uporabe v njej pa je odvisno, koliko se bosta procesa prepletala. V vsakem primeru mora med njima obstajati "povezava" oziroma pretok informacij. Navadno to nalogo odigrata knjižnica ponovno uporabnih komponent s svojim upraviteljem in drugimi udeleženci procesa upravljanja komponent, zelo pomemben pa je tudi mehanizem medsebojnega komuniciranja in posredovanja informacij.

4.1.3.1 Razvoj s ponovno uporabo

Razvoj s ponovno uporabo ali sestavljanje programskih sistemov oziroma aplikacij zahteva razmišljanje o ponovni uporabi znotraj čisto vsake razvojne aktivnosti, vključno s projektnim planiranjem, izvedbo in preskušanjem. Treba je raziskati vse možnosti razvoja programskih in iz njih izhajajočih izdelkov iz že obstoječih ponovno uporabnih komponent. Izdelava komponent na novo naj bi bila skrajna možnost za izjemne primere, ko bi bilo prilagajanje obstoječih komponent bolj potratno od razvoja komponent na novo. Osnovno vodilo za vsako

razvijalčevo odločitev naj bi bile ravno razpoložljive ponovno uporabne komponente – njihova funkcionalnost, prenosljivost, prilagodljivost, cena itn.

Znotraj življenjskega cikla, ki podpira razvoj s ponovno uporabo, je treba razširiti razvojni plan še s planom ponovne uporabe, v katerem naj bo opredeljeno, katere komponente je možno ponovno uporabiti pri izdelavi programskega sistema ali aplikacije, kateri so ciljni nivoji ponovne uporabe programskega projekta, katera orodja v podporo izvajanju ponovne uporabe se bodo uporabljala znotraj projekta ter na kakšen način se bo meril vpliv ponovne uporabe na projekt. Treba je tudi poiskati in ovrednotiti tiste dele aplikacij, ki bi jih lahko uporabili znotraj razvojnega projekta, ovrednotiti prednosti in stroške, povezane z izvajanjem ponovne uporabe znotraj projekta, poiskati knjižnice ponovno uporabnih komponent ter druge notranje in zunanje vire komponent, ki jih je možno pri izdelavi programskega izdelka ponovno uporabiti. Ob beli ponovni uporabi je treba ustvariti tudi spremljajoče izdelke (specifikacija, načrt, dokumentacija itn.) ter vključiti preverjanja ponovne uporabe v revizije uspešnosti oziroma izvajanja projekta.

4.1.3.2 Razvoj za ponovno uporabo

Medtem ko je razvoj s ponovno uporabo samo indirektno povezan z ustvarjanjem novih ponovno uporabnih komponent, je to primarna naloga razvoja za ponovno uporabo ali inženirstva domene. Razvoj za ponovno uporabo vključuje gradnjo modelov in arhitektur domene, novih ponovno uporabnih komponent ter reinženiring obstoječih komponent z namenom povečanja njihove ponovne uporabnosti.
Aktivnosti, ki jih je treba dodati razvojnemu življenjskemu ciklu, da bo prilagojen za razvoj za ponovno uporabo, so izvedba analize domene, izvedba načrtovanja domene in gradnja ponovno uporabnih komponent. Če želimo, da bo ponovna uporaba znotraj razvojnega življenjskega cikla eksplicitno izvajana, je treba:

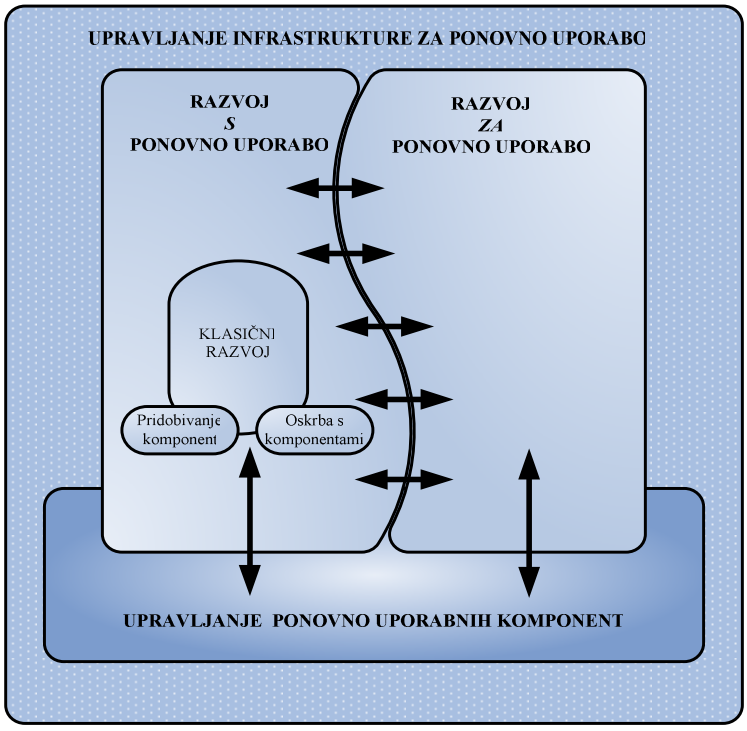
1. med izvajanjem ponovne uporabe poleg tekočega projekta obravnavati tudi pretekle in bodoče projekte[105];
2. ponovno uporabo planirati, saj je to ključno za uspeh[106];

[105] Razvojni ekipi je tako omogočeno, da izkoristi, kar je razpoložljivega za ponovno uporabo ter prepozna možnosti ponovne uporabe v bodoče.

3. vzpostaviti in vzdrževati dobro komunikacijo med udeleženci znotraj posameznega projekta (med razvijalci aplikacij in inženirji domene) ter med udeleženci različnih projektov.

4.2 OGRODJE NA PONOVNI UPORABI TEMELJEČEGA RAZVOJNEGA PROCESA

Pri definiciji splošnega ogrodja programskega življenjskega cikla, kjer razvoj poteka na temelju ponovne uporabe, je treba razlikovati med različnimi skupinami procesov. Najbolj splošna delitev, ki jo lahko zasledimo v večini literature s tega področja, je delitev na razvoj za ponovno uporabo, razvoj s ponovno uporabo in upravljanje komponent (slika 3). Pogosto je proces ponovne uporabe opisan s štirimi ključnimi aktivnostmi, to so *upravljanje infrastrukture za ponovno uporabo*, *razvijanje ponovno uporabnih komponent*, *posredovanje teh komponent* ter *njihova uporaba*.

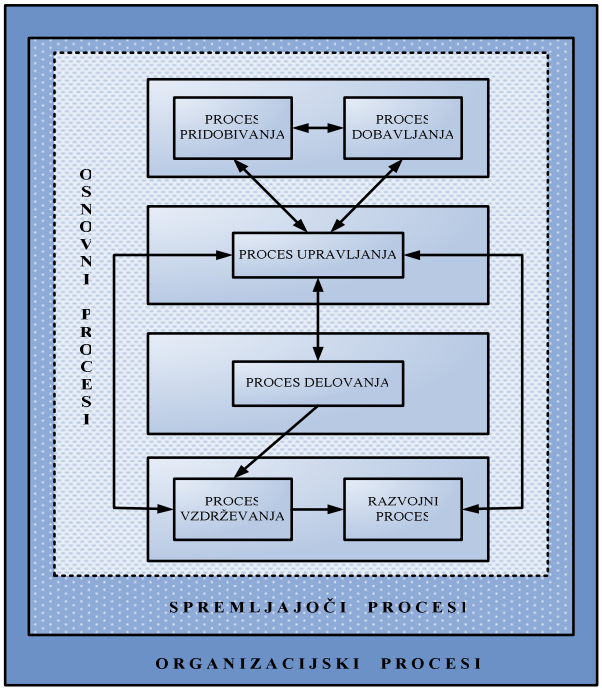


Slika 3: Povezanost in odvisnost podprocesov in aktivnosti ponovne uporabe.

[106] Prej uspemo prepoznati možnost ponovne uporabe, prej jo lahko vključimo v razvojni proces oziroma projekt. Poudarek bi moral biti na ponovno uporabnih izdelkih, ki nastanejo v zgodnjih razvojnih fazah (npr. specifikacije, domenske arhitekture, modeli), saj tak pristop ponuja oziroma omogoča večje izkoristke ponovne uporabe (večje prednosti oziroma pridobitve).

Če pa poskušamo procese razvoja na temelju ponovne uporabe ločevati glede na njihovo funkcijo, jih lahko razdelimo na *osnovne procese* (pridobivanje, oskrba, razvoj in vzdrževanje komponent), *spremljajoče procese* (izdelava dokumentacije, upravljanje konfiguriranje, zagotavljanje kakovosti, verifikacija, validacijo ter revizije programskega življenjskega cikla), *organizacijske procese* (upravljanje, izboljševanje, infrastruktura ter izobraževanje znotraj življenjskega cikla). Povezave in odvisnosti med posameznimi procesi so prikazani na sliki 4.

V nadaljevanju je predstavljeno ponovno uporabno ogrodje razvojnega življenjskega cikla na temelju ponovne uporabe, na podlagi katerega je možno vsak razvojni življenjski cikel razširiti s procesi, aktivnostmi ter opravili, ki omogočajo gradnjo programskih sistemov ali njihovih delov iz posameznih vnaprej pripravljenih komponent ter identifikacijo, gradnjo, vzdrževanje in upravljanje komponent. Ogrodje je zastavljeno splošno in opisuje proces ponovne uporabe, a se ne spušča v izvedbo posameznih aktivnosti in opravil, ki jih proces vključuje. Nadalje opisuje odgovornosti, ki so neločljivo povezane s procesi, a ne opisuje podrobnosti o povezavah med temi procesi. Ogrodje določa procese ponovne uporabe v razvojnem ciklu, a ne predpisuje specifičnega razvojnega modela ali razvojne metodologije. Proces ponovne uporabe je v ogrodju obravnavan kot zaporedje nalog in vlog posameznih udeležencev, ne pa kot zaporedje korakov ali procedur, ki jih je treba izvesti.



Slika 4: Procesi razvojnega življenjskega cikla in njihova medsebojna povezava.

Razlogi za "ohlapno" definiranje ogrodja oziroma za njegovo splošnost so v tem, da je mogoče tako ogrodje uporabiti ne glede na izbrano metodologijo razvoja in strukturo razvojnega življenjskega cikla. Zato mora biti ogrodje čim bolj univerzalno. Namen takega ogrodja je definirati procese, aktivnosti in opravila, ki so potrebni za izvajanje ponovne uporabe znotraj posameznega projekta ter med različnimi projekti in celo različnimi organizacijami. Z aktivnostmi, procesi in enotno terminologijo s področja ponovne uporabe, ki so dodani posameznim korakom življenjskega cikla, je omogočena tudi lažja komunikacija med dobavitelji, pridobitelji in razvijalci komponent, administratorji ponovne uporabe, upravitelji in vzdrževalci komponent, inženirji domene ter uporabniki komponent in programskih izdelkov. Pri tem so pridobitelji in dobavitelji lahko znotraj ali zunaj organizacije.

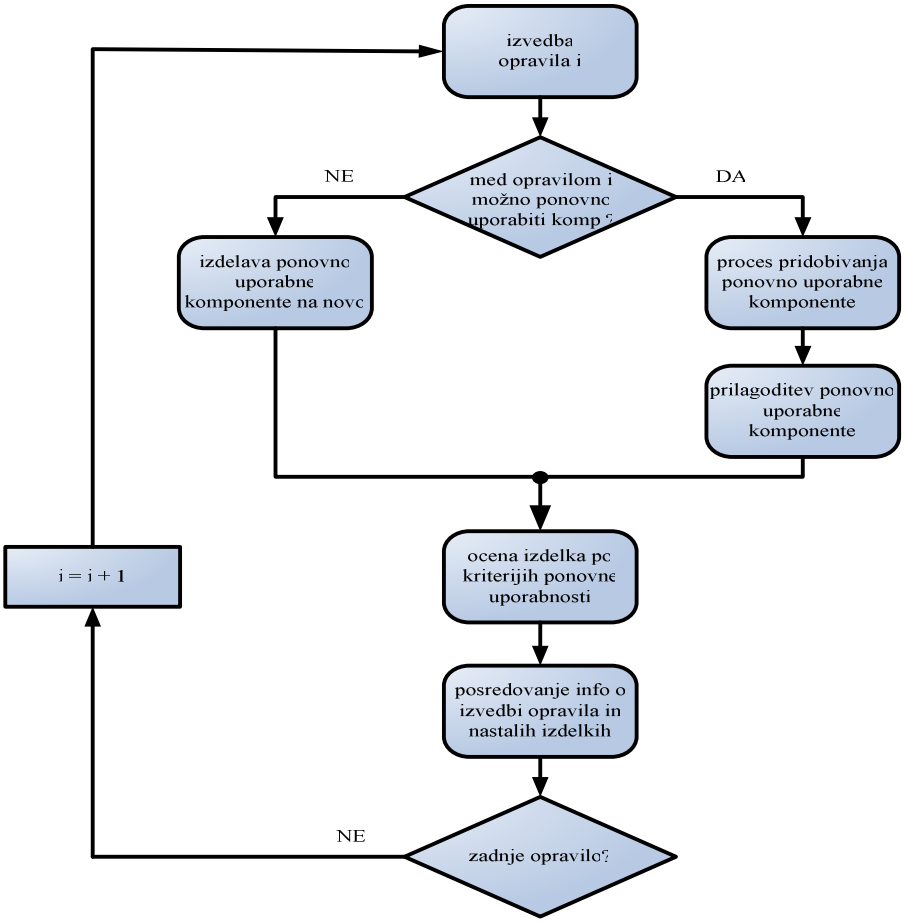
Ogrodje je uporabno tako v procesih razvoja, uporabe, vzdrževanja in upravljanja sistemov ter programskih izdelkov, skupaj s komponentami, iz katerih so zgrajeni (inženirstvo aplikacij ali razvoj s ponovno uporabo), kakor tudi v procesih razvoja, vzdrževanja in upravljanja samih ponovno uporabnih komponent (inženirstvo domene ali razvoj za ponovno uporabo). Skupaj pa ta dva procesa tvorita razvojni proces programskih izdelkov na temelju ponovne uporabe.

4.2.1 Integracija osnovnih razvojnih aktivnosti in za ponovno uporabo specifičnih aktivnosti

Aktivnosti in opravila ponovne uporabe so splošna in neodvisna od izbranega življenjskega cikla. Med razvojem, ko se implementira izbrani razvojni življenjski cikel, se te aktivnosti povežejo z drugimi osnovnimi razvojnimi aktivnostmi in se tako integrirajo v sam razvojni proces. Narava ponovne uporabe nekako zahteva, da izberemo življenjski cikel, ki omogoča evolutivni razvoj. To pomeni, da mora biti iterativen in inkrementalen ter tako omogočati postopno gradnjo končnega programskega izdelka.

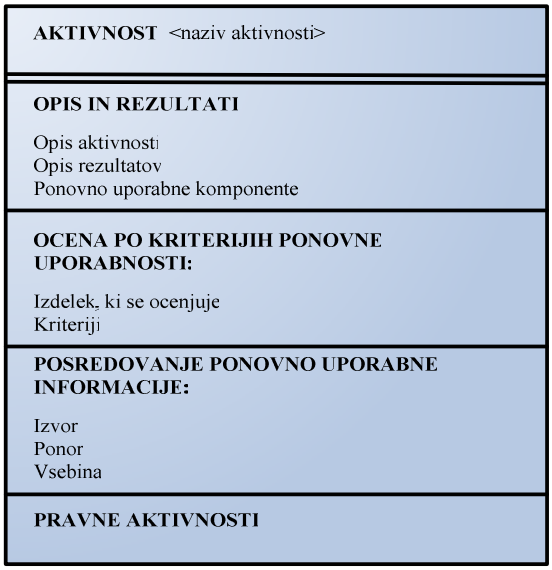
Z analizo aktivnosti, ki so specifične za ponovno uporabo [56], je mogoče izpostaviti tri korake, ki se znotraj teh aktivnosti ponavljajo in za katere lahko torej sklepamo, da sodijo med najpomembnejše korake ponovne uporabe.

1. Pri razvoju kateregakoli elementa ali izdelka je treba najprej *v čim večji meri ponovno uporabiti obstoječe komponente*, pri njihovem morebitnem prilagajanju oziroma sestavljanju v nove komponente pa upoštevati lastnosti in koncepte predhodno ponovno uporabljenih komponent.
2. Pred prehodom na naslednjo aktivnost je treba *ovrednotiti nastale izdelke tudi z vidika ponovne uporabnosti*. Osnovni kriteriji ponovne uporabnosti se nanašajo na uporabnost in ponovno uporabnost izdelka, ki je nastal kot rezultat neke aktivnosti, možnost ponovne uporabe tega izdelka v različnih okoljih (kontekstih), združljivost izdelka z modeli domene in arhitekturo domene ter skladnost izdelka s standardi ponovne uporabe organizacije.
3. Vse postopke in ocene je treba skrbno *dokumentirati* zaradi poznejših ponovnih uporab in o vseh dogodkih, ki v času izvajanja neke aktivnosti nastanejo, obveščati inženirje domene in upravitelja ponovno uporabnih komponent.



Slika 5: Diagram zaporedja izvajanja aktivnosti ponovne uporabe.

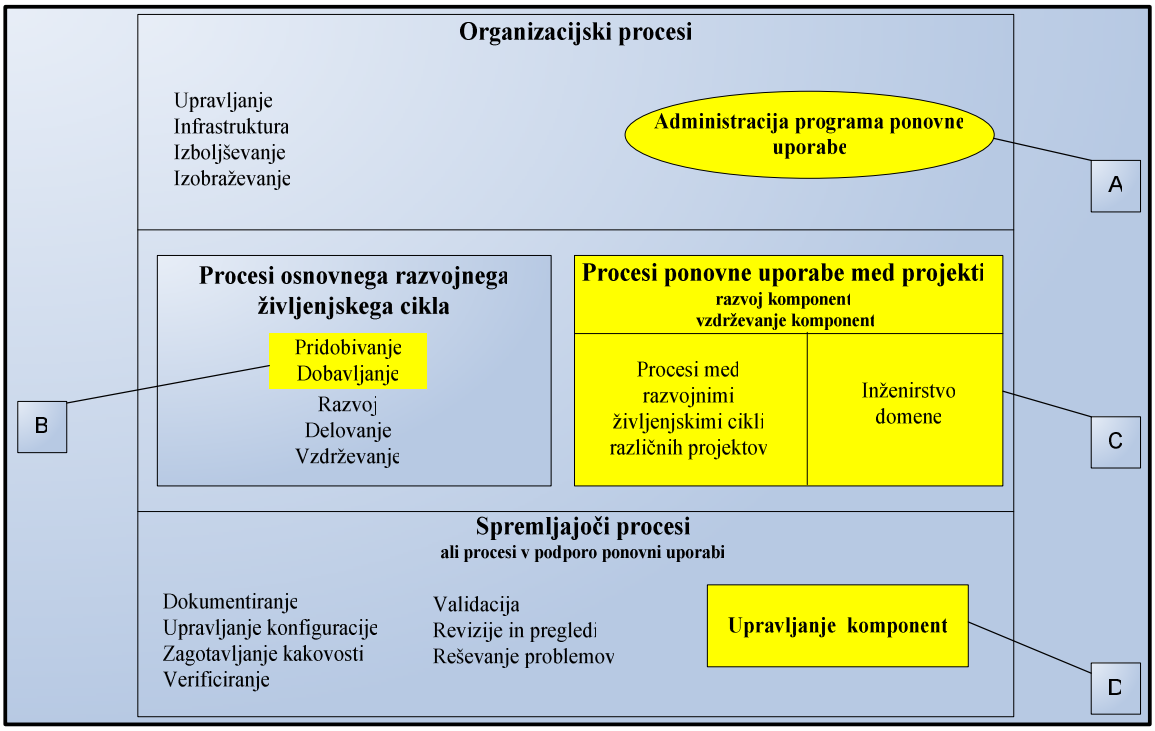
Zaporedje, po katerem se aktivnosti izvajajo, je prikazano na sliki 5. Na temelju identificiranega zaporedja aktivnosti ter ponavljajočih se korakov znotraj posamezne aktivnosti, definiram ponovno uporabni vzorec ali predlogo za opis posamezne aktivnosti razvojnega procesa (slika 6).



Slika 6: Vzorec ali predloga za opis aktivnosti, specifičnih za ponovno uporabo.

Vse za ponovno uporabo specifične aktivnosti so torej dodane preostalim aktivnostim razvojnega življenjskega cikla. Glede na to, da za splošni razvojni življenjski cikel že obstaja standard (ISO/IEC 12207, 1995 in 2002), sem bila mnenja, da je smiselno ta standard uporabiti kot osnovo ogrodja. Slika 7 prikazuje združitev procesov, aktivnosti in opravil ponovne uporabe s tem standardnim razvojnim življenjskim ciklom.

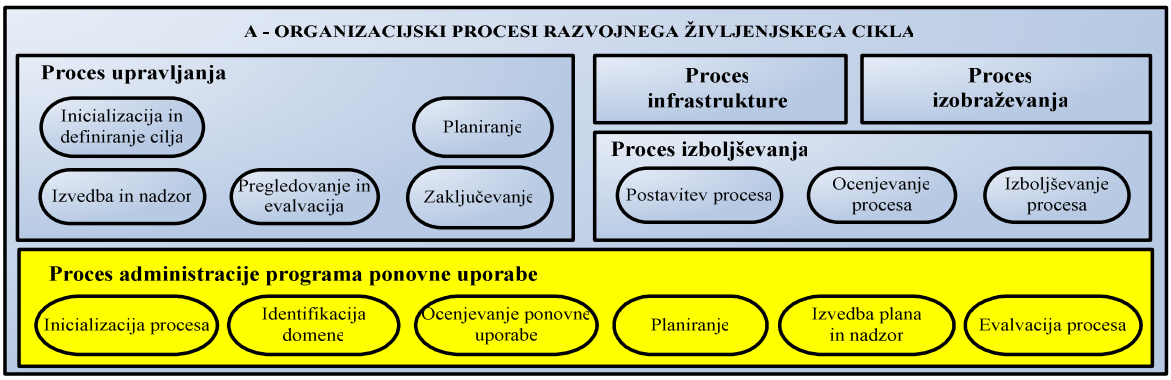
Procesi in aktivnosti, ki so specifični za ponovno uporabo, so dodani v likih z rumenim ozadjem. Opisi procesov, aktivnosti in opravil osnovnega življenjskega cikla so zaradi splošnosti izpuščeni, opisane pa so samo za ponovno uporabo specifične aktivnosti in opravila, ki se dodajo posameznim fazam osnovnega razvojnega življenjskega cikla. Pri tem zaporedje opisa ne predpisuje nujno tudi zaporedja izvajanja teh aktivnosti in opravil. Namen opisa je samo podati seznam procesov, aktivnosti in opravil, ki jih je nujno treba izvesti znotraj posamezne faze razvojnega cikla, če naj razvojni cikel prilagodimo sistematični ponovni uporabi.



Slika 7: Integracija procesov, aktivnosti in opravil ponovne uporabe z osnovnim razvojnim življenjskim ciklom.

4.2.1.1 A – Organizacijski procesi razvojnega življenjskega cikla

Med aktivnosti in opravila standardnega razvojnega procesa, ki so prikazani na sliki 7, je treba dodati tudi tista, ki se nanašajo na postavitev, upravljanje in podporo izvajanju sistematične ponovne uporabe v organizaciji. Osnovna za ponovno uporabo specifična organizacijska aktivnost je vzpostavitev administracije programa ponovne uporabe.



Slika 8: Organizacijski procesi razvojnega življenjskega cikla po [145,146] z dodanimi aktivnostmi in opravili procesa administracije programa ponovne uporabe.

A.1 Proces administracije programa ponovne uporabe

Če želimo z implementacijo sistematične ponovne uporabe na organizacijskem nivoju uspeti, jo je treba skrbno planirati in ustrezno voditi. Ker so poslovni, upravljavski in problemi človeške narave navadno veliko teže rešljivi kakor problemi tehnične narave, povezani z implementacijo ponovne uporabe, je treba v programu ponovne uporabe prav posebno skrb nameniti ravno vodenju programa, uresničevanju v programu zastavljenega, podpori programa z vsemi potrebnimi aktivnostmi ter vzgoji miselnosti sodelujočih v procesu, ki mora biti pozitivno naravnana do ponovne uporabe. Za uspešnost implementacije programa ponovne uporabe je zelo pomembno tudi tesno medsebojno sodelovanje vseh udeležencev ter izmenjevanje znanja, izkušenj in ponovno uporabnih komponent.

Administrator programa ponovne uporabe je npr. tista oseba, ki je zadolžena za postavitev in izvedbo plana ponovne uporabe. Proces administracije programa ponovne uporabe je namenjen postavitvi plana ponovne uporabe, njegovemu upravljanju in nadzoru ter spremljanje izvajanje programa znotraj organizacije. Posamezne aktivnosti so prikazane na sliki 8 v rumenem pravokotniku.

Aktivnost (A.1.1) Inicializacija procesa administracije

Na temelju strategije ponovne uporabe organizacije, v kateri so opredeljeni cilji ponovne uporabe, nameni, naloge in obseg ponovne uporabe, se izdela program ponovne uporabe, ki naj bi vključeval [10, 46]:

- postavitev in vzdrževanje infrastrukture ponovne uporabe (strojna in programska oprema, orodja, tehnike, standardi, metrike in pripomočki za izvajanje ponovne uporabe);
- navedbo sodelujočih v programu ponovne uporabe ter njihove vloge;
- investicije v ponovno uporabo in druge vire;
- funkcije v podporo programu ponovne uporabe: sodelovanje pri ustvarjanju in implementaciji izvedbenega plana programa ponovne uporabe; identificiranje udeležencev programa ponovne uporabe, dokumentiranje njihovih aktivnosti in seznanjanje udeležencev programa ponovne uporabe s strategijo ponovne uporabe; pospeševanje ponovne uporabe v praksi in s tem spodbujanje pozitivne miselnosti o ponovni uporabi; iskanje vseh možnosti za izvedbo ponovne uporabe v tekočih in

prihodnjih projektih; zagotavljanje svetovanja o ponovni uporabi za vse projekte, znotraj katerih se ponovna uporaba izvaja.

mehanizmi komuniciranja, povratnega informiranja in obveščanja: mehanizem povratnega informiranja, s katerim razvijalci posameznih programskih izdelkov inženirjem domene in upravitelju komponent posredujejo informacije o uporabi in učinkih uporabe posameznih programskih izdelkov ter komponent znotraj projekta; mehanizem komuniciranja med razvijalci, operaterjem, vzdrževalcem, inženirjem domene, upraviteljem komponent in administratorjem programa ponovne uporabe pri odpravljanju problemov, odgovarjanju na vprašanja in priporočanju ter svetovanju, ki se nanašajo na programske izdelke in komponente, uporabljene znotraj projekta; mehanizme obveščanja, ki razvijalcu, upravitelju komponent in inženirju domene omogočajo, da se sprotno obveščajo o veljavnih trgovinskih in drugih zakonodajah, licenčnih lastnostih programskih izdelkov in komponent, omejitvah organizacije, ki ščitijo njene lastniške interese ter o pogodbah, ki lahko omejujejo ali izključujejo uporabo specifičnih programskih izdelkov ali komponent znotraj nekega programskega projekta, v procesu vzdrževanja ali znotraj procesa inženirstva domene; mehanizem, ki inženirju domene omogoča pridobitev informacij in sodelovanje z ustreznimi viri, ki jih potrebuje za dokončanje svojih inženirskih aktivnosti.

- odbor za vodenje in nadzor nad izvajanjem programa ponovne uporabe (sodelujoči v odboru naj bi bili sponzor ponovne uporabe, vodja programskega razvoja, vodja izvajanja, vodja vzdrževanja ter izvedenec za ponovno uporabo), katerega naloga je preverjanje izvajanja ponovne uporabe v organizaciji, identificiranje področij, na katerih so možnosti ponovne uporabe še posebno velike, določanje odgovornosti posameznim sodelujočim v izvajanju programa ponovne uporabe, spodbujanje ponovne uporabe znotraj organizacije.

Aktivnost (A.1.2) Identifikacija domene

Domeno označuje množica sistemov ali aplikacij, ki imajo določene skupne lastnosti in jih je zato mogoče organizirati kot zbirko ponovno uporabnih komponent.

Administrator ponovne uporabe mora identificirati posamezne domene, znotraj katerih bo organizacija pri razvoju novih sistemov izvajala ponovno uporabo ter ugotovitve ustrezno dokumentirati. Domena mora resnično sovpadati s strategijo ponovne uporabe, ki jo je

organizacija sprejela in postavila. Pri izbiri in ocenjevanju domen administratorju pomagajo tudi drugi udeleženci, to so upravitelji knjižnic, inženirji domene, uporabniki in razvijalci aplikacij. Administrator mora izvajati preglede domen periodično. Na podlagi pridobljenih informacij o domenah ter morebiti rezultatih analiz domen lahko množico ciljnih domen po potrebi skrči.

Aktivnost (A.1.3) Ocenjevanje ponovne uporabe

Ocenjevanje ponovne uporabe predstavlja osnovo za merjenje izvajanja ponovne uporabe v organizaciji. Brez tovrstnega ocenjevanja je namreč zelo težko ugotoviti, katere prednosti je ponovna uporaba organizaciji sploh prinesla. Na temelju ocen ponovne uporabe je možno pridobiti oceno zrelosti ponovne uporabe v organizaciji, oceniti možnosti ponovne uporabe znotraj ciljnih domen organizacije, izdelati predloge, kako izboljšati prakso ponovne uporabe v organizaciji (kako izboljšati strategijo ponovne uporabe in izvedbeni plan programa ponovne uporabe), izvesti izboljšave na čim več področjih organizacije, posebno na področju šolanja in infrastrukture (izboljševanje veščin, tehnologij, procesov ponovne uporabe, organizacijskih struktur in metrik). Rezultate ocen mora administrator dokumentirati in posredovati članom odbora za vodenje in nadzor izvajanja programa ponovne uporabe.

Aktivnost (A.1.4) Planiranje

Pri izdelavi, dokumentiranju in vzdrževanju izvedbenega plana programa ponovne uporabe je treba, če seveda obstaja, ponovno uporabiti predlogo plana, v planu pa določiti:

- aktivnosti programa ponovne uporabe,
- postopke in časovni razpored izvedb teh aktivnosti,
- odgovorne za izvedbo posameznih aktivnosti,
- medsebojne odnose med odgovornimi za izvedbo aktivnosti in preostalimi (razvijalci ali inženirji domene),
- vire (resurse), potrebne za izvedbo programa.

Plan je treba uskladiti s strategijo ponovne uporabe organizacije ter sposobnostjo organizacije za njegovo izvedbo. Odbor za vodenje in nadzor nad izvajanjem plana mora plan pred pričetkom njegovega izvajanja potrditi. Plan oziroma njegovo izvajanje se periodično pregledujeta. Po potrebi (glede na ocene ponovne uporabe) se plan sproti spreminja.

Aktivnost (A.1.5) Izvedba plana in nadzor

Vse aktivnosti ponovne uporabe se morajo izvajati po izvedbenem planu programa. Izvajanje plana je treba spremljati in stalno primerjati s strategijo ponovne uporabe. Po potrebi se plan spremeni oziroma prilagodi, tako da bo postavljeno strategijo možno izvesti. Vse težave, ki nastajajo pri izvajanju plana, ali morebitna nesoglasja s strategijo, zaradi katerih je prišlo do sprememb plana, je treba ustrezno dokumentirati. Administrator programa pa mora periodično tudi pridobiti podaljšanje sponzorstva, podpore in naklonjenosti programu ponovne uporabe.

Aktivnost (A.1.6) Evalvacija procesa

Administrator mora periodično oceniti program ponovne uporabe in s tem ugotoviti doseganje zastavljene strategije ter primernost in učinkovitost samega programa. Rezultate ocenjevanja mora posredovati članom odbora za vodenje in nadzor izvajanja programa, katerim predlaga tudi morebitne spremembe programa, njegovo razširitev ali izboljšave.

4.2.1.2 B – Osnovni procesi razvojnega življenjskega cikla na temelju ponovne uporabe

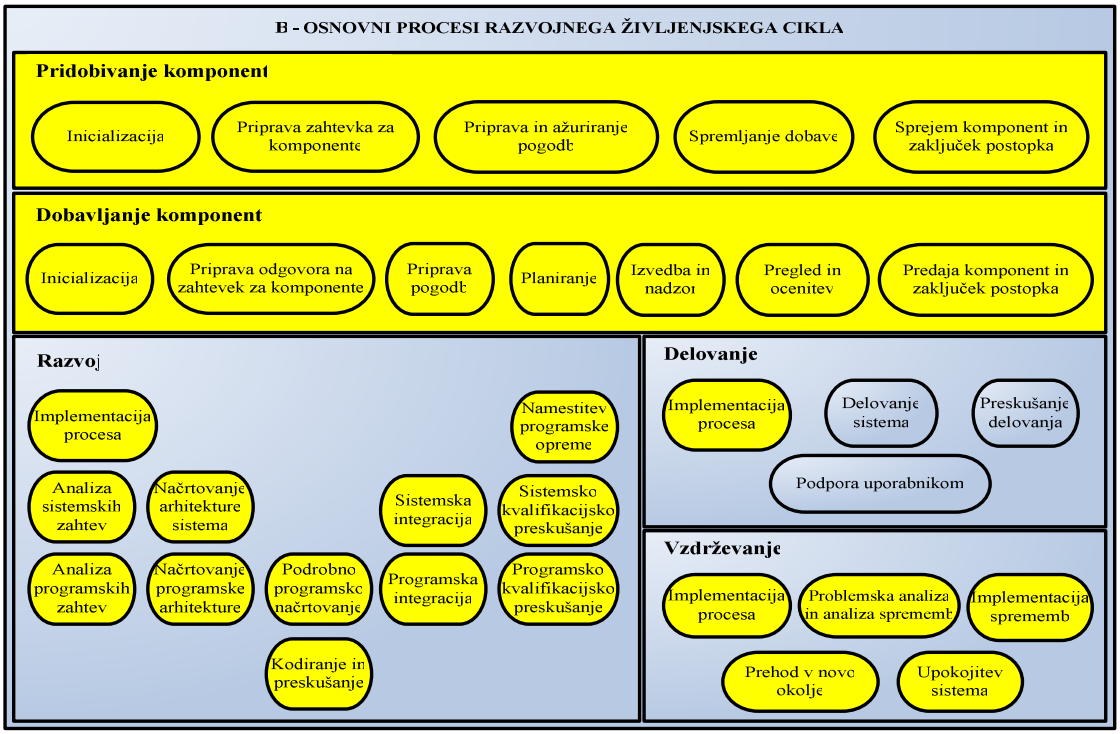
Osnovni procesi razvojnega življenjskega cikla po [41,42] so prikazani na sliki 9 v modrih pravokotnikih. Specifični procesi, aktivnosti in opravila za ponovno uporabo, s katerimi je treba nadgraditi te aktivnosti za dosego razvojnega življenjskega cikla na temelju ponovne uporabe in ki so v nadaljevanju tudi podrobneje opisani, pa so predstavljeni v rumenih pravokotnikih.

B.1 Proces pridobivanja komponent

V procesu pridobivanja komponent so združene aktivnosti in opravila, ki jih mora izvesti pridobitelj komponente, pri tem pa je komponenta lahko nek programski izdelek, ki ga že lahko obravnavamo kot celoto, ali pa je to samo sestavni del programskega izdelka. Proces se prične z oblikovanjem potrebe po komponenti oziroma z izdelavo zahtevka za komponente (ang. *RFP – Request For Proposal*). Med prijavljenimi dobavitelji se nato izbere najprimernejši, spelje in vodi se postopek pridobitve komponente. V vlogi dobaviteljev so

lahko posamezni razvijalci ponovno uporabnih komponent ali knjižnica ponovno uporabnih komponent.

Proces pridobivanja komponent bi lahko razdružili v naslednje aktivnosti: inicializacija postopka pridobivanja komponent, priprava zahtevka za komponente, priprava in ažuriranje pogodb, spremljanje dobave ter sprejem komponent, s čimer se postopek pridobivanja zaključi. Za ponovno uporabo so še posebno pomembne aktivnosti inicializacije postopka pridobivanja, izdelave zahtevka za komponente ter zaključni postopki.



Slika 9: Osnovni procesi razvojnega življenjskega cikla [povzeto po [144]] z dodanimi aktivnostmi pridobivanja in oskrbe s komponentami.

Aktivnost*: *(B.1.1) Inicializacija postopka pridobivanja komponent

Izvajalec aktivnosti: pridobitelj komponent.

Opis in rezultati

Opis aktivnosti: pregleda vse komponente, ki so že na razpolago ali so v fazi pridobivanja, da ne pride do nepotrebnega podvajanja komponent; opredeli, katere komponente potrebuje pred pričetkom razvoja nekega izdelka; pri odločanju o tem, katere komponente je treba pridobiti,

katere pa razviti samostojno oziroma na novo, sodeluje z dobavitelji komponent, inženirji domene in/ali upravitelji komponent;

Opis rezultatov: Pridobitelj komponent mora izdelati plan nabave komponent, v katerem opredeli izvore komponent ter postopke njihove nabave.

Ponovno uporabljene komponente: v preteklosti izdelani plani.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: vsaka posamezna komponenta.

Kriteriji: ponovna uporabnost izbranih komponent; združljivost vmesnika komponente z arhitekturo domene in modelom domene; sovpadanje programskih zahtev s funkcionalnostjo posamezne komponente; ocena stroškov in pridobitev, ki bi jih dosegli s prilagoditvijo programskih zahtev izbrani komponenti[107]; ustreznost komponente s tržišča programskim zahtevam, standardom in kriterijem ponovne uporabe organizacije.

Posredovanje ponovno uporabne informacije

Izvor: pridobitelj komponent oziroma izvajalec ocenjevanja.

Ponor: razvijalec aplikacij, upravitelj komponent.

Vsebina: dokumentirati ocenjene stroške in pridobitve s ponovno uporabo posamezne komponente.

Pravne aktivnosti: pred dokončnim izborom posamezne komponente pregledati "pravno zgodovino"[108] komponente in se prepričati v njeno pravno neoporečnost.

Aktivnost*: *(B.1.2) Priprava zahtevka za komponente

Izvajalec aktivnosti: pridobitelj komponent.

Opis in rezultati

Opis aktivnosti: sestavi dokument, v katerem so opisane vse zahteve[109], ki se nanašajo na ponovno uporabo komponent in jih mora dobavitelj komponent obvezno izpolniti.

Opis rezultatov: zahtevek za komponente.

---

[107] S takim prilagajanjem zahtev komponenti je namreč mogoče ponovno uporabo neke komponente bistveno povečati ali jo celo sploh omogočiti.

[108] Lastništvo nad komponento, omejitve pri uporabi, pogoji za pridobitev licence.

[109] Osnovni elementi zahtevka so opis sistemskih ali aplikacijskih zahtev, cilji, ki jih želimo doseči s komponentami, navodila ponudnikom, kako naj oblikujejo svoje ponudbe in kaj morajo ponudbe vsebovati, seznam programskih izdelkov ali komponent, določbe in pogoji, pod katerimi se bodo komponente sprejele, nadzor nad poddobavitelji, tehnične omejitve, ki se nanašajo na ciljno okolje komponente. [144, str. 11] ed najpomembnejše zahteve pa gotovo sodi zahteva po ustrezni spremljajoči dokumentaciji za posamezno komponento.

Ponovno uporabljene komponente: predloga zahtevka za komponente, izdelana v preteklosti.

Ocena po kriterijih ponovne uporabnosti -

Posredovanje ponovno uporabne informacije -

Pravne aktivnosti: oblikovanje pogodbenih določil in pogojev za prevzem oziroma ponovno uporabo komponent.

Aktivnost: ***(B.1.3) Priprava in ažuriranje pogodb***

Izvajalec aktivnosti: pridobitelj komponent.

Opis in rezultati (glej pravne aktivnosti)

Ocena po kriterijih ponovne uporabnosti -

***Posredovanje ponovno uporabne informacije*:** povratne informacije razvijalcem komponent in/ali upravitelju komponent.

Pravne aktivnosti: pridobitev dovoljenj – licenc za uporabo komponente in definiranje pogojev uporabe ter izdelava seznama potrebnih pravic na posameznih komponentah.

Aktivnost: ***(B.1.4) Sprejem komponent in zaključek postopka pridobivanja***

Izvajalec aktivnosti: pridobitelj komponent.

Opis in rezultati

Opis aktivnosti: pregleda ustreznost posameznim ponudb in izbere najprimernejšo.

*Opis rezultatov:*seznam izbranih ponovno uporabljenih komponent ter njihovih virov.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: vsaka pridobljena komponenta.

Kriteriji: preskus sprejemljivosti komponente ter njene združljivosti z arhitekturo domene.

***Posredovanje ponovno uporabne informacije*:** povratne informacije razvijalcem komponent in/ali upravitelju komponent.

Pravne aktivnosti: razširitev seznama izbranih ponovno uporabnih komponent z opisom morebitnih omejitev na posameznih komponentah in dovoljenega obsega prilagajanja ali spreminjanja komponent.

B.2 Proces dobavljanja komponent

Izvedba opravil in aktivnosti procesa dobavljanja komponent je v pristojnosti dobavitelja komponent, ki mora pripraviti predlog in/ali pogodbo o dobavi komponent glede na zahteve, ki jih pridobitelj komponent postavi v zahtevku. Dobavitelj mora določiti tudi potrebne postopke in vire za razvoj in dobavo zahtevanih komponent.

Ponovna uporaba bo dosegla neko želeno stopnjo, če bodo na razpolago komponente, ki jih razvijalci domene potrebujejo oziroma po njih povprašujejo. Zato je planiranje komponent, ki jih bodo razvijalci aplikacij neke domene potrebovali, ena najbistvenejših aktivnosti ponovne uporabe, ki jo je treba izvesti v procesu dobave komponent.

Aktivnost*: *(B.2.1) Planiranje

Izvajalec aktivnosti: dobavitelj komponent.

Opis in rezultati

Opis aktivnosti: Skupaj s pridobiteljem komponent ocenita možnosti za spremembe programskih zahtev oziroma njihovo prilagoditev komponentam, kar omogoča ali poveča ponovno uporabnost izbranih komponent. Dobavitelj zbere kandidatne komponente oziroma izdelke, lahko tudi iz različnih virov[110].

Opis rezultatov: projektni plan, v katerem so zbrane ocene tveganj in pridobitev, ki bi nastale kot posledica prilagoditve programskih zahtev za aplikacijo izbranim komponentam, opis strategije razvoja programskega izdelka ali aplikacije iz komponent, opis strategije razvoja sestavnih delov aplikacije kot ponovno uporabnih komponent ter opis ali navedba kandidatnih ponovno uporabnih komponent, ki jih bo mogoče razviti na novo v času trajanja projekta.

Ponovno uporabljene komponente: predloge planov in opisov.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: (prilagojene) programske zahteve.

Kriteriji: ocena tveganj ter pridobitev kot posledic uresničitve v zahtevku postavljenih zahtev za komponente.

Posredovanje ponovno uporabne informacije

Izvor: dobavitelj komponent.

[110] Viri komponent so lahko: rezultati inženirstva domene, notranji ali zunanji dobavitelji, domače knjižnice ponovno uporabnih komponent, drugi notranji projekti v teku, vnaprej pripravljeni programski izdelki in komponente (komponentna tržišča), zunanji dobavitelji ali poddobavitelji.

Ponor: razvijalec aplikacije.

Vsebina: ocena tveganj in pridobitev, ki jih dosežemo ob spreminjanju programskih zahtev, da jih prilagodimo komponentam, ki so na voljo za ponovno uporabo.

***Pravne aktivnosti*:** preveriti je treba omejitve spreminjanja in prilagajanja posameznih komponent.

B.3 Razvojni proces

Razvojni proces[111] vključuje aktivnosti in opravila razvijalca aplikacij ali sistemov, ki se nanašajo na analizo zahtev, načrtovanje, kodiranje, integracijo, preskušanje ter namestitev aplikacij, razvitih iz komponent. Komponente so lahko pridobljene iz kateregakoli vira, notranjega ali zunanjega, vključevati pa morajo arhitekturo domene, v katero sodi programski izdelek v razvoju, ter komponente, zgrajene za ponovno uporabo v tej domeni.

Aktivnost*: *(B.3.1) Implementacija razvojnega procesa

Izvajalec aktivnosti: razvijalec aplikacije.

Opis in rezultati

Opis aktivnosti: na temelju velikosti in kompleksnosti razvojnega projekta ter postavljenih ciljev izbere ustrezen model razvojnega življenjskega cikla; nadgradi ga z aktivnostmi in opravili ponovne uporabe, obstoječim zahtevam doda še za ponovno uporabo specifične zahteve; glede na zahteve ponovne uporabe izbere standarde, metode, orodja in programski jezik, ki bodo podpirali, omogočali in spodbujali izvedbo aktivnosti ponovne uporabe; izdela projektni plan razvoja; vzpostavi dobre komunikacijsko-informacijske mehanizme[112].

Opis rezultatov: projektni plan razvoja (vsebuje tudi vse podatke o standardih, metodah, orodjih in programskem jeziku, ki bodo uporabljeni pri izvedbi projekta).

Ponovno uporabljene komponente: obstoječi projektni plan ali njegova predloga.

[111] Pogosto mu rečemo kar aplikacijsko inženirstvo ali *razvoj s ponovno uporabo.*

[112] Med pomembne informacijsko komunikacijske mehanizme sodijo: mehanizem povratnega informiranja, s katerim lahko razvijalec aplikacij inženirju domene poroča o učinkih ponovne uporabe neke komponente znotraj projekta, komunikacijo med razvijalcem in upraviteljem komponent, saj mora slednji rešiti vse morebitne probleme pri ponovni uporabi komponente, razvijalcu aplikacij pa mora ponujati tudi vsa potrebna pojasnila v zvezi s ponovno uporabljeno komponento ter mu po potrebi svetovati pri ponovni uporabi, ter komunikacijo med razvijalcem komponent in razvijalcem aplikacije, ki mora biti obveščen o morebitnih spremembah komponent, problemih pri njihovi ponovni uporabi in o vsem drugem, kar je kakorkoli povezano s ponovno uporabljenimi komponentami znotraj projekta.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: model razvojnega življenjskega cikla in orodja.

Kriteriji: ustreznost modela življenjskega cikla in izbranih orodij za izvedbo razvoja na temelju ponovne uporabe.

Pravne aktivnosti: razvijalec mora biti stalno na tekočem z obstoječo predmetno zakonodajo, licenčnimi lastnostmi programskih izdelkov in komponent, omejitvami, ki ščitijo lastniške interese (pravice) organizacije ter s pogodbami, ki omejujejo ali izključujejo uporabo specifičnih programskih izdelkov ali komponent v danem projektu.

Aktivnost*: *(B.3.2) Analiza sistemskih zahtev

Izvajalec aktivnosti: razvijalec aplikacije.

Opis in rezultati

Opis aktivnosti: identificira domene, iz katerih lahko pridobi komponente; izbere ustrezne modele domene; izdela specifikacijo zahtev za razvijajočo aplikacijo.

Opis rezultatov: specifikacija sistemskih zahtev (pri njenem oblikovanju mora uporabljati jezik in koncepte predhodno izbranega modela domene; poleg drugega[113] mora vsebovati tudi opis zahtev po ponovni uporabnosti aplikacije ter zahteve za vmesnik arhitekture domene).

Ponovno uporabljene komponente: modeli domene, obstoječe specifikacije.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: sistemske zahteve.

Kriteriji: uporabnost in ponovno uporabnost sistemskih zahtev, identifikacija prilagojenih zahtev ponovno uporabljenim komponentam ter možnosti ponovne uporabe zahtev v različnih kontekstih.

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije.

Ponor: upravitelj komponent in inženir domene.

Vsebina: informacije o ponovni uporabnosti zahtev, ki morajo vključevati: podatke o manjkajočih komponentah v domeni, seznam nominalno (potencialno) ponovno uporabnih komponent znotraj projekta, ki pa so bile zavrnjene ter razloge za njihovo zavrnitev oziroma ne-ponovno uporabo, predlagane spremembe (modifikacije) komponent, seznam ponovno

[113] Specifikacija sistemskih zahtev vključuje opis funkcij sistema, njegovih zmožnosti, poslovnih, organizacijskih in uporabniških zahtev, varnostnih zahtev, ergonomskih zahtev, zahtev po vzdrževanju, načrtovalskih omejitev ter kvalifikacijskih zahtev [144].

uporabljenih komponent v projektu, seznam novih kandidatnih komponent kot predlog tekočega projekta.

Pravne aktivnosti: ocena ponovne uporabnosti specifikacije sistemskih zahtev z vidika pravnih omejitev.

Aktivnost*: *(B.3.3) Načrtovanje arhitekture sistema

Izvajalec aktivnosti: razvijalec aplikacije.

Opis in rezultati

Opis aktivnosti: zgraditi ali ponovno uporabiti arhitekturo domene, ki ustreza izbranemu modelu domene ter iz nje izpeljati arhitekturo sistema na temelju sistemskih ali aplikacijskih zahtev.

Opis rezultatov: arhitektura sistema, ki je usklajena s sistemskimi zahtevami.

Ponovno uporabljene komponente: arhitektura domene.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: arhitektura sistema.

Kriteriji: uporabnost in ponovna uporabnost sistemske arhitekture in njenih delov; možnost ponovne uporabe sistemske arhitekture ali njenih delov v različnih kontekstih; združljivost sistemske arhitekture z modeli domene in domenskimi arhitekturami.

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije.

Ponor: upravitelj komponent in inženir domene.

Vsebina: morebitne težave pri ponovni uporabi arhitekture sistema.

Pravne aktivnosti: ocena ponovne uporabnosti arhitekture sistema z vidika pravnih omejitev.

Aktivnost*: *(B.3.4) Analiza programskih zahtev

Izvajalec aktivnosti: razvijalec aplikacije.

Opis in rezultati

Opis aktivnosti: za vsako programsko komponento določi in ustrezno dokumentira programske zahteve.

Opis rezultatov: specifikacija programskih zahtev, skupaj s specifikacijo kakovosti za posamezno komponento[114].

[114] Vključno s programskimi zahtevami komponent, ki naj bi jih ponovno uporabili.

Ponovno uporabljene komponente: obstoječe programske zahteve; predloga za pisanje novih programskih zahtev, ki upošteva jezik in koncepte prehodno izbranega modela domene.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: komponenta (njena kakovost).

Kriteriji: osnova za oceno so izkušnje z zanesljivostjo komponente, ko je bila uporabljena v drugih projektih, rezultati pregledovanja razpoložljivih komponent in ugotavljanja njihovih morebitnih defektov ter rezultati preskušanj ustreznosti komponent programskim in sistemskim zahtevam.

Izdelek, ki se ocenjuje: programske zahteve.

Kriteriji: preverjanje združljivosti zahtev z modeli domene in arhitekturo domene; skladnost s standardi ponovne uporabe organizacije; uporabnost in ponovna uporabnost programskih zahtev; identifikacija prilagojenih programskih zahtev, ki jih je mogoče zamenjati s ponovno uporabnimi zahtevami, če pride do prilagoditve zahtevam komponenti.

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije.

Ponor: inženir domene in upravitelj komponent.

Vsebina: ocena ponovne uporabnosti programskih zahtev.

Pravne aktivnosti: ocena ponovne uporabnosti programskih zahtev z vidika pravnih omejitev.

Aktivnost: ***(B.3.5) Načrtovanje programske arhitekture***

Izvajalec aktivnosti: razvijalec aplikacij.

Opis in rezultati

Opis aktivnosti: programske zahteve preoblikuje v ustrezno arhitekturo; dokumentira arhitekturo; izdela načrt podatkovne baze na temelju modela domene; izdela uporabniško dokumentacijo in preskusi zahteve.

Opis rezultatov: programska arhitektura, usklajena s programskimi zahtevami[115].

Ponovno uporabljene komponente: obstoječe programske arhitekture in dokumentacijske komponente ter načrti podatkovnih baz, posamezni elementi za preskušanje zahtev.

Ocena po kriterijih ponovne uporabnosti

[115] Izpeljana je bodisi iz obstoječe arhitekture bodisi je zgrajena na novo na temelju izbranega modela domene, ki opisuje strukturo aplikacije in komponent, ki jo sestavljajo. Zunanji vmesniki (za povezovanje posameznih modulov med seboj) in notranji vmesniki (med posameznimi komponentami, ki sestavljajo nek programski modul), morajo biti usklajeni z vmesniki arhitekture domene.

Izdelek, ki se ocenjuje: programska arhitektura, vmesniki in načrt podatkovne baze.

Kriteriji: sovpadanje programske arhitekture, vmesnikov in načrta podatkovne baze s standardi ponovne uporabe organizacije, uporabnost in ponovna uporabnost programske arhitekture in načrta podatkovne baze ter zmožnost uporabe programske arhitekture in njenih komponent v različnih okoljih.

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije.

Ponor: inženir domene in upravitelj komponent.

Vsebina: vse potrebne informacije o načrtu programske arhitekture: probleme z arhitekturo domene ali modeli domene, s katerimi se je razvijalec srečeval med njihovo ponovno uporabo, razloge za ne-uporabo neke domenske arhitekture ali modela, predlagane spremembe arhitekture ali modelov, seznam v projektu uporabljenih arhitektur in modelov ter seznam novih kandidatov za ponovno uporabne komponente, ki so nastale znotraj projekta.

Pravne aktivnosti: preverjanje in upoštevanje morebitnih omejitev pri ponovni uporabi posameznih komponent.

Aktivnost*: *(B.3.6) Podrobno programsko načrtovanje

Izvajalec aktivnosti: razvijalec aplikacije.

Opis in rezultati *Opis aktivnosti:* izdelati podrobni načrt za vsako programsko komponento, vmesnike med komponentami, ki bodo ponovno uporabljene, natančni načrt in dokumentacijo za podatkovno bazo.

Opis rezultatov: podrobni načrt za vsako komponento aplikacije, ki sloni na jeziku in konceptih modelov domene, vmesniki za ponovno uporabljene komponente, podrobni načrt podatkovne baze, pri opisu katerega uporabimo podatkovne strukture in poimenovanja konceptov kot sledi iz modelov domene.

Ponovno uporabljene komponente: najbolj primeren obstoječi načrt komponente ali podatkovne baze, dokumentacijske in preskusne komponente.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: podrobni načrti.

Kriteriji: konsistentnost podrobnih načrtov z načrtom programske arhitekture, arhitekturo domene in modeli domene; sovpadanje podrobnega načrta s standardi ponovne uporabe organizacije; uporabnost ter ponovno uporabnost načrtov in preskusnih zahtev; identifikacija

prilagojenih načrtov in preskusnih zahtev, ki jih je mogoče nadomestiti s ponovno uporabnimi načrti in zahtevami, v kolikor pride do sprememb začetnih programskih zahtev; ponovna uporabnost podrobnih načrtov posameznih komponent ter preskusov zahtev v različnih kontekstih.

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije.

Ponor: inženir domene in upravitelj komponent.

Vsebina: informacije o ponovni uporabi načrtov ter ponovno uporabne informacije v zvezi z načrti.

Pravne aktivnosti: preverjanje in upoštevanje morebitnih omejitev pri ponovni uporabi posameznih komponent.

Aktivnost*: *(B.3.7) Kodiranje in preskušanje

Izvajalec aktivnosti: razvijalec aplikacije.

Opis in rezultati

Opis aktivnosti: izbere vse komponente in izdelke, ki jih lahko ponovno uporabi, druge razvije na novo, pri tem pa mora upoštevati jezik in koncepte modelov domene ter arhitekture domene; dokumentirati mora posamezne komponente in podatkovne baze.

Opis rezultatov: programska koda, preskusne komponente.

Ponovno uporabljene komponente: programske komponente, podatkovne baze, preskusni postopki in podatki, dokumentacijske komponente.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: koda in preskusne komponente.

Kriteriji: sovpadanje kode in preskusov s standardi ponovne uporabe organizacije, uporabnost in ponovno uporabnost kode in preskusnih komponent ter ponovno uporabnost kodnih in preskusnih komponent v različnih okoljih.

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije.

Ponor: inženir domene in upravitelj komponent.

Vsebina: ponovno uporabne informacije o novih kodnih in preskusnih komponentah ter podatkih.

Pravne aktivnosti: določitev omejitev uporabe za novo razvite ponovno uporabne komponente in sklenitev ustreznih pogodb z upraviteljem komponent; preverjanje in upoštevanje morebitnih omejitev pri ponovni uporabi posameznih komponent.

Aktivnost*: *(B.3.8) Programska integracija

Izvajalec aktivnosti: razvijalec aplikacije.

Opis in rezultati

Opis aktivnosti: razvije (ga ponovno uporabi ali izdela po predlogi) in dokumentira plan integracije; razvije kvalifikacijske preskuse, preskusne primere in postopke.

Opis rezultatov: plan integracije[116].

Ponovno uporabljene komponente: obstoječi plan integracije ali predloga za njegovo izdelavo; obstoječe dokumentacijske komponente; kvalifikacijski preskusi, preskusni primeri in postopki.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: integracijski plan, programski načrt, koda, preskusi in uporabniške informacije.

Kriteriji: sovpadanje vseh izdelkov, ki so rezultat aktivnosti, s standardi ponovne uporabe organizacije; uporabnost in ponovna uporabnost programskih izdelkov, preskusnih primerov in postopkov; ponovna uporabnost vseh izdelkov v različnih kontekstih.

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije.

Ponor: inženir domene in upravitelj komponent.

Vsebina: ponovno uporabne informacije o postopku integracije.

Pravne aktivnosti preverjanje in upoštevanje morebitnih omejitev pri ponovni uporabi posameznih komponent.

Aktivnost*: *(B.3.9) Programsko kvalifikacijsko preskušanje

Izvajalec aktivnosti: razvijalec aplikacije.

Opis in rezultati

Opis aktivnosti: preverja doseganje kvalifikacijskih zahtev za posamezno programsko komponento ter rezultate ustrezno dokumentira.

[116] V njem je navedeno, kako bodo posamezne programske komponente, definirane v fazi podrobnega načrtovanja, povezane v celoto, določene pa so tudi preskusne zahteve, postopki in podatki, odgovornosti posameznikov ter časovni plan izvajanja aktivnosti.

Opis rezultatov: kvalifikacijski preskusi.

Ponovno uporabljene komponente: dokumentacijske komponente.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: kvalifikacijske zahteve.

Kriteriji: preveriti sovpadanje zahtev s standardi ponovne uporabe organizacije, njihovo uporabnost ter ponovno uporabnost znotraj projekta ter v različnih okoljih.

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije.

Ponor: inženir domene in upravitelj komponent.

Vsebina: ponovno uporabne informacije, ki se nanašajo na kvalifikacijsko preskušanje.

Pravne aktivnosti: preverjanje in upoštevanje morebitnih omejitev pri ponovni uporabi posameznih komponent.

Aktivnost*: *(B.3.10) Sistemska integracija

Izvajalec aktivnosti: razvijalec aplikacije.

Opis in rezultati

Opis aktivnosti: izvede sistemsko integracijo, ki se nanaša na združitev programskih in strojnih elementov aplikacije ali sistema. Za vsako kvalifikacijsko sistemsko zahtevo mora razviti množico preskusov, preskusnih primerov in postopkov.

Opis rezultatov: integracija ali agregat, ki predstavlja aplikacijo.

Ponovno uporabljene komponente: ponovno uporabni preskusi, preskusni primeri in podatki za preskušanje kvalifikacijskih zahtev.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: integriran sistem ali aplikacija.

Kriteriji: sovpadanje s standardi ponovne uporabe organizacije, uporabnost in ponovna uporabnost aplikacije in kvalifikacijskih preskusov ter ponovna uporabnost aplikacije, njenih delov in kvalifikacijskih preskusov v različnih kontekstih.

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije.

Ponor: inženir domene in upravitelj komponent.

Vsebina: ponovno uporabna informacija o celotnem sistemu ali aplikaciji, množici preskusov, preskusnih primerov ter postopkov.

Aktivnost*: *(B.3.11) Sistemsko kvalifikacijsko preskušanje

Izvajalec aktivnosti: razvijalec aplikacije.

Opis in rezultati

Opis aktivnosti: preverjanje uresničitve sistemskih ali aplikacijskih zahtev in pripravljenost sistema ali aplikacije na namestitev v ciljno delovno okolje.

Opis rezultatov: rezultati preskusov.

Ponovno uporabljene komponente: elementi za preskušanje.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: celoten sistem ali aplikacija.

Kriteriji: ocenimo uporabnost in ponovno uporabnost programskega dela sistema ali aplikacije ter ponovno uporabnost posameznih delov sistema ali aplikacije v različnih kontekstih (okoljih).

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije.

Ponor: inženir domene in upravitelj komponent.

Vsebina: ponovno uporabne informacije o sistemskem kvalifikacijskem preskušanju.

Pravne aktivnosti -

Aktivnost*: *(B.3.12) Namestitev programske opreme

Izvajalec aktivnosti: razvijalec aplikacije.

Opis in rezultati

Opis aktivnosti: namestitev programskega izdelka v ciljno okolje delovanje.

Opis rezultatov: namestitveni plan, v katerem so definirani in dokumentirani viri ter postopki namestitve.

Ponovno uporabljene komponente: predloga namestitvenega plana.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: plan namestitve.

Kriteriji: preverjanje uporabnosti in ponovno uporabnosti plana ter možnost ponovne uporabe namestitvenega plana v različnih kontekstih.

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije.

Ponor: inženir domene in upravitelj komponent.

Vsebina: ponovno uporabne informacije o namestitvi sistema ali aplikacije.

Pravne aktivnosti -

B.4 Proces delovanja

Za ponovno uporabo najpomembnejša aktivnost znotraj procesa delovanja je implementacija samega procesa.

Aktivnost*: *(B.4.1) Implementacija plana delovanja

Izvajalec aktivnosti: razvijalec aplikacije (operater).

Opis in rezultati

Opis aktivnosti: skrbi za nemoteno delovanje sistema ter zagotavljanje ustrezne podpore uporabnikom sistema, izdela plan delovanja oziroma ponovno uporabi predlogo za izdelavo plana delovanja, v katerem so definirani in opisani viri ter postopki delovanja sistema.

Opis rezultatov: plan delovanja.

Ponovno uporabljene komponente: predloga plana delovanja.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: plan delovanja.

Kriteriji: preverjanje uporabnosti in ponovne uporabnosti plana delovanja ter možnosti njegove ponovne uporabe v različnih kontekstih.

Posredovanje ponovno uporabne informacije

Izvor: razvijalec aplikacije (operater).

Ponor: inženir domene in upravitelj komponent.

Vsebina: ponovno uporabne informacije o delovanju programskega sistema.

Pravne aktivnosti -

B.5 Proces vzdrževanja

Proces vzdrževanja združuje aktivnosti in opravila, ki se izvajajo pri odpravljanju morebitnih pomanjkljivosti ali nepravilnosti v programskem izdelku oziroma pri prilagajanju izdelka novim ali spremenjenim zahtevam, pri čemer mora izdelek ohraniti svojo integriteto.

Namen procesa vzdrževanja je spreminjanje obstoječega izdelka, preselitev izdelka v drugo okolje ali umik (upokojitev) izdelka.

Aktivnost*: *(B.5.1) Implementacija procesa

Izvajalec aktivnosti: vzdrževalec sistema ali aplikacije.

Opis in rezultati

Opis aktivnosti: izdela plan vzdrževanja; določi postopke sprejemanja, zapisovanja, odpravljanja in spremljanja težav ter poskrbi, da se ustrezne povratne informacije pošljejo upravitelju komponent vsakič, ko se težave nanašajo na ponovno uporabljene komponente v sistemu ali aplikaciji ali se za te komponente spremenijo zahteve. Vse morebitne težave s ponovno uporabljenimi komponentami mora dokumentirati. [117]

Opis rezultatov: plan vzdrževanja.

Ponovno uporabljene komponente: predloga plana vzdrževanja, dokumentacijske komponente.

Posredovanje ponovno uporabne informacije

Izvor: vzdrževalec aplikacije ali sistema.

Ponor: razvijalec aplikacije, inženir domene, upravitelj komponent.

Vsebina: vse informacije o težavah in spremembah komponent, ki so bile ponovno uporabljene.

Pravne aktivnosti: proženje ustreznih pravnih mehanizmov v primeru, da se ne spoštujejo določbe o vzdrževanju ponovno uporabnih komponent oziroma da knjižnica razvijalca aplikacije ne obvešča o spremembah na ponovno uporabljeni komponenti ali morebitnem umiku komponente iz knjižnice.

Aktivnost*: *(B.5.2) Problemska analiza in analiza sprememb

Izvajalec aktivnosti: vzdrževalec aplikacije.

Opis in rezultati

Opis aktivnosti: analizira vsa poročila o težavah s komponentami oziroma zahteve po spremembah ter ugotovi morebitne vplive teh sprememb na preostale ponovno uporabljene komponente.

[117] Na temelju tako zbranih informacij je možno pregledati tudi vse druge sisteme ali aplikacije, v katerih so bile "sporne" komponente ponovno uporabljene, ugotoviti pa je mogoče tudi vpliv nastalih problemov oziroma sprememb na aplikacijo, v kateri so bile ponovno uporabljene.

Opis rezultatov: poročilo o izvedeni problemski analizi.

Ponovno uporabljene komponente: vzdrževalne komponente, vzorec ali predloga za izvedbo problemske analize.

Posredovanje ponovno uporabne informacije

Izvor: vzdrževalec aplikacije.

Ponor: upravitelj komponent.

Vsebina: vse ugotovitve in izsledke o ponovno uporabljenih komponentah.

Pravne aktivnosti: uveljavljanje določil o vzdrževanju ponovno uporabnih komponent in obveščanju njihovih ponovnih uporabnikov o spremembah ali težavah s komponentami, ki so sestavni del licenčnih pogodb za posamezne komponente.

Aktivnost*: *(B.5.3) Implementacija sprememb

Izvajalec aktivnosti: vzdrževalec aplikacije.

Opis in rezultati *Opis aktivnosti:* z analizo ugotovi, katero dokumentacijo, katere komponente in katere različice komponent je treba spremeniti in upravitelju komponent naroči spremembo oziroma popravke.

Opis rezultatov: spremenjene ali "popravljene" komponente.

Posredovanje ponovno uporabne informacije

Izvor: vzdrževalec aplikacije.

Ponor: upravitelj komponent.

Vsebina: povratne informacije o ponovni uporabi spremenjenih komponent.

Pravne aktivnosti: podpis ustrezne pogodbe o vzdrževanju med knjižnico in razvijalcem, če naj predlagane spremembe ali popravke izvede kar razvijalec.

Aktivnost*: *(B.5.4) Selitev oziroma preselitev izdelka v novo okolje (migracija)

Izvajalec aktivnosti: upravitelj komponent.

Opis in rezultati

Opis aktivnosti: preselitev napove z objavo na elektronski oglasni deski ali po elektronski pošti; dokumentira obnašanje sistema in posameznih ponovno uporabljenih komponent v novem okolju.

Opis rezultatov: poročilo o preselitvi.

Ponovno uporabljene komponente: predloga za izdelavo poročila o preselitvi.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: v aplikaciji ponovno uporabljene komponente v novem okolju.

Kriteriji: stabilnost komponent, nespremenjenost funkcionalnosti komponent in integracije.

Posredovanje ponovno uporabne informacije

Izvor: upravitelj komponent.

Ponor: vsi uporabniki komponent (razvijalci aplikacij), razvijalci komponent.

Vsebina: informacija o preselitvi iz ponovno uporabljenih komponent sestavljene aplikacije v novo okolje ter morebitnih učinkih in vplivih nastalih sprememb, ki so še posebno pomembne za inženirje domene.

Pravne aktivnosti: zahtevanje povrnitve stroškov ali plačila odškodnine, če je bilo v specifikaciji ponovno uporabljene komponente določeno tudi njeno delovanje v okolju, ki ustreza novemu okolju aplikacije, pa se izkaže, da v tem okolju komponenta ne deluje pravilno.

Aktivnost*: *(B.5.5) Upokojitev sistema

Izvajalec aktivnosti: vzdrževalec aplikacije.

Opis in rezultati

Opis aktivnosti: umik sistema ali aplikacije iz delovanja.

Opis rezultatov: dokument o umiku aplikacije oziroma trajni prekinitvi njenega delovanja, skupaj z razlogi, ki so do tega privedli.

Ponovno uporabljene komponente: dokumentacijske komponente ali predloga poročila o upokojitvi sistema.

Posredovanje ponovno uporabne informacije

Izvor: vzdrževalec aplikacije, upravitelj komponent.

Ponor: vsi uporabniki knjižničnih komponent.

Vsebina: informacije o upokojitvi aplikacije, seznam v tej aplikaciji ponovno uporabljenih komponent, ki so še v knjižnici ter razlogi za upokojitev sistema.

Pravne aktivnosti: prekinitve še veljavnih licenc za ponovno uporabljene komponente, ki so bile uporabljene v sistemu, ki se ga upokojuje.

4.2.1.3 C – Procesi med razvojnimi življenjskimi cikli različnih projektov

Proces, ki podpira ponovno uporabo med različnimi projekti, se imenuje inženirstvo domene ali tudi *razvoj za ponovno uporabo*.

C.1 Proces inženirstva domene

Proces inženirstva domene pokriva razvoj in vzdrževanje modelov domene, arhitekture domene in drugih ponovno uporabnih komponent domene in je v celoti za ponovno uporabo specifičen proces. Osnovne aktivnosti procesa inženirstva domene so analiza domene, načrtovanje domene, pridobivanje komponent ter njihovo vzdrževanje. Aktivnosti se med seboj lahko prekrivajo, lahko se izvajajo iterativno ali rekurzivno. Aktivnosti inženirstva domene lahko sovpadajo z aktivnostmi razvoja in vzdrževanja aplikacij.

Slika 10: Aktivnosti procesa inženirstva domene.

Aktivnost*: *(C.1.1) Implementacija procesa inženirstva domene

Izvajalec aktivnosti: inženir domene.

Opis in rezultati

Opis aktivnosti: izdela plan, v katerem opiše standarde, metode, orodja, aktivnosti, dodelitve nalog in odgovornosti za izvedbo procesa inženirstva domene, pri čemer se posvetuje z izvedenci domene, razvijalci in uporabniki programskih izdelkov domene, po možnosti pa si pomaga tudi z razpoložljivo literaturo in drugimi podatkovnimi viri za domeno; za modele domene in arhitekturo domene izbere obliko predstavitve, ki se mora ujemati s standardi ponovne uporabe organizacije; opredeli postopke izmenjave informacij, reševanja težav in povratnega informiranja, ki jih bo upravitelj komponent uporabil vsakič, ko pride do težav z neko komponento, ki jo je on razvil, oziroma ko pride do sprememb zahtev za ponovno uporabno komponento.

Opis rezultatov: plan implementacije procesa inženirstva domene.

Ponovno uporabljene komponente: predloga plana za izvedbo inženirstva domene.

Ocena po kriterijih ponovne uporabnosti -

Posredovanje ponovno uporabne informacije

Izvor: inženir domene.

Ponor: izvedenec (ekspert) domene, razvijalec aplikacije, uporabnik aplikacije.

Vsebina: način predstavitve modelov domene in arhitekture domene.

Pravne aktivnosti -

Aktivnost: ***(C.1.2) Analiza domene***

Izvajalec aktivnosti: inženir domene.

Opis in rezultati *Opis aktivnosti:* preuči podobnosti in razlike znotraj domene ter pridobljene informacije zajame v modelih domene; domeni mora postaviti meje in ugotoviti, kakšne so povezave med izbrano domeno in drugimi domenami; identificira trenutne in bodoče potrebe razvijalcev aplikacij znotraj domene; modele domene oblikuje v skladu s predstavitvijo, ki je bila izbrana med implementacijo procesa inženirstva domene; priporočljivo je, da poleg modelov domene zgradi tudi slovar terminov, ki opisujejo pomembne koncepte domene ter medsebojne odnose med podobnimi ali skupnimi komponentami domene; modele domene ustrezno dokumentira in klasificira.

Opis rezultatov: modeli domene, slovar domene.

Ponovno uporabljene komponente: znanje o domeni, obstoječi modeli domene.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: modeli domene, slovar domene.

Kriteriji: skladnosti modelov in slovarja domene z izbrano modelirno tehniko ter s postopki sprejema in certificiranja komponent, ki jih je sprejela organizacija; skladnost modelov domene z dejanskim stanjem domene (periodično ob pomoči razvijalcev aplikacij, upravitelj komponent, ekspertov domene in uporabnikov aplikacij).

Posredovanje ponovno uporabne informacije

Izvor: inženir domene.

Ponor: upravitelj komponent.

*Vsebina:*informacije o stanju domene, modeli domene in njihove spremembe.

Pravne aktivnosti: določitev morebitnih omejitev pri ponovni uporabi modelov domene.

Aktivnost*: *(C.1.3) Načrtovanje domene

Izvajalec aktivnosti: inženir domene.

Opis in rezultati

Opis aktivnosti: definira in dokumentira arhitekturo domene[118], ki mora biti usklajena z izbranim modelom domene in standardi organizacije; specificira komponente, ki jih je mogoče z definirano arhitekturo ponovno uporabiti pri gradnji novih programskih izdelkov; za vsako komponento, ki bo načrtovana za ponovno uporabo, razvije in dokumentira specifikacijo te komponente; periodično pregleduje ustreznost arhitekture domene z dejanskim stanjem domene in jo po potrebi prilagodi.

Opis rezultatov: arhitektura domene, specifikacije za ponovno uporabne komponente.

Ponovno uporabljene komponente: obstoječa arhitektura domene in dokumentacijske komponente.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: arhitektura domene, specifikacija vsake ponovno uporabne komponente.

Kriteriji: ovrednoti skladnost arhitekture z uporabljeno načrtovalsko tehniko, ki je bila izbrana v postopkih sprejemanja in certificiranja komponent, ovrednoti skladnost specifikacije za vsako komponento s postopki za sprejemanje in certificiranje.

Posredovanje ponovno uporabne informacije

Izvor: inženir domene.

Ponor: upravitelj komponent.

Vsebina: arhitektura in vse njene morebitne spremembe.

Pravne aktivnosti: opremi arhitekturo in vse druge ponovno uporabne komponente z zaznamki o lastništvu in informacijami pravne narave.

Aktivnost*: *(C.1.4) Oskrbovanje s komponentami

Izvajalec aktivnosti: inženir domene.

Opis in rezultati

Opis aktivnosti: oskrbuje proces razvoja aplikacij s komponentami (po postopku A.1 ali jo razvije na novo); klasificira in dokumentira komponente; skupaj z razvijalci aplikacij in

[118] Arhitektura domene je visokonivojski načrt, v katerem so vmesniki med komponentami samo formalno definirani. Rabi kot ogrodje za ponovno uporabo komponent pri gradnji programskih izdelkov.

upravitelji komponent periodično pregleduje komponente in njihove specifikacije ter jih po potrebi prilagaja nastalim spremembam v domeni; posreduje komponente upravitelju komponent.

Opis rezultatov: ponovno uporabne komponente.

Ponovno uporabljene komponente: ogrodja za klasifikacije in specifikacije komponent, dokumentacijske komponente.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: posamezna ponovno uporabna komponenta.

Kriteriji: ovrednoti skladnost komponente s postopki sprejemanja in certificiranja komponent.

Posredovanje ponovno uporabne informacije

Izvor: inženir domene.

Ponor: upravitelj komponent, razvijalec aplikacij.

*Vsebina:*ponovno uporabne komponente, opremljene z informacijami za ponovno uporabo.

Pravne aktivnosti: s postopkom pridobivanja komponent se sproži tudi postopek priprave enega ali več predlogov licence za komponento.

Aktivnost*: *(C.1.5) Vzdrževanje komponent

Izvajalec aktivnosti: inženir domene.

Opis in rezultati

Opis aktivnosti: spreminja komponente[119], vključno z modeli in arhitekturami domene, na zahtevo upravitelja komponent - analizira zahteve po spremembah in izbira možnosti za izvedbo teh sprememb, za to pa izdela terminski in vsebinski plan sprememb.

Opis rezultatov: spremenjene ponovno uporabne komponente, opremljene z vso potrebno dokumentacijo, terminski in vsebinski plani sprememb.

Ponovno uporabljene komponente: predloge planov, dokumentacijske komponente, preskusne komponente.

Ocena po kriterijih ponovne uporabnosti

Izdelek, ki se ocenjuje: spremenjena ponovno uporabna komponenta.

[119] Odpravljanje določene napake ali pomanjkljivosti ali prilagoditev komponente novim oziroma spremenjenim zahtevam.

Kriteriji: skladnost komponente z modeli in arhitekturo domene, vpliv sprememb na programske izdelke in aplikacije, v katerih je ta komponenta uporabljena, vpliv sprememb na bodočo uporabo komponente ter splošno na njeno ponovno uporabnost.

Posredovanje ponovno uporabne informacije

Izvor: inženir domene.

Ponor: upravitelj komponent in razvijalci aplikacij.

Vsebina: spremenjene komponente, skupaj z navodili za njihovo (ponovno) uporabo in pripadajočimi preskusnimi komponentami.

Pravne aktivnosti: izvajanje določb o vzdrževanju komponent (iz pogodbe o vzdrževanju ali posameznih licenčnih pogodb) in preverjanje izvajanja teh določb, morebitne spremembe lastništva in omejitev uporabe posameznih komponent po izvedenem vzdrževanju.

4.2.1.4 D – Spremljajoči procesi

Z vidika ponovne uporabe imajo spremljajoči procesi, ki so prikazani na sliki 11, posebno vlogo, saj dejansko omogočajo in podpirajo ponovno uporabo komponent, zato sem jih poimenovala tudi *procesi v podporo ponovni uporabi*.

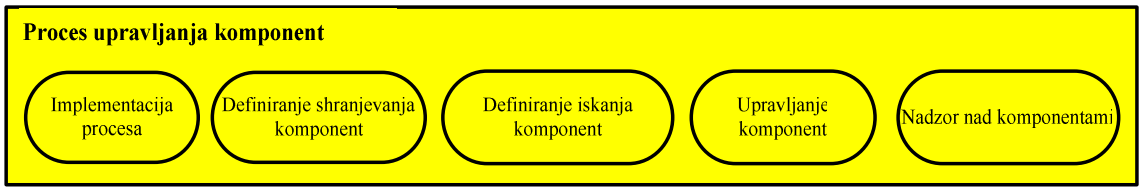
Slika 11: Spremljajoči procesi razvojnega življenjskega cikla [144].

Glede na to, da se vse aktivnosti in opravila tako ali drugače nanašajo na ponovno uporabne komponente, pa so za ponovno uporabo najpomembnejše aktivnosti in opravila procesa upravljanja komponent.

D.1 Proces upravljanja komponent

Same ponovno uporabne komponente za organizacijo nimajo bistvenega, sploh pa ne strateškega pomena, če potencialni ponovni uporabniki zanje ne vedo in jih ne razumejo. Zato so vsi procesi in aktivnosti, povezani z upravljanjem komponent, bistvenega pomena za

uspešnost celotnega razvojnega procesa na temelju ponovne uporabe. Ključni akter znotraj teh procesov je upravitelj ali posrednik komponent, katerega primarna skrb je vzdrževanje knjižnice ponovno uporabnih komponent in odnosov med knjižnico in njenimi člani.
Upravitelj komponent je zadolžen za izvajanje vseh administrativnih in tehničnih postopkov v času življenja komponente. Ti postopki se nanašajo na identifikacijo, definiranje, certificiranje, klasificiranje, dokumentiranje komponent, sledenje njihovim spremembam, preselitvam v nova okolja delovanja ter nastajanjem novih različic. Poleg tega pa mora upravitelj komponent zapisovati in poročati o statusu komponente, postaviti in nadzirati sistem shranjevanja ter rokovanja s komponentami, dostavljati komponente ponovnim uporabnikom ter poskrbeti za njihovo upokojitev.



Slika 12: Glavne aktivnosti procesa upravljanja komponent.

Aktivnost (D.1.1) Implementacija procesa upravljanja komponent

Upravitelj komponent mora najprej postaviti plan svojega dela, po možnosti na temelju predloge plana, ki jo ponovno uporabi. Plan, ki je tesno povezan s pravili knjižnice, mora določati: mehanizem shranjevanja in iskanja komponent, postopke za sprejem, certificiranje, upokojitev ter licenciranje komponente, medsebojni odnos med upraviteljem komponent ter razvijalci, vzdrževalci in inženirji domene, mehanizem komuniciranja oziroma obveščanja o vsem, kar je povezano s komponentami knjižnice, klasifikacijske sheme. Vsi našteti postopki so podrobno opisani v 5. poglavju.
Upravitelj komponent mora dokumentirati celoten proces upravljanja komponent, poskrbeti za ustreznost oblike vsake komponente, dokumentirati in odpravljati težave in neskladnosti komponent ter periodično izvajati preglede (revizije) komponent.

Aktivnost (D.1.2) Definicija shranjevanja in iskanja komponent

Mehanizma shranjevanja in iskanja komponent ponovnemu uporabniku omogočata, da na preprost in hiter način najde in razume najdeno komponento.

Za implementacijo teh mehanizmov je zadolžen upravitelj komponent. Poleg tega pa mora upravitelj komponent tudi razviti, dokumentirati in vzdrževati klasifikacijsko shemo, ki se bo uporabljala pri klasifikaciji komponent v postopku njihovega shranjevanja v knjižnico.

Aktivnost (D.1.3) Upravljanje komponent in nadzor nad njimi

Vsaka komponenta, ki je upravitelju poslana kot kandidatna komponenta za shranitev v knjižnico, mora iti skozi postopek ocenjevanja, ki se izvede po kriterijih za sprejem in certificiranje komponente, definiranih v planu ali pravilih knjižnice. Vsaka v knjižnico sprejeta komponenta mora biti za ponovno uporabo dostopna prek mehanizmov shranjevanja in iskanja. Komponenta mora biti klasificirana po klasifikacijski shemi za ponovno uporabo. Nad komponento je treba izvesti postopek konfiguracije, saj morajo biti komponente shranjene v predpisani obliki. Upravitelj komponent mora tudi vzdrževati sled o vsaki ponovni uporabi komponente in inženirju domene poročati o tekočih ponovnih uporabah komponente[120].

Zahteve po modifikacijah komponent ali poročila o težavah s komponentami skupaj s komponentami upravitelj komponent posreduje inženirjem domene v pregled in izvedbo popravkov oziroma sprememb. Inženirji domene poskrbijo za posredovanje povratnih informacij o vseh izvedenih akcijah. Upravitelj komponent pa je zadolžen za spremljanje in zapisovanje vseh zahtev oziroma poročil o težavah ter akcijah, ki sledijo. O težavah s komponento, na njej narejenih spremembah, novih različicah ali brisanju komponente iz knjižnice je upravitelj komponent dolžan poročati ponovnim uporabnikom komponent (uporabnikom knjižnice) in inženirjem domene.

4.2.2 Cilji posameznih procesov ponovne uporabe

Pri definiranju splošnih ogrodij ali modelov se je treba odreči definiranju podrobnosti in posebnosti, ki splošnost zmanjšujejo. Zato pa je toliko bolj pomembno, da se ta ogrodja in modeli postavijo na temelju jasno določenih ciljev, ki naj bi jih z aplikacijo ogrodij dosegli. V nadaljevanju so po posameznih, za ponovno uporabo specifičnih procesih, ki so bili opisani v

[120] Informacija o ponovni uporabi komponente naj bi vključevala ime ponovnega uporabnika, podatke o projektu, podatke o originalnem razvijalcu (avtorju) ali lastniku komponente, stroške ponovne uporabe komponente ter prihranke in prednosti, ki izhajajo iz ponovne uporabe komponente.

predhodnih delih poglavja, povzeti cilji ali dosežki, ki jih lahko izvajalec procesa ponovne uporabe pričakuje oziroma doseže z izvajanjem ponovni uporabi prilagojenega življenjskega cikla.

Proces administracije programa ponovne uporabe

- opredeliti strategijo ponovne uporabe organizacije, ki vključuje namene, naloge, cilje in njen obseg;
- identificirati sodelujoče (udeležence) v programu/procesu ponovne uporabe;
- ustanoviti vodstveno in nadzorno telo – odbor za vodenje in nadzor nad izvajanjem programa ponovne uporabe, ki bdi nad načrtovanjem in izvajanjem ponovne uporabe;
- ustanoviti aktivnosti v podporo programu ponovne uporabe;
- izdelati izvedbeni plan ponovne uporabe za organizacijo;
- vzpostaviti mehanizme komuniciranja, obveščanja ter povratnega informiranja med administratorji ponovne uporabe, upravitelji komponent, inženirji domene, razvijalci aplikacij in vzdrževalci;
- identificirati domene, v katerih namerava organizacija preučiti možnosti ponovne uporabe in v njih ponovno uporabo tudi izvajati;
- oceniti sposobnost oziroma zrelost organizacije za izvajanje sistematične ponovne uporabe;
- dodelati oziroma nadgraditi strategijo ponovne uporabe organizacije ter planov ponovne uporabe na temelju rezultatov, ki izhajajo iz ocen ponovne uporabe;
- izvajati plan implementacije ponovne uporabe;
- sprotno spremljati in vrednotiti ter izboljševati program ponovne uporabe;

Proces pridobivanja

- pregledati obstoječe komponente in s tem ugotoviti, ali komponente, ki jih pri razvoju programske opreme potrebujemo, že obstajajo;
- ponovno uporabo obravnavati kot del analize tveganj, stroškov in pridobitev;
- izdelati plan pridobivanja komponent na temelju predloge, če je na voljo;
- v dokumentacijo, povezano s pridobivanjem komponent, vključiti vse zahteve za izvajanje ponovne uporabe, ki jih bo moral dobavitelj komponent izpolniti;

- med preskusne primere za sprejem komponente vključiti preskuse ponovne uporabnosti in preskuse združljivosti z arhitekturo domene.

Proces dobavljanja komponent

- pripraviti predlog (ponudbo) na temelju predloge, če le ta obstaja; ponudba je odgovor na zahtevek za komponente (RFP), ki ga izdela pridobitelj komponent;
- pripraviti pogodbo na temelju ustreznih ponovno uporabnih predlog;
- določiti postopke in vire, ki so potrebni za upravljanje projekta, katerega rezultat bo sistem oziroma aplikacija ali komponenta, ki bodo posredovani pridobitelju.

Razvojni proces

- izbrati razvojni življenjski cikel, ki podpira ponovno uporabo;
- izbrati in uporabiti standarde, metode, orodja ter programske jezike, ki podpirajo, spodbujajo in omogočajo izvajanje ponovne uporabe znotraj projekta;
- pri izdelavi vseh planov (projektnega plana, plana integracije ter plana namestitve) in specifikacije zahtev ponovno uporabiti ustrezajoče predloge, če le obstajajo;
- zbrati kandidatne programske izdelke in komponente, ki ustrezajo specifikaciji sistemskih zahtev in jih je mogoče ponovno uporabiti pri razvoju aplikacije;
- identificirati in dokumentirati nove programske izdelke in komponente, ki jih je možno izdelati znotraj projekta (kot ponovno uporabne komponente);
- pri opredelitvi, dokumentiranju in ovrednotenju sistemskih zahtev, načrtovanju, dokumentiranju in ovrednotenju podatkovne baze na temelju modela domene, izdelavi uporabniške dokumentacije, opredelitvi preskusnih zahtev, izdelavi podrobnega načrta za vsak sestavni del programskega izdelka, izdelavi kode in dokumentacije za vsak programski delček in podatkovno bazo ter njihovem preskušanju, izvajanju kvalifikacijskega programskega preskušanja ter kvalifikacijskega sistemskega preskušanja ponovno uporabiti ustrezne obstoječe komponente, če so na voljo;
- ovrednotiti ponovno uporabnost specifikacije sistemskih zahtev, sistemske arhitekture, specifikacije programskih zahtev, programske arhitekture, programskega načrta, kode ter preskusnih komponent.

Proces delovanja

- razviti plan delovanja sistema, pri tem pa po možnosti ponovno uporabiti obstoječe predloge takih planov.

Proces vzdrževanja

- analizirati zahteve po spremembah, poročila o težavah pri ponovni uporabi komponent in izvedbene možnosti teh sprememb glede na njihov vpliv na preostale komponente ter analizirati priložnosti za uporabo modificiranih komponent;
- upravitelju komponent posredovati zahteve po spremembah ter poročila o težavah;
- obvestiti upravitelja komponent o načrtih za migracijo ali upokojitev programskih izdelkov, zgrajenih iz komponent knjižnice.

Proces inženirstva domene

- izbrati obliko predstavitve za modele in arhitekture domene;
- določiti meje domene ter medsebojne odnose te domene z drugimi domenami;
- opredeliti, klasificirati in dokumentirati množico modelov domene, ki zajemajo skupne lastnosti in razlike znotraj domene, zmožnosti, koncepte in funkcije domene;
- ovrednotiti modele domene;
- specificirati komponente, ki pripadajo domeni;
- opredeliti, klasificirati in dokumentirati ter ovrednotiti arhitekture domene;
- razviti ali pridobiti, dokumentirati in klasificirati ter ovrednotiti ponovno uporabne komponente;
- modele in arhitekture domene ter komponente posredovati upravitelju komponent.

Proces upravljanja komponent

- razviti in dokumentirati plan upravljanja komponent na temelju ponovno uporabne predloge takega plana, če je ta na voljo;
- razviti, dokumentirati in vzdrževati klasifikacijsko shemo za komponente;
- opredeliti zahteve organizacije, ki se nanašajo na mehanizme shranjevanja in iskanja komponent ter te mehanizme načrtovati, izvesti in vzdrževati;
- opredeliti postopke in kriterije dostopov, sprejema in upokojitve komponent iz knjižnice;

- vsako komponento, ki kandidira za vstop v knjižnico, je treba ovrednotiti v skladu s kriteriji in postopki za sprejem komponente v knjižnico, torej v skladu s pravili knjižnice;
- vsako sprejeto komponento je treba klasificirati v skladu s klasifikacijsko shemo in omogočiti razpoložljivost te komponente prek mehanizmov shranjevanja in iskanja komponent ter izvesti konfiguracijo komponente;
- vzdrževati sled za vsako ponovno uporabo komponente, spremembo zahtev za komponento ter poročati o napakah oziroma težavah pri uporabi te komponente, izpeljanih komponentah in različicah komponent ter podeljenih licencah za komponento; vsa obvestila s tem v zvezi posredovati administratorju programa ponovne uporabe, inženirju domene ter ponovnim uporabnikom komponente.

5 MODEL KNJIŽNICE PONOVNO UPORABNIH KOMPONENT

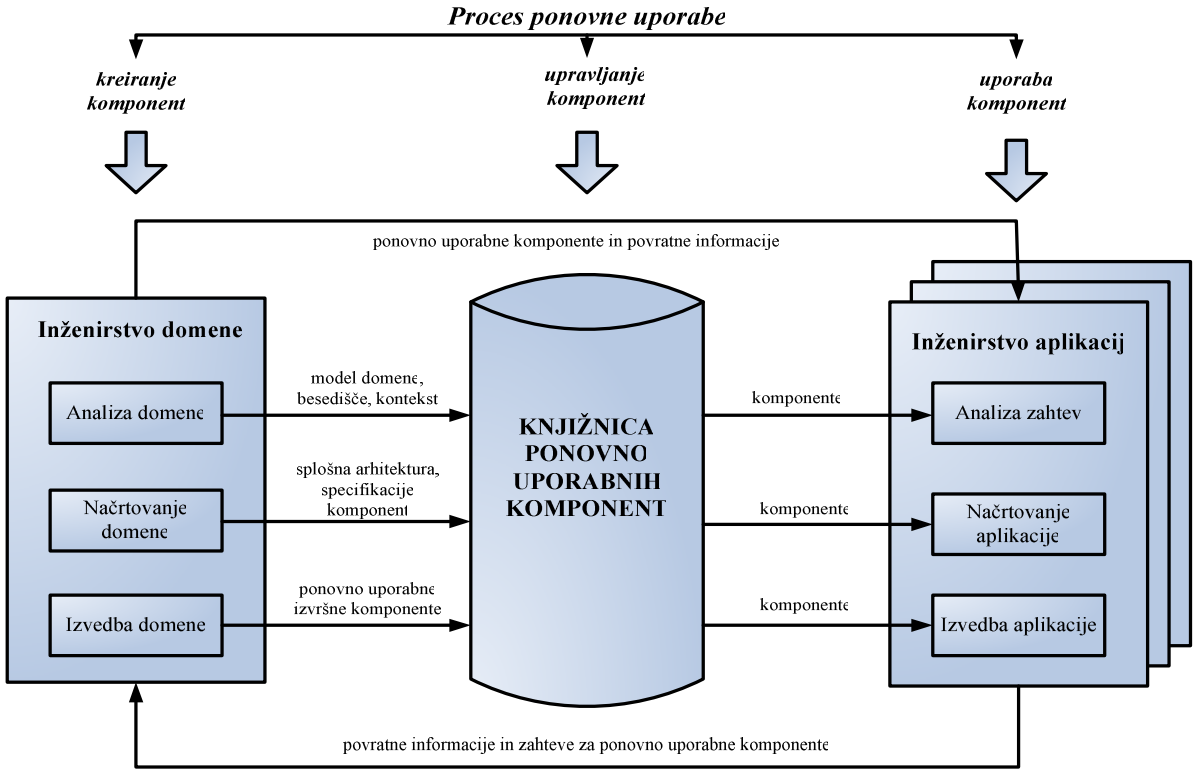
Bistvo procesa ponovne uporabe so ponovno uporabne komponente ter njihovo upravljanje. Da bi z njihovo pomočjo dosegli čim večji učinek, torej skrajšali in olajšali razvoj programskih sistemov z uporabo in sestavljanjem teh komponent, je treba komponente zgraditi in opremiti kakor samostojne, neodvisne pakete, ki bodo vsebovali vse potrebno za uspešno ponovno uporabo, omogočali razvoj iz njih izpeljanih komponent, omogočali sledljivost njihovi uporabi ter izdelavi različic itn. Komponente je treba shraniti v knjižnico, ki temelji na strukturi domene. Poleg natančne definicije strukture knjižnice je treba natančno opredeliti tudi vse postopke in pravila njenega delovanja. Struktura knjižnice je v marsičem odvisna od vsebine komponent in njihovih medsebojnih odvisnosti ter povezav. Odvisna pa je tudi od tega, ali bo knjižnica zaprtega tipa (samostojna, neodvisna enota) ali se bo povezovala navzven z drugimi knjižnicami. Uspešnost posamezne knjižnice in povezovanja knjižnice v komponentna tržišča je v veliki meri odvisna tudi od arhitekture knjižnice. Najbolje bi seveda bilo, da bi bila arhitektura standardizirana. V vsakem primeru, ne glede na to, za kakšne vrste in razsežnost knjižnice gre, pa je treba vse naštete bistvene elemente knjižnice zgraditi in medsebojno povezati tako, da bo zagotovljena celostnost komponent in s tem posredno tudi knjižnice.

V nadaljevanju je predstavljen model knjižnice ponovno uporabnih komponent z vsemi procesi, aktivnostmi in transakcijami, ki so potrebni za učinkovito dostopanje in brskanje po knjižnici, iskanje, izbiro in ponovno uporabo komponent knjižnice, skupaj s pravnimi varovalnimi mehanizmi, s katerimi je mogoče v zelo veliki meri preprečiti kršenje pravic intelektualne lastnine na komponentah in istočasno spodbujati ponovno uporabo. Model je zgrajen modularno in se da uporabiti tako v primeru interne knjižnice, do katere dostopajo samo člani ene organizacije ali celo samo razvojne skupine, kakor v primeru knjižnice, ki je dostopna tudi zunanjim uporabnikom, bodisi prek direktne komunikacijske povezave bodisi

prek spleta, vsebuje pa tudi vse elemente za povezovanje knjižnice navzven v komponentna tržišča.

5.1 KONCEPT SODOBNE KNJIŽNICE PONOVNO UPORABNIH KOMPONENT

Knjižnica ponovno uporabnih komponent je podatkovna baza komponent, ki so bile načrtno razvite za ponovno uporabo v različnih kontekstih. Predstavlja vezni člen med razvojem ponovno uporabnih komponent (razvoj za ponovno uporabo) in razvojem programskih sistemov iz ponovno uporabnih komponent (razvoj s ponovno uporabo), kar prikazuje spodnja slika.



Slika 13: Vloga knjižnice ponovno uporabnih komponent v procesu ponovne uporabe [56].

Uspešna knjižnica ponovno uporabnih komponent se mora prilagoditi potrebam organizacije, če je interna, ali potrebam tržišča, v katero se vključuje. V veliki meri je njena uspešnost odvisna tudi od strukture in procesov knjižnice. Pričakujemo namreč lahko, da se količina

komponent v knjižnici sčasoma tako poveča, da jih je brez dobro opredeljene strukture in procesov nemogoče učinkovito obvladovati. Bistvo vsake knjižnice naj bi bilo, da njeni uporabniki čim hitreje najdejo komponente, ki jih potrebujejo, poleg tega pa dobijo tudi zadostno količino kakovostnih informacij, ki jim omogočajo preprosto, hitro in učinkovito ponovno uporabo ali morebiti zavrnitev najdenih komponent.

Poleg dobro opredeljene strukture in procesov knjižnice pa na drugi strani predstavlja temelj dobro opredeljene knjižnice komponentni model, ki določa, katere informacije je treba shraniti v knjižnico skupaj s komponento. Dejansko predstavlja komponentni model vsebine komponent (elemente komponente) ter odvisnosti in povezave med njimi. Informacije, ki komponento spremljajo, so njen sestavni del, četudi niso fizično neposredno shranjene v komponenti, ampak so v njej samo povezave nanje.

Pri gradnji sodobne knjižnice ponovno uporabnih komponent, katere temeljni cilj naj bi bil lajšanje in spodbujanje ponovne uporabe, je treba upoštevati številne zahteve, tako udeležencev v procesu ponovne uporabe kakor tehnologij, ki so povezane z razvojem komponent in aplikacij ter dostopanjem do vsebin digitalnih podatkovnih baz prek spleta. Nekaj najpomembnejših zahtev glede tega, kaj mora sodobna knjižnica omogočati, je strnjenih v naslednjih alinejah. Te zahteve so hkrati predstavljale temeljne smernice pri oblikovanju pričujočega modela.

Sodobna knjižnica ponovno uporabnih komponent mora omogočiti in zagotoviti:

- shranjevanje visoko prilagodljivih komponent različne zrnatosti,
- ocenjevanje kakovosti, ponovne uporabnosti in pravic intelektualne lastnine,
- preprosto zamenljivost ene komponente z drugo,
- meta podatke o komponentah, ki pa so fizično ločeno shranjeni,
- vmesnik za dialog z uporabnikom in za predstavitev komponent,
- preprost in varen brskalnik,
- povezavo s številnimi razvojnimi in drugimi orodji[121],

[121] Npr. dokumentacijska orodja, konfiguracijska orodja, orodja za vizualizacijo komponent, orodja za šifriranje in dešifriranje vsebine komponent, orodja za licenciranje, orodja in postopki za oceno kakovosti komponent,

- preverjanje podobnosti med komponentami pred sprejemom komponente,
- vzdrževanje različic iste komponente,
- zelo kakovostne in hitre iskalne stroje ter njihovo stalno izboljševanje,
- ocenjevanje pridobitev s komponento[122],
- orodja za ocenjevanje izpeljanih del,
- mehanizme varovanja pravic intelektualne lastnine,
- preprosto in varno razpečevanje knjižničnih komponent,
- čim večjo standardiziranost[123] ter
- svoje povezovanje z drugimi knjižnicami domene.

Mehanizmi, ki jih knjižnica ponuja, morajo biti učinkoviti, člani knjižnice morajo biti z njihovim delovanjem seznanjeni, predvsem pa morajo biti ti mehanizmi varni tako z vidika varnosti komponent in storitev knjižnice kakor z vidika varnosti osebnih podatkov, povezanih s člani knjižnice ter varnega prenosa informacij, dokumentov in komponent med člani in knjižnico. Varnost mehanizmov knjižnice je eden od ključnih predpogojev za ustvarjanje zaupanja v knjižnico in njene komponente. Odsotnost tega zaupanja sodi med ene bistvenih zaviralnih elementov v procesu vzpostavljanja sistematične ponovne uporabe. Veliko težavam, med njimi tudi težavam pravne narave, se knjižnica lahko izogne z jasno opredeljenimi pravili dostopanja do virov knjižnice in rokovanja z njimi, opisom posameznih postopkov in mehanizmov, terminov, dokumentov itn. Pravila knjižnice predstavljajo najpomembnejšo pogodbo med knjižnico in njenimi člani.

Sodobna knjižnica mora biti večplastna: uporabnike mora ločevati glede na dostope do posameznih komponent, združevati pa mora tudi različne vrste komponent. Dostop je pogosto odvisen od vrste komponent, od vrste dostopa pa sta odvisna tako način izračuna članarine kakor vrsta licence in način licenciranja komponente. Predlagam, da bi knjižnica komponente neke domene razvrščala v naslednje štiri skupine: proste in javnodomenske komponente,

orodja za spremljanje stanja knjižnice in pridobivanje povratnih informacij, mehanizmi za sledenje uporabam komponente.

[122] Gre za pridobitve z njeno ponovno uporabo v smislu prihranka pri razvoju, pa tudi zaslužek pri prodaji komponente, v primerjavi s stroški njenega vzdrževanja.

[123] Standardna arhitektura, katere temelj so modeli domene, standardna terminologija, standardni nabor metrik, licenc ipd.

odprtokodne komponente, paketne (črne) komponente ter poslovne komponente. Pri slednjih dveh lahko komponente razvrstimo še glede na to, ali so dostopne samo notranjim ali tudi zunanjim članom[124]. Za dodeljevanje dostopov si knjižnica zgradi avtorizacijsko shemo, za njeno postavitev in posodabljanje ter dodeljevanje dostopov pa je odgovoren programski svet knjižnice[125].

5.2 VSEBINA PONOVNO UPORABNE KOMPONENTE

Zgraditi učinkovito ponovno uporabno komponento z visoko stopnjo ponovne uporabnosti je težko, saj je treba ustvariti nekaj, kar lahko tudi drugi razumejo in spet uporabijo brez posvetovanja z razvijalcem komponente. Dobra komponenta mora biti po eni strani dovolj majhna, da ostane pregledna in preprosta za uporabo, po drugi strani pa dovolj velika, da lahko že nekaj naredi. Z njo mora biti povezanih dovolj ključnih informacij, ki morajo biti ravno tako pregledne in preproste, sicer jih potencialni ponovni uporabnik ne bo pregledoval ali morda razumel. Predvsem pa morata biti iz teh informacij razvidna namen in kontekst uporabe komponente.

Namen uporabe komponente je mogoče opisati na različne načine. Lahko ga podamo s scenariji uporabe komponente, dokumentacijo, ki je nastala med preskušanjem, specifikacijo, načrtovalsko dokumentacijo, izvorno kodo in pripadajočo razvojno dokumentacijo, ki opisuje razne omejitve komponente. Eno temeljnih vodil pri razvoju ponovno uporabnih komponent pa mora biti njena preprostost. Ta je namreč predpogoj za ponovno uporabo vsake komponente, saj prekompleksne komponente razvijalcem niso zanimive, ker potrebujejo za njihovo razumevanje in vgraditev v razvijajoč sistem ali aplikacijo preveč časa.

Ponovno uporabno komponento lahko definiramo kot "paket nekih operacij ali funkcij, ki ga je mogoče uporabiti pri gradnji večje komponente ali programskega sistema" [30]; kot "predstavitev določenih vidikov sistema, ki jih je mogoče uporabiti v različnih aplikacijah"

[124] Če se lastnik knjižnice ukvarja z razvijanjem aplikacij, lahko do določenih komponent, v katere so vgrajene poslovne skrivnosti, omeji dostop samo na notranje člane.
[125] Programski svet knjižnice je telo, sestavljeno iz predstavnikov lastnikov in/ali upravljavcev knjižnice ter po možnosti tudi dobaviteljev komponent in njihovih ponovnih uporabnikov. Ukvarjal naj bi se predvsem z vprašanji, povezanimi z delovanjem in vsebino knjižnice, reševal morebitne težave med člani in knjižnico ter budno spremljal učinkovitost delovanja knjižnice.

[54]; kot "programski paket, ki ga je mogoče neodvisno razviti in uporabljati kot enoto, ki ponuja vmesnike, prek katerih ga je mogoče nespremenjenega povezati z drugimi komponentami v večji sistem" [24] itn. V splošnem pa lahko rečemo, da je *ponovno uporabna komponenta del nekega izdelka v razvojni fazi (specifikacija, načrtovanje, izvedba itn.), opremljen še z informacijami, ki omogočajo njeno ponovno uporabo.*

Z vidika ponovne uporabe najpomembnejši lastnosti, ki ju je v opisu komponente treba nujno zajeti, sta ponovna uporabnost komponente in njena kakovost. Med odločilne lastnosti za določanje ponovne uporabnosti lahko uvrstimo:

- prenosljivost komponente (komponente prenašamo iz okolja v okolje, zato morajo biti zelo prenosljive in neodvisne od okolja, v katerem so nastale),
- prilagodljivost komponente (komponento je treba pogosto prilagajati specifičnim zahtevam, zato mora biti modularna in splošna),
- razumljivost komponente (razumeti je treba funkcionalnost komponente, zato mora biti ustrezno dokumentirana) in
- zaupanje v komponento (ponovno uporabljena komponenta ne sme v sistem, v katerega je vgrajena, vnesti dodatnih tveganj, zato mora delovati v skladu s svojo specifikacijo).

Med odločilne lastnosti za določanje kakovosti pa uvrstimo:

- zanesljivost komponente (nizka stopnja napak) in
- vzdrževalnost komponente (za spreminjanje komponente v primeru sprememb funkcionalnih zahtev je nujno, da je komponenta konsistentna, samoopisna, preprosta, modularna in jo je mogoče preskušati).

Informacije, ki jih hranimo skupaj s komponento, so različne narave, opisujejo pa posamezne lastnosti ali značilnosti komponente, ki jih mora ponovni uporabnik nujno poznati, preden se odloči za njeno ponovno uporabo. V pričujočem delu sem te lastnosti razdelila v tri kategorije: tiste, ki se nanašajo na temeljne podatke o komponenti (opisne), tiste, ki govorijo o njeni notranji izvedbi in zgradbi (notranje) ter tiste, ki se nanašajo na interakcije komponente z okoljem (zunanje). Odločitev o vrstah informacij, ki naj bi jih s komponento hranili, sloni na modelu knjižnice, ki je predstavljen v nadaljevanju, predvsem pa na dejstvu, da bo ponovni

uporabnik izbral komponento, ki jo bo hitro razumel in preprosto uporabil. Katere lastnosti bodo pri opisu komponente izpuščene, je odvisno od vrste in zrnatosti komponente.

Pomembnost in nujnost opremljanja komponent z nekim minimalnim naborom informacij potrjujejo tudi razni predlogi za specifikacije komponente, npr. v maju 2004 sprejete specifikacije komponente RAS (ang. *Reusable Asset Specification*) [48], ki predstavlja predlog za standardiziran način pakiranja informacij s ponovno uporabno komponento ter minimalni nabor meta informacij, ki morajo spremljati posamezno komponento ali BIDM (ang. *Basic Interoperability Data Model*) podatkovni model za shranjevanje in izmenjavo metapodatkov [61]. Temelji v nadaljevanju predstavljene delitve in opisa informacij izhajajo iz obdobja zgodnjega avtoričinega raziskovanja vsebine ponovno uporabnih komponent [59], ki je bil ustrezno dodelan in dopolnjen v času postavitve modela knjižnice ponovno uporabnih komponent, predstavljenega v tem poglavju. Nabor informacij, katerih podrobnejši opis sledi, je z vsebinskega vidika popolnoma primerljiv z naboroma iz specifikacije RAS in modela BIDM, je pa hkrati tudi veliko obsežnejši.

5.2.1 Opisne lastnosti

Med splošne lastnosti o komponenti in njeni uporabi sodijo:

1. *Identifikacijska številka komponente (IDK)*: je enolični identifikator komponente, ki ga dobi ob sprejemu v knjižnico. Vsi sestavni deli komponente nosijo isto IDK številko, poleg tega omogoča sledenje transakcijam s komponento.
2. *Klasifikacija*: je eno od možnih sredstev za iskanje komponent[126], saj se nanaša na informacijo, prek katere lahko najdemo to komponento. S procesom klasifikacije se komponenti določi njeno mesto znotraj klasifikacijske sheme v podatkovnem modelu knjižnice. Klasifikacija mora biti ponovnemu uporabniku razumljiva, predvsem pa taka, da jo je mogoče uporabiti na komponentah različne zrnatosti, komponentah za različne razvojne faze, biti mora preprosta za uporabo, omogočati mora nadaljnji razvoj ter razširitev in zato ne sme biti preveč kompleksna.

[126] Drugi načini iskanja, ki se najpogosteje uporabljajo: brskanje, iskanje po atributih komponente, direktno iskanje prek identitete komponente, iskanje po ključnih besedah, ki naj bi jih komponenta vsebovala v opisu specifikacije ipd.

3. *Starost*: Na temelju starosti je mogoče sklepati o stabilnosti in zrelosti komponente. Starost se lahko poda z datumom, ko je bila komponenta zgrajena, datumom sprejetja v knjižnico, datumom zadnje spremembe komponente ali kako drugače. Tveganje pri ponovni uporabi nove (mlade) komponente je veliko večje kakor tveganje pri ponovni uporabi že večkrat uporabljene komponente, zato je pri izboru komponente za ponovno uporabo njena starost bistvena.
4. *Izvor*: Podatki o razvijalcu (avtorju) komponente, o dobavitelju komponente (ni nujno, da je to vedno razvijalec) in o trenutnem vzdrževalcu komponente so lahko zelo pomembni pri odločanju ponovnega uporabnika, ali bo neko komponento ponovno uporabil, saj želi pravno čiste komponente iz zanesljivih virov.
5. *Namen*: Podan je z opisom problema ali množice problemov, ki jih rešuje ali prispeva k njihovemu reševanju, iz katerega mora biti razvidna njena funkcionalnost.
6. *Kontekst*: Opisuje primere, v katerih je mogoče komponento ponovno uporabiti. Vse primere uporabe, splošne in posebne, je zelo težko določiti vnaprej, zato je treba določiti vsaj ciljne domene in okolja, v katerih jo je mogoče zagotovo uporabiti.
7. *Faza ponovne uporabe*: Komponento je mogoče uporabiti v različnih razvojnih fazah, odvisno od njene narave.
8. *Ocene*: so zbrane informacije o značilnostih komponente in njeni učinkovitosti, pridobljene na temelju metrik, ki podajajo številske vrednosti za te lastnosti. Shraniti je mogoče informacijo o velikosti komponente, njeni kompleksnosti, pogostnosti dostopov do komponente, številu poročanj uporabnikov komponente o napakah ali nepravilnostih in druge. [127]
9. *Podobne komponente*: v opis komponente je koristno dodati seznam podobnih, dobro znanih komponent, ki rešujejo isti problem ali isto množico problemov. Če gre za izpeljano komponento ali različico neke komponente, se lahko na tem mestu navede izvorna komponenta.
10. *Zgodovina ponovne uporabe*: ob vsaki ponovni uporabi komponente se shranijo podatki o ponovnem uporabniku, okolju in platformi, v katerih je bila komponenta ponovno

[127] Ponovni uporabnik si s temi informacijami pomaga pri razumevanju komponente in njenem vrednotenju, programski odbor ali svet knjižnice pa pri ocenjevanju koristnosti, ustreznosti, ponovne uporabnosti in kakovosti posamezne komponente.

uporabljena ter morebitna poročila o napakah ali nepravilnostih, ki so bile ugotovljene med ponovno uporabo.[128]

11. *Preskusni elementi*: v knjižnico se sprejmejo samo komponente, ki so že šle skozi proces preskušanja. Shraniti je treba vse podatke o izvedenih preskusih, podatke, na katerih so bili preskusi izvedeni ali preskusne primere, podatke o obnašanju komponente med preskušanjem ter o pričakovanih in doseženih rezultatih preskušanja.
12. *Nivo dostopa*: na podlagi avtorizacijske sheme je treba določiti, kdo ima dostop do dane komponente. Dostop je lahko omejen ali ni dovoljen posameznemu uporabniku ali skupini uporabnikov, preverja pa se z identifikacijsko številko člana in uporabnika ter gesla. Posebno občutljive komponente se lahko opremijo s posebnimi atributi, ki nakazujejo omejitve v uporabi komponente. Posebno pomembne so omejitve dostopa takrat, ko interno knjižnico povezujemo navzven z drugimi knjižnicami. V takem primeru je smiselno do komponent, v katere je vgrajena poslovna logika, omejiti dostope vsem zunanjim in morda tudi delu notranjih članov knjižnice. Dostopi so lahko omejeni na status članstva (notranji, zunanji), na vrsto ali kategorijo komponente (proste komponente, odprtokodne, črne komercialne komponente ali poslovne komponente) ali po različnih kombinacijah navedenih možnosti. Politiko dodeljevanja dostopov določa programski odbor knjižnice, lahko pa se s časom tudi spreminja.
13. *Status pregledanosti*: uporabniku mora biti omogočeno na preprost način pridobiti informacije o kakovosti posamezne komponente, saj se ravno na podlagi tega odloča o njeni ponovni uporabi. Informacije o kakovosti se komponenti pripišejo po končanem postopku pregledovanja komponent[129], ki se izvede pred sprejemom komponente v knjižnico. Komponente se po pregledu lahko uvrstijo v različne nivoje glede na status pregledanosti in s tem pridobijo posebno oznako, npr. nepregledana, preverjena, pregledana, pri čemer nepregledana pomeni, da je bila komponenta kar uvrščena v knjižnico, ker je specifična za področje knjižnice, preverjena pomeni, da je upravitelj knjižnice pregledal, ali komponenta ustreza vsem postavljenim merilom, pregledana pa pomeni, da je komponento pregledal izvedenec s področja knjižnice. Postopek pregledovanja se da nadgraditi še z dodatnimi merili, ocenjevalnimi lastnostmi,

[128] Del komponente je lahko tudi kazalec na tiste podatke od naštetih, za katere se knjižnica odloči, da jih bo javno odstopala svojim članom in je za to dobila ustrezno pooblastilo članov.
[129] Ta postopek mora biti dovolj prožen, da ga je mogoče uporabiti znotraj različnih strokovnih področij in za raznovrstne komponente.

metodami in orodji. Poleg oznake statusa lahko dodamo kazalec na povzetek pregleda, v katerem je opisan trenutni status pregledanosti te komponente in v katerem je tudi kazalec na pripadajoče dokumente in materiale, povezane s pregledi.

14. *Ocena stroškov*: na temelju podanega stroška razvoja komponente ter ocene stroškov prilagajanja komponente novemu kontekstu ali domeni (ocena kontekstne in domenske razdalje), lahko ponovni uporabnik oceni, ali se mu bolj splača komponento ponovno uporabiti ali jo razviti na novo. Ocene se lahko podajo tudi z zapisi konkretnih primerov ponovne uporabe.
15. *Certifikacijske informacije*: navedejo se certifikacijski postopki in metode, ki so bile izvedene nad komponento, kdo jih je izvedel, katere certifikate je komponenta pridobila.
16. *Pravne omejitve*: knjižnica lahko ta atribut uporabi za označitev komponente z vsemi z njo povezanimi pravicami intelektualne lastnine in pravnimi omejitvami. Tako označevanje rabi predvsem kot opozorilo vsem uporabnikom komponente in knjižnicam, ki to komponento uvažajo, da neke omejitve nad to komponento obstajajo, hkrati pa predstavljajo temelj za oblikovanje licenčnih določil. S takim označevanjem naj bi bilo izključeno kršenje raznih pravic in omejitev zaradi nevednosti ali nemarnosti uporabnikov. Poleg tega posredovanje vseh omejitev nad komponento pred njeno vizualizacijo ali zagonom ali celo nakupom uporabniku omogoča pravočasen odstop od te komponente ali zavrnitev komponente.

5.2.2 Zunanje lastnosti

Med lastnosti, ki se nanašajo na interakcije komponente z drugimi izdelki in platformo, za katero je bila grajena, prištevamo:

1. *Interoperativnost*: določa, kako komponenta sodeluje z drugimi komponentami sistema (nanaša se torej na aplikacijski vmesnik). [130] Podatke o interoperativnosti komponente naj bi posredovali tudi ponovni uporabniki.

[130] Stopnja interoperativnosti ponovno uporabne komponente mora biti zelo visoka, kar zahteva zelo natančno opredeljen vmesnik. Komponente naj bi sicer bile čim bolj neodvisne in samostojne, še vedno pa jih je treba povezovati z drugimi komponentami v celoto ter prilagajati novim potrebam. To pa je možno le prek parametrov vmesnika komponente.

2. *Prenosljivost*: določa sodelovanje komponente s platformo (nanaša se torej na vmesnik do platforme). Navesti je treba, za katero platformo je bila komponenta razvita in na katere platforme jo lahko prenesemo.
3. *Vloga*: pove, kaj komponenta ponuja drugim aplikacijskim izdelkom. Njena vloga je lahko aktivna (vpliva na druge izdelke in je podvržena vplivu drugih izdelkov) ali pasivna (je samo podvržena vplivom drugih izdelkov).
4. *Čas integracije*: določa, kdaj komponento integriramo v aplikacijo, med razvojem ali v času izvajanja aplikacije.
5. *Odvisnost od tehnologije*: opis tehnologije, od katere je komponenta odvisna, omejitve tehnologije, potrebne za razvoj in ponovno uporabo komponente v drugi aplikaciji.
6. *Nefunkcionalne lastnosti*: se nanašajo na razne vidike komponente, ki niso neposredno povezani z njeno funkcionalnostjo. Taki so npr. varnostni vidiki (nadzor dostopa, dokaz pristnosti), izvedbeni vidiki (zahteve procesorja in zahteve za delovanje v realnem času), zanesljivost komponente.

5.2.3 Notranje lastnosti

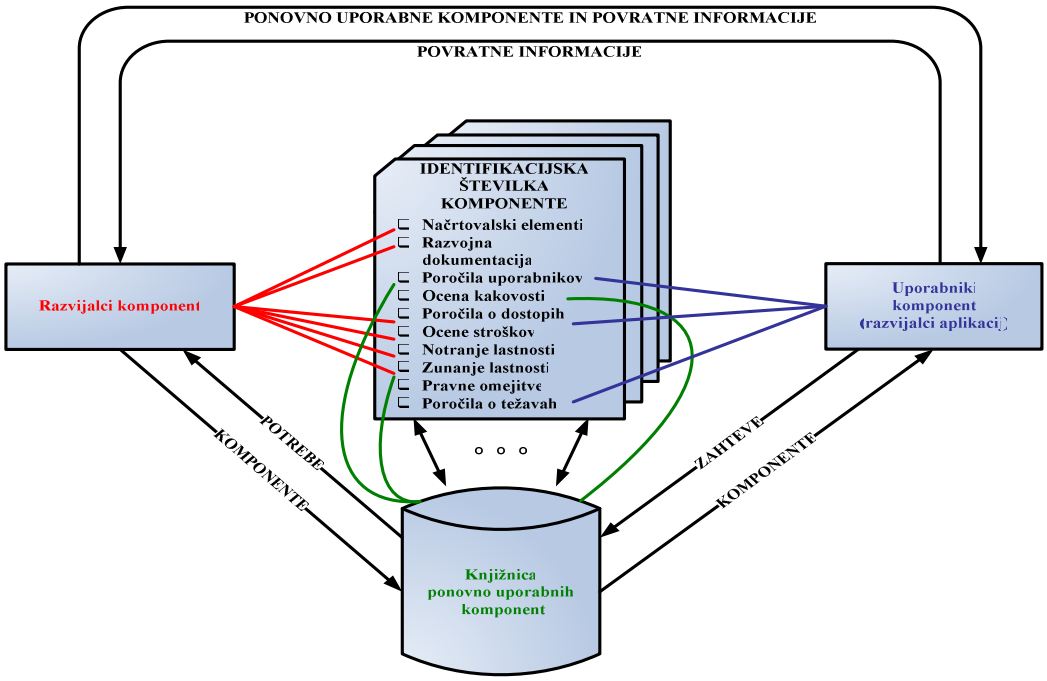
Nanašajo se na notranje vidike komponente, kot so:

1. *Tip ali narava komponente*: pove, kje in kdaj lahko v razvojnem procesu uporabimo komponento. Po tipu lahko komponente delimo na načrtovalske, specifikacije, izvršljive komponente ali kodo.
2. *Zrnatost*: navesti je treba zrnatost komponente po klasifikaciji zrnatosti, ki jo privzame knjižnica. Klasifikacijo zrnatosti je mogoče določati po različnih merilih, npr. glede na velikost komponente, fazo, v kateri je lahko ponovno uporabljena ali glede na poslovni vidik.
3. *Enkapsulacija ali skrivanje vsebine*: netrivialna komponenta združuje različne funkcionalnosti in odločitve. Nekatere od njih morajo biti skrite, sploh če so v ozadju poslovne skrivnosti.
4. *Struktura komponente*: opis sestavnih delov komponente (njenih elementov) ter njihovega medsebojnega sodelovanja in odvisnosti.
5. *Obnašanje komponente*: se nanaša na obnašanje komponente na specifično množico vhodnih podatkov in obnašanje komponente na zaporedje akcij.

6. *Dostop do izvorne kode*: razpoložljivost izvorne kode komponente določa stopnjo dostopnosti do te komponente in stopnjo njenega spreminjanja (izvorna koda je lahko na voljo, ni je pa dovoljeno spreminjati).[131] Od dostopnosti izvorne kode je tudi odvisna oblika ali strategija ponovne uporabe (bela, črna ali siva ponovna uporaba).

5.2.4 Združevanje informacij o komponenti

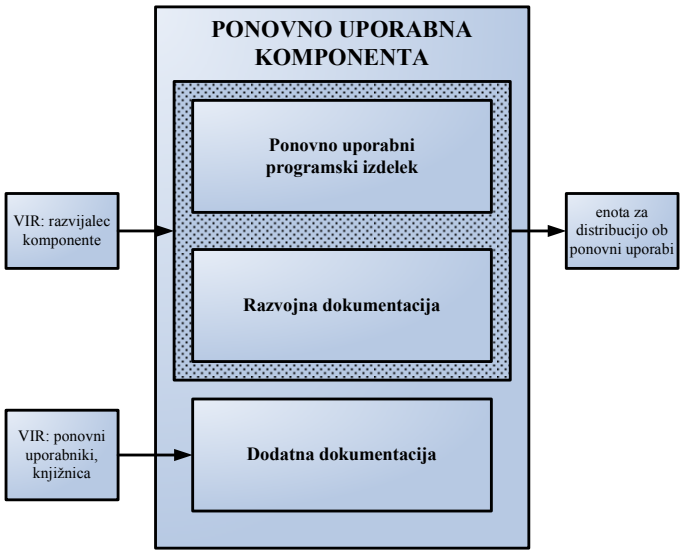
Označevanje komponent z navedenimi lastnostmi, ki se jim lahko dodajo tudi druge, prav gotovo pozitivno vpliva na uporabnost knjižnice in seveda ponovno uporabnost komponent. Razumevanje lastnosti komponent je zelo pomembno pri dokumentiranju, katalogiziranju in klasificiranju komponent. Ravno tako je z razumevanjem lastnosti in značilnosti posameznih komponent mogoče doseči boljšo uporabo komponent, pa tudi izbiro prave komponente glede na potrebe. Označevanje komponent z lastnostmi pripomore k lajšanju pridobivanja, spoznavanja in izbiranja komponent, torej na splošno izboljša proces iskanja komponent v knjižnici.



Slika 14: Vsebina komponente ter izvori posameznih elementov komponente.

[131] Čeprav večina raziskovalcev meni, da naj izvorna koda ne bi bila na razpolago, saj morebitno spreminjanje komponente povzroča padec zaupanja komponenti, sta stopnja in pogostnost ponovne uporabe komponente velikokrat odvisni ravno od razpoložljivosti njene izvorne kode.

Konceptualno obravnavam ponovno uporabno komponento kot paket, sestavljen iz treh obveznih delov, kot prikazuje slika 15. Glavni del predstavlja programski izdelek, ki naj bi se ponovno uporabil, drugi del predstavlja razvojna dokumentacija (načrtovalski elementi, notranje in zunanje lastnosti), tretji pa preostale informacije, združene pod imenom dodatna dokumentacija (poročila uporabnikov, ocene kakovosti in stroškov, poročila o dostopih, pravne omejitve, opisne lastnosti idr.). Pri razpečevanju komponente se vedno pošiljata programski izdelek in razvojna dokumentacija, medtem ko je dodatna dokumentacija permanentno shranjena v knjižnici in je na voljo samo na vpogled. Na tak način lahko knjižnica dodatno dokumentacijo stalno posodablja in dopolnjuje in jo tako ponuja prihodnjim ali aktualnim ponovnim uporabnikom kot pomoč pri odločanju ali že pri ponovni uporabi. Vir programskega izdelka in razvojne dokumentacije je razvijalec komponente, informacije, ki sestavljajo dodatno dokumentacijo, pa proizvajajo ali prispevajo bodisi ponovni uporabniki bodisi knjižnica, saj nastanejo kot rezultat raznih postopkov ocenjevanja, preskušanja, certificiranja, klasificiranja, merjenja, spremljanja stanja in uporab ipd.

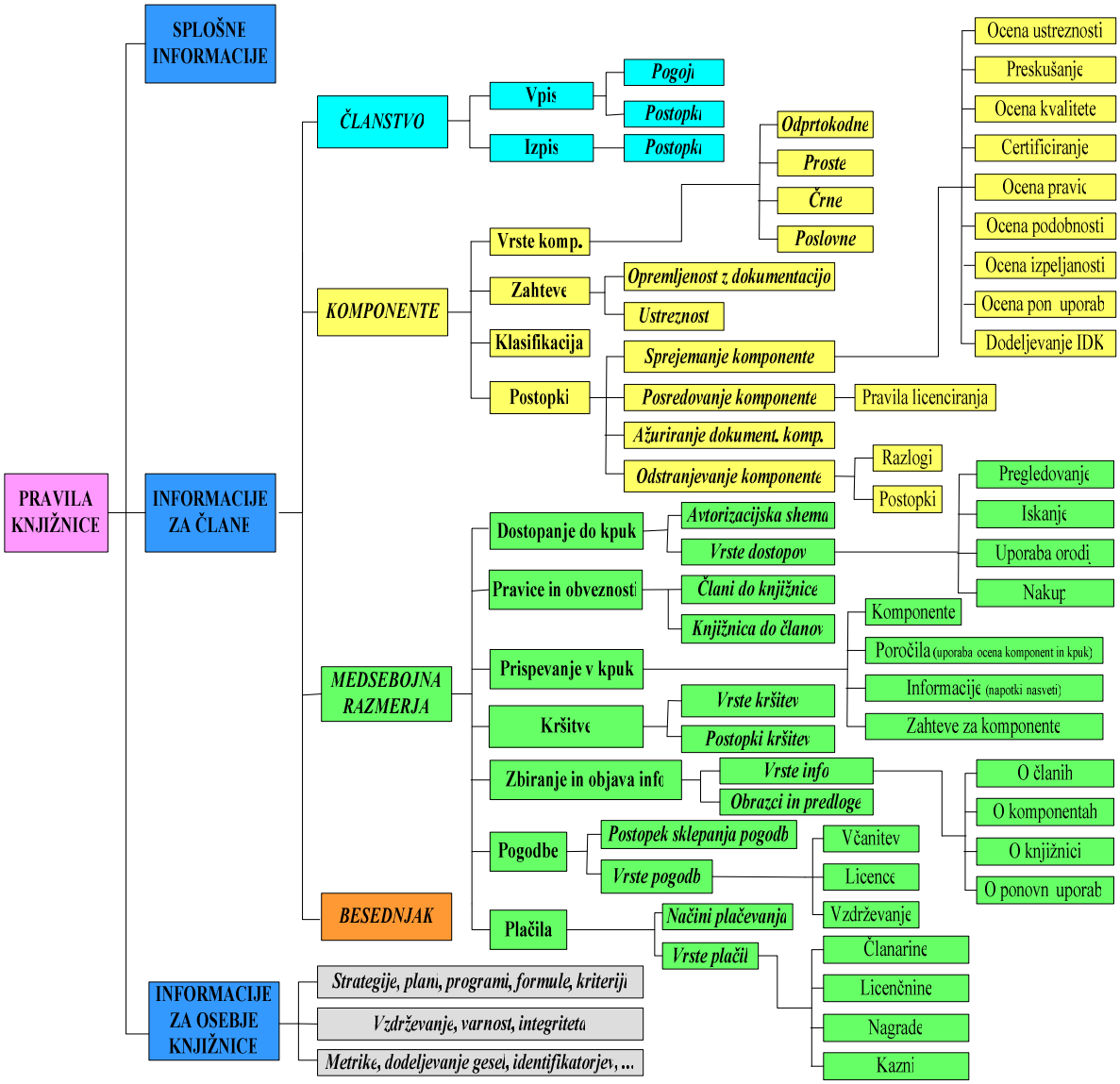


Slika 15: Ponovno uporabna komponenta kot paket.

5.3 PRAVILA KNJIŽNICE

Pravila so najpomembnejši dokument vsake knjižnice ponovno uporabnih komponent. Predstavljajo temeljno pogodbo med posameznim članom knjižnice in knjižnico, hkrati pa so

osebna izkaznica knjižnice, saj so v njih opisani vsebina knjižnice, temeljni koncepti in načini njenega delovanja ter rokovanja z njenimi viri. Za oblikovanje, posodabljanje ter nadzorovanje izvajanja pravil je zadolžen in odgovoren programski svet knjižnice.



Slika 16: Predlog vsebine in strukture pravil knjižnice.

Knjižnica se lahko odloči, da do dela vsebine pravil, ki se nanaša na splošne pogoje članstva ter uporabe virov knjižnice dovoli dostop vsem obiskovalcem spletne strani knjižnice, ne glede na to, ali se pozneje včlanijo ali ne (npr. splošne informacije). Del vsebine je dostopen samo članom knjižnice. V slednjem je predvsem vsebina, ki se nanaša na podrobnejši opis

delovanja knjižnice in njenih mehanizmov, pravil licenciranja, certificiranja, avtentifikacije, dostopanja, izračunov članarin, licenčnin ipd., kar lahko že prištevamo med poslovne skrivnosti knjižnice. Pred prvim dostopom do teh vsebin mora vsak član elektronsko podpisati izjavo o varovanju poslovnih skrivnosti, povezanih z vsebino, zgradbo in delovanjem knjižnice. Do dela pravil, ki se nanaša na opredelitev notranjih operacij, strukture, arhitekture, podrobnosti v zvezi z načrtovanjem in strategijami nadaljnjega razvoja knjižnice ter drugimi pomembnimi poslovnimi skrivnostmi, pa ima dostop samo osebje knjižnice.

Oblikovanje in izvedba pravil sta zelo zahtevni nalogi. Po eni strani morajo pravila obsegati vse informacije o knjižnici in njenem delovanju, kar vodi v zelo obsežen dokument z zelo prepletenimi in kot logično posledico tudi zelo podvojenimi vsebinami, po drugi strani pa morajo biti pregledna, razumljiva, preprosta za uporabo. Treba je zagotoviti tudi preprostost njihovega vzdrževanja in prilagajanja spremembam v delovanju knjižnice. Zato je nujno vsebino pravil ustrezno strukturirati ter posamezne module medsebojno povezati. Realizirana so lahko po vzoru elektronske pomoči, ki je vgrajena v vse okenske programe, in omogoča pregledovanje vsebine prek brskanja po posameznih poglavjih, iskanja po indeksnem kazalu ali s pomočjo ključnih besed.

Predlog strukture pravil sem na sliki 16 prikazala v drevesni obliki. V tej strukturi so vsebine predstavljene konceptualno, hierarhično. Koncept v ničemer ne zavezuje oblikovalcev izhodiščne spletne strani ali uporabniškega vmesnika knjižnice glede razporeditve vsebin na strani, s katero se knjižnica predstavlja.

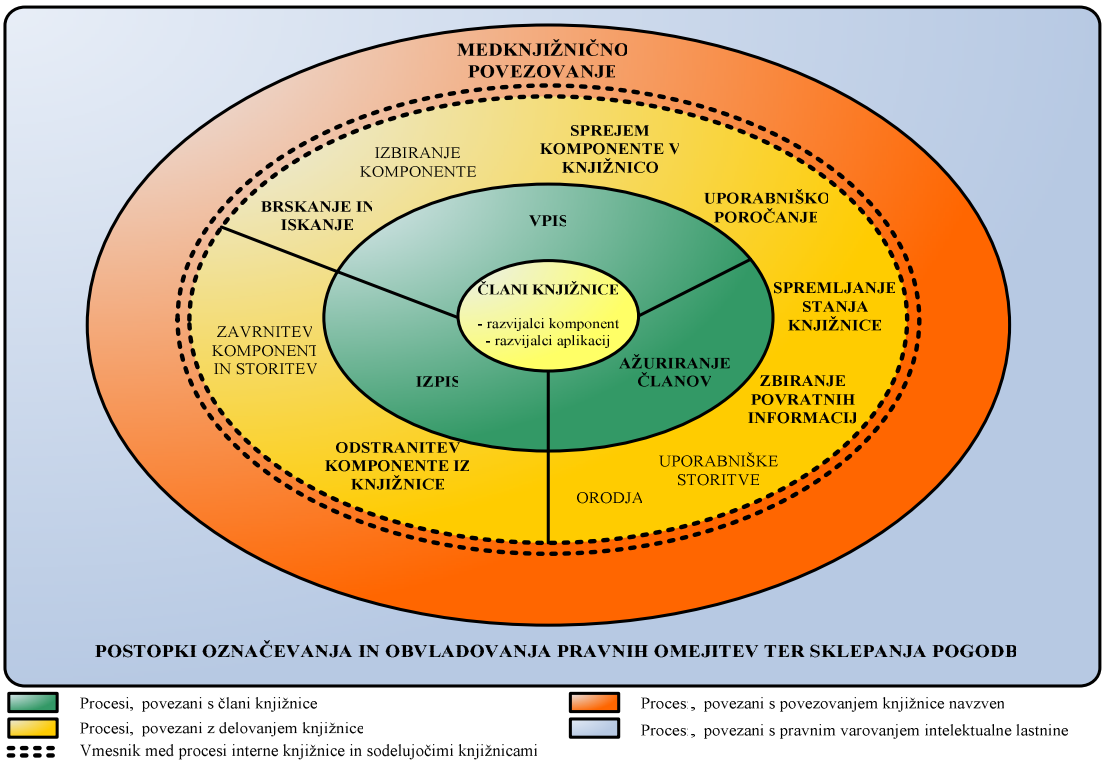
5.4 TEMELJNI PROCESI IN TRANSAKCIJE KNJIŽNICE PONOVNO UPORABNIH KOMPONENT

Procesi so navadno opredeljeni kot množice aktivnosti in opravil ali množice transakcij. Kakor posledice transakcij s knjižnico ali nad knjižnico nastajajo ali se prekinjajo različna pogodbena razmerja (članstvo, prenos lastništva, podeljevanje licenc, vzdrževanje ipd.), nastajajo stroški ali prihodki, prihaja do sprejema ali odstranitve komponent iz knjižnice ipd.

Transakcije se večinoma izvajajo elektronsko, možno pa je, da je treba kakšno transakcijo izvesti tudi ročno ali kombinirano.

Med glavne ali temeljne procese vsake knjižnice lahko štejemo vpis v knjižnico, posodobitev ali osveževanje seznama pooblaščenih uporabnikov in njihovih dostopov do virov knjižnice, izpis iz knjižnice, sprejem komponente v knjižnico, odstranitev komponente iz knjižnice, brskanje po knjižnici, iskanje komponente, poročanje uporabnikov, spremljanje stanja knjižnice in zbiranje povratnih informacij od uporabnikov knjižnice.

Glede na to, na kaj se našteti procesi nanašajo in katere aktivnosti zajemajo, sem jih razdelila v dve večji skupini in sicer procese, ki so povezani s člani knjižnice, in tiste, ki so povezani z delovanjem knjižnice. V tretjo skupino bi lahko uvrstili vse tiste procese in aktivnosti, ki so potrebni za povezovanje knjižnice z drugimi prek spleta ali za t.i. medknjižnično povezovanje. Procesi, ki so povezani z označevanjem pravnih omejitev na komponentah, obvladovanjem teh, sklepanjem pogodb ter nasploh s pravnim varovanjem intelektualne lastnine, so v modelu opisani vzporedno z drugimi procesi.



Slika 17: Procesi knjižnice ponovno uporabnih komponent.

5.4.1 Procesi, povezani s člani knjižnice

Član knjižnice je organizacija ali posameznik, ki v skladu s pogoji iz pogodbe o včlanitvi, katere sestavni del so pravila knjižnice, uporablja knjižnico, to pomeni njene komponente, orodja in storitve. Vsak član ima lahko enega ali več pooblaščenih uporabnikov, ki nastopajo v vlogi neposrednih ponovnih uporabnikov ali neposrednih razvijalcev komponent.
Temeljni trije procesi, povezani s člani knjižnice, so: vpis ali včlanitev, izpis ali prenehanje članstva in ažuriranje ali posodabljanje seznama pooblaščenih uporabnikov knjižnice ter njihovih dostopov do virov knjižnice.

5.4.1.1 Vpis v knjižnico

Član knjižnice lahko postane kdorkoli, ki želi uporabljati komponente, zbrane v knjižnici (razvijalci ali sestavljavci aplikacij), v knjižnico prispevati povsem nove komponente ali iz obstoječih komponent izpeljane komponente ali celo oboje (razvijalci ali dobavitelji komponent) in je pripravljen sprejeti koncept knjižnice ali upoštevati njena pravila delovanja. Član knjižnice torej lahko nastopa tudi v dvojni vlogi: kot razvijalec in kot uporabnik komponent[132]. Član je lahko organizacija ali posameznik. Ker so postopki, povezani z vpisom posameznika v knjižnico, le posplošitev postopkov vpisa organizacije, je v nadaljevanju opisan zgolj slednji.

Ko želi neka organizacija postati član knjižnice, se mora najprej seznaniti s pravili knjižnice. Pred pregledom pravil mora podpisati izjavo o varovanju poslovnih skrivnosti. Če se z njimi strinja ali jih je pripravljena spoštovati, poskrbi najprej za vzpostavitev varne elektronske povezave do knjižnice[133]. Nato odda pisno zahtevo po vpisu v knjižnico. Priložiti mora tudi poimenski seznam svojih delavcev, ki jih bo pooblastila za dostop do virov knjižnice (*pooblaščeni uporabniki*). Upravitelj knjižnice preveri, ali organizacija in njeni pooblaščeni uporabniki izpolnjujejo pogoje za vpis v knjižnico. Predvsem pomembno je, da preveri, ali je morda kdo od potencialnih uporabnikov med tistimi, ki so bili zaradi kršitev pravil knjižnice, povzročanja škode knjižnici ali zaradi kršitev pravic iz naslova intelektualne lastnine v

[132] Posebno pogosta oblika v manjših razvojnih organizacijah, kjer isti ljudje sodelujejo v procesu inženirstva domene in inženirstva aplikacij.
[133] Kakšna naj bo ta povezava, lahko knjižnica opredeli v svojih pravilih.

preteklosti izpisani iz knjižnice ali drugače kaznovani[134]. Če knjižnica ugotovi, da je ali so na seznamu tudi taki uporabniki, zavrne zahtevo organizacije po vpisu. Če pa upravitelj knjižnice ugotovi, da potencialni član izpolnjuje pristopne zahteve, najprej dodeli nivoje dostopov njegovim uporabnikom ter nato pošlje v podpis pogodbo o včlanitvi. S to pogodbo se član in lastnik knjižnice obvežeta, da bosta upoštevala in spoštovala pogodbena določila in pogoje. V pogodbi se določijo tudi višina in načini plačila članarine ter nivoji dostopa ali obseg dovoljenih dostopov za posamezne uporabnike knjižnice. Ko novi član vrne podpisano pogodbo, se prične postopek dodeljevanja identifikacijskih številk, gesel in računov. Najprej svojo identifikacijsko številko dobi organizacija (IDČ – identifikacijska številka člana), ki bo to številko uporabljala pri elektronskem podpisovanju licenc in drugih elektronskih transakcijah. Novemu članu knjižnica odpre tudi poseben račun, prek katerega član plačuje finančne obveznosti knjižnici ali mu nanj knjižnica nakazuje prispevke iz naslova licenčnin, nagrad ali stimulacij. Računi članov predstavljajo hkrati tudi neke vrste zbirnik transakcij člana, saj je iz njih razviden finančni "promet" člana in seznam komponent, ki jih je ponovno uporabil ali prispeval v knjižnico.

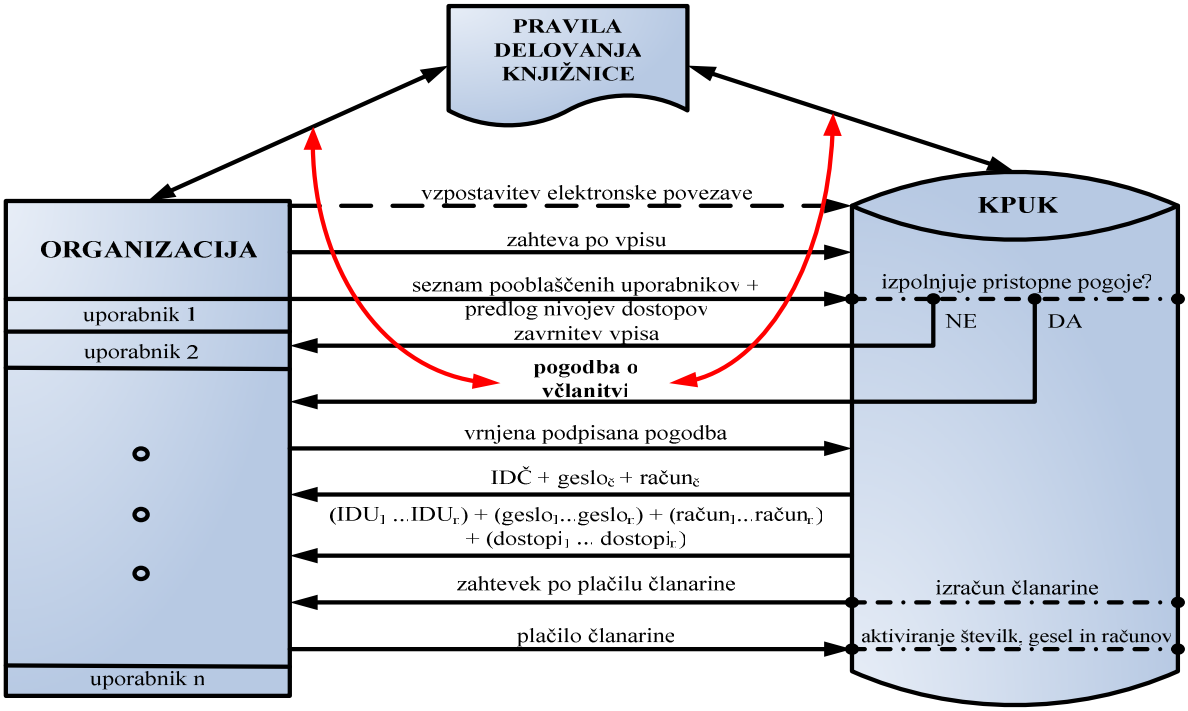
Vsakemu delavcu včlanjene organizacije, ki je na seznamu njenih pooblaščenih uporabnikov, dodeli še osebno identifikacijsko številko (IDU – identifikacijska številka pooblaščenega uporabnika), ki je izpeljanka iz IDČ, začasno geslo ter osebni uporabniški račun, poleg tega pa še morebitne omejitve dostopov do posameznih komponent ali storitev knjižnice. Postopek vpisa se konča z izračunom članarine, ki se izračuna na temelju števila pooblaščenih uporabnikov organizacije ter njihovih nivojev dostopov do vsebine knjižnice. S plačilom članarine se postopek vpisa zaključi. Celoten postopek vpisa je prikazan na sliki 18.

5.4.1.2 Proces ažuriranja seznama pooblaščenih uporabnikov

Vsak član ob včlanitvi knjižnici posreduje seznam svojih, za dostop do knjižnice pooblaščenih uporabnikov (navadno so to zaposleni iz te organizacije, ni pa vedno nujno). Ta seznam se lahko spreminja, zato ga je treba periodično ažurirati ali osveževati. Predstavnik včlanjene organizacije elektronsko posreduje knjižnici svež seznam svojih pooblaščenih

[134] Knjižnica te podatke shrani v posebni podatkovni bazi.

uporabnikov, ki so registrirani pod isto identifikacijsko številko člana (IDČ) vsakič, ko pride do spremembe. V seznam se vključijo tudi morebitne posebnosti, na primer omejitev uporabe ali dostopa do določenih komponent ali storitev knjižnice za posameznega uporabnika.



Slika 18: Proces vpisa v knjižnico ponovno uporabnih komponent.

Upravitelj knjižnice dodeli novim uporabnikom osebne identifikacijske številke (IDU) in začasna gesla, istočasno pa ukine osebne številke in gesla vseh uporabnikov, ki so izgubili status pooblaščenega uporabnika. Veljavne licence, ki so bile tem uporabnikom dodeljene, se na predlog člana prenesejo na druge pooblaščene uporabnike. Upravitelj knjižnice odpre uporabniške račune vsem novim pooblaščenim uporabnikom, zapre pa račune pooblaščenim uporabnikom, ki so bili odstranjeni s seznama.

Med procesom ažuriranja je torej treba urediti spremembe veljavnosti dostopov do posameznih virov knjižnice. Rešitev je lahko v koncept delovanja knjižnice vgrajeno pravilo, da se veljavnost dostopov in s tem obseg dostopov samodejno periodično preverjajo v nekem vnaprej določenem časovnem intervalu. Ravno tako je smiselno vgraditi mehanizem, ki bi bdel nad transakcijami uporabnikom in ko bi ugotovil, da je nek uporabnik daljši čas

neaktiven, bi samodejno predlagal njegovi organizaciji izpis tega člana ali odstranitev uporabnika z njenega seznama pooblaščenih uporabnikov.

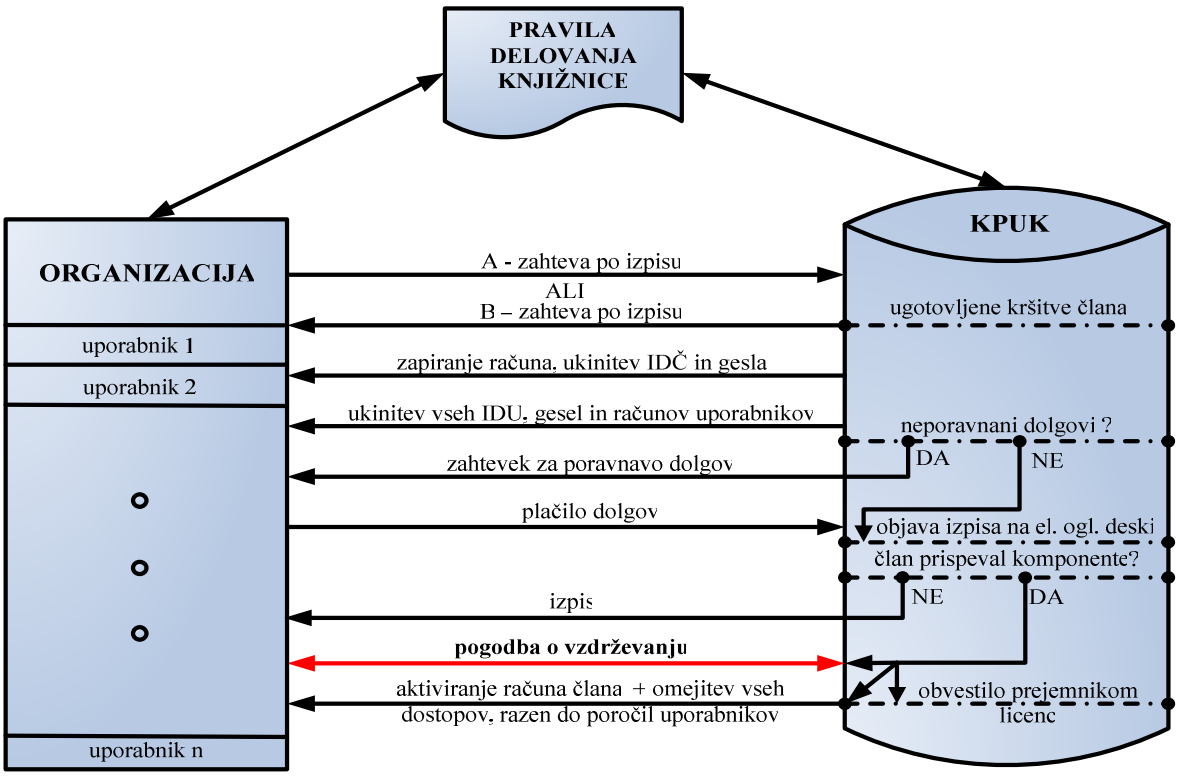
5.4.1.3 Izpis iz knjižnice

Ko nek član knjižnice (organizacija) poda zahtevo za izpis iz knjižnice, mu upravitelj najprej zapre račun ter ukine identifikacijsko številko člana in geslo, ki ju je organizacija uporabljala za elektronsko podpisovanje licenc in izvedbo vseh drugih, s pogodbo zavezujočih elektronskih transakcij. Nato ukine vse osebne identifikacijske številke in pripadajoča gesla pooblaščenih uporabnikov organizacije, ki se izpisuje, zapre pa tudi uporabniške račune in prekine vsakršen (do)tok plačil tej organizaciji. Organizacija, ki se izpisuje, je dolžna pred izpisom poravnati vse morebitne z izpisom nastale stroške (npr. oprema, komunikacija, ugotovljena škoda ipd.) ali iz preteklosti neporavnane finančne obveznosti. Upravitelj knjižnice na elektronski oglasni deski knjižnice objavi novico o izpisu člana, hkrati pa tudi identificira komponente, ki jih je ta organizacija posredovala v knjižnico ter o njenem izpisu obvesti vse uporabnike njenih komponent. Če pa se iz knjižnice izpisuje ponovni uporabnik, se z njim prekinejo vse veljavne licenčne pogodbe.

Pred izpisom se knjižnica s članom lahko še dogovori za morebitno nadaljnje vzdrževanje komponent, ki jih je član prispeval v knjižnico, če te komponente po izpisu člana ostanejo v knjižnici. V ta namen se lahko sklene poseben dogovor ali pogodba o vzdrževanju. Organizacija, ki prekinja članstvo, ohrani svoj račun, saj se nanj nakazujejo vsi prilivi iz naslova vzdrževanja, popolnoma pa se omejijo ali onemogočijo vsi drugi dostopi do komponent in storitev knjižnice, razen dostopi do poročil uporabnikov o ponovni uporabi, saj so ravno morebitne težave pri ponovni uporabi največkrat povod za vzdrževanje komponent.

Članstvo v knjižnici lahko prekine član knjižnice ali pa prekinitev članstva zahteva upravitelj knjižnice. Slednje se lahko zgodi v primeru, ko knjižnica ugotovi, da je član kršil določila in pogoje pogodbe o včlanitvi ali/in temeljne koncepte in pravila delovanja knjižnice, ali če kršitve člana povzročijo knjižnici ali njenim posameznim članom kakršnekoli stroške ali druge negativne posledice.

Veljavnosti dostopov do virov knjižnice se z izpisom seveda spremenijo. Če je lastnik knjižnice hkrati tudi lastnik komponente, ki jo je v knjižnico posredovala organizacija, ki se izpisuje[135], potem se vsi morebitni finančni prilivi preusmerijo na račun lastnika knjižnice. Ko pa je lastnik komponente organizacija, ki se izpisuje, ostanejo podlicence, ki jih je knjižnica podelila preostalim članom za te komponente[136], veljavne, dokler jih organizacija – lastnica ne razveljavi. Ob razveljavitvi podlicenc gredo vse pravice, povezane s komponentami, ki so predmet podlicence, lastniku knjižnice. Če pa se komponenta ob izpisu njenega lastnika odstrani iz knjižnice, je lastnik komponente dolžan vzdrževati veljavnost podlicenc toliko časa, kolikor je dolžina najdaljšega obdobja, za katero je bila podeljena podlicenca za uporabo te komponente. Z licencami, ki jih lastnik knjižnice podeljuje za svoje komponente, ni sprememb. Proces izpisa je prikazan na sliki 19.



Slika 19: Proces izpisa iz knjižnice.

[135] Ta organizacija je lahko po naročilu knjižnice zgradila določene komponente, za kar je knjižnica organizaciji plačala in s tem pridobila materialne pravice na komponentah. Organizacija lahko komponente knjižnici tudi proda ali nanjo prenese lastništvo na komponentah v celoti.

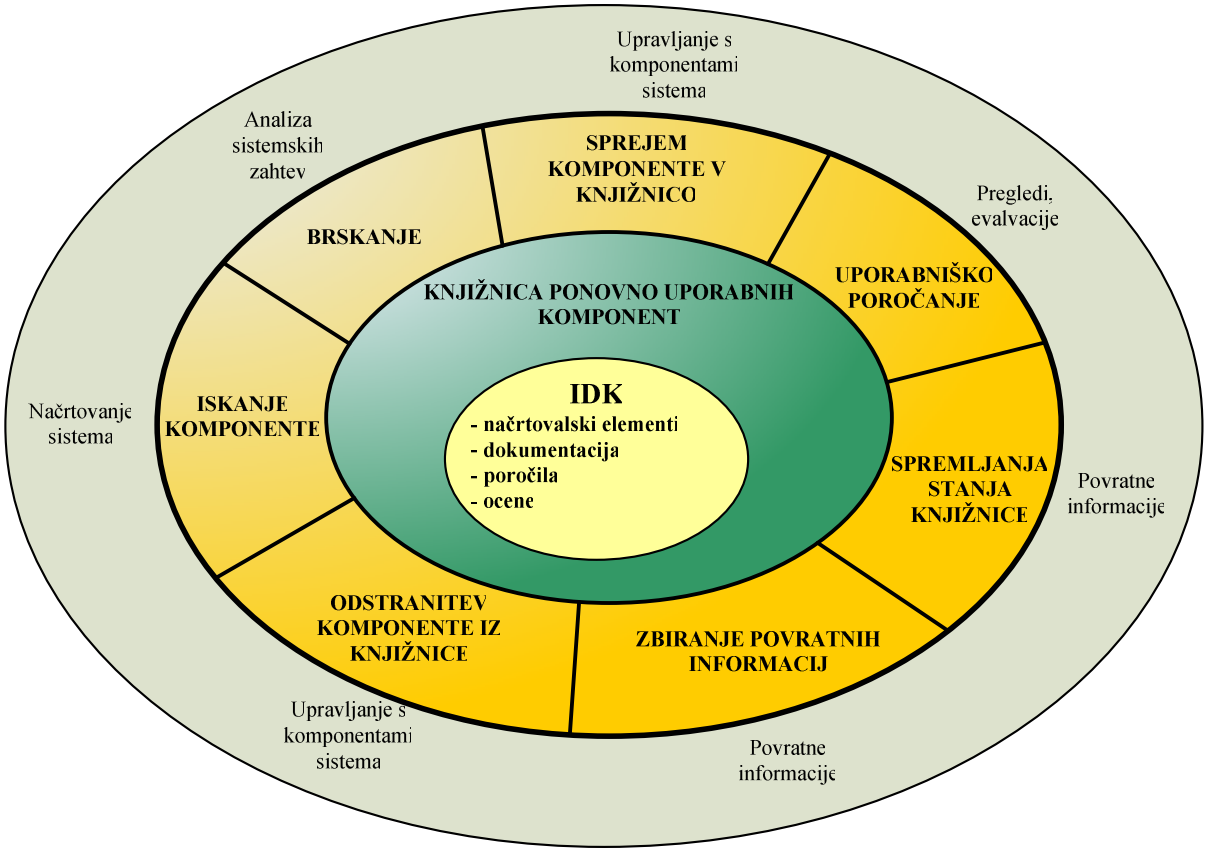
[136] Lastnik komponente ob posredovanju komponente v knjižnico z licenco dodeli tudi pravico nadaljnjega licenciranja te komponente.

5.4.2 Procesi, povezani z delovanjem knjižnice

Med temeljne procese knjižnice, ki so povezani z njenim delovanjem, sodijo:

- proces sprejemanja komponente v knjižnico,
- proces brskanja ali iskanja po knjižnici,
- proces odstranitve komponente iz knjižnice,
- proces uporabniškega poročanja o komponentah,
- proces spremljanja stanja knjižnice in zbiranja povratnih informacij od uporabnikov knjižnice.

Od naštetih procesov, ki so prikazani na sliki 20, sta v zelo veliki meri odvisni učinkovitost knjižnice in pa stopnja ponovne uporabe posameznih komponent knjižnice.



Slika 20: Procesi, povezani z delovanjem knjižnice.

5.4.2.1 Proces sprejemanja komponente v knjižnico

V knjižnico lahko posreduje ali prispeva komponente samo organizacija, ki je član knjižnice. Pri tem gre lahko za posredovanje popolnoma nove komponente ali spremenjene obstoječe komponente (različica obstoječe komponente) ali iz obstoječe komponente izpeljane nove komponente.

Razvijalec ali dobavitelj komponente[137], ki mora biti hkrati eden od pooblaščenih uporabnikov včlanjene organizacije[138], upravitelju knjižnice elektronsko posreduje razvojno in ostalo dokumentacijo, v kateri morajo biti specificirane tudi vse morebitne omejitve dostopov do komponente, ki jo predlaga za sprejem v knjižnico.

Upravitelj knjižnice najprej pregleda popolnost dokumentacije ter iz opisa komponente ugotovi, kam znotraj klasifikacijske sheme sodi komponenta. Nato s postopkom za ugotavljanje podobnosti med komponentami in orodij za ugotavljanje izpeljave iz obstoječih komponent ugotovi, ali gre za povsem novo komponento, različico obstoječe komponente ali izpeljano komponento. Novi ali izpeljani komponenti dodeli identifikacijsko številko (IDK), pregleda popolnost dokumentacije, deponira varnostno kopijo komponente v arhiv knjižnice, delovno različico pa v knjižnico, kjer čaka na oceno pred sprejemom v knjižnico (komponenta dobi oznako nepregledana). Če komponente ne spremlja ustrezna dokumentacija, se od njenega razvijalca ali dobavitelja zahteva, da jo dopolni, preden se nad komponento izvede postopek certificiranja.

Komponento nato prevzame ocenjevalec in nad njo izvede postopek ocenjevanja (certificiranje komponente). O oceni in samem postopku izdela poročilo. Na podlagi rezultatov ocenjevanja dodeli komponenti ustrezen indeks ali faktor kakovosti. Če se v postopku certificiranja ugotovi, da komponenta ne ustreza postavljenim merilom za sprejem, se zavrne. Ko je komponenta pozitivno ocenjena, jo upravitelj knjižnice skupaj z njeno

[137] Dobavitelj komponente je lahko njen razvijalec, organizacija, v kateri je razvijalec zaposlen in za katero razvija, posrednik, ki za knjižnico išče ustrezne komponente (ima z njo sklenjeno posebno pogodbo), ali posrednik, ki trži razvijalčeve komponente (neke vrste založnik).
[138] Knjižnica mora od razvijalca ali dobavitelja komponente zahtevati, da se najprej včlani v knjižnico, šele nato se prične postopek sprejema ali prevzema komponente.

dokumentacijo ter poročilom o oceni kakovosti umesti v knjižnico ali ji dodeli oznako pregledana ali preverjena. S tem omogoči ponovno uporabo te komponente. Za komponento se določijo tudi morebitne omejitve dostopa, aktivira se samodejna izdelava poročil o dostopih do komponente ter sistem za poročanje uporabnikov komponente.

Upravitelj knjižnice in dobavitelj komponente se morata dogovoriti, ali se lastništvo na komponenti iz razvijalca prenese na knjižnico (neke vrste prodaja komponente knjižnici, pri kateri razvijalcu ostanejo samo nematerialne avtorske pravice na komponenti) ali bo knjižnica odigrala samo vlogo posrednika pri licenciranju komponent. V prvem primeru bo knjižnica za komponento plačala razvijalcu odkupnino, potem pa sama podeljevala licence za to komponento drugim uporabnikom knjižnice in licenčnine zadržala. Knjižnica in razvijalec se lahko tudi dogovorita, da bo knjižnica v primeru višjih finančnih dobičkov, kakor sta jih razvijalec in upravitelj knjižnice predvidela ob določitvi odkupnine, razvijalcu periodično nakazovala dogovorjen odstotek od dobička z licenciranjem komponente, ki jo je knjižnici prodal. V pogodbi se morata tudi dogovoriti, ali lahko razvijalec, ki nastopa kakor prodajalec komponente, to komponento ponovno uporablja ali ne, in če lahko, ali mora za to pridobiti licenco od knjižnice in plačati licenčnino ali ne. V pogodbi je nujno opredeliti tudi vzdrževanje komponente. V drugem primeru pa bo razvijalec knjižnici podelil izključno licenco za komponento in ji omogočil za to komponento podeljevati podlicence drugim članom knjižnice. Knjižnica si lahko del licenčnine zadrži za kritje stroškov, preostali del pa posreduje dobavitelju komponente.

Za komponento je treba v vsakem primeru določiti še licenčnino. Na višino licenčnine v veliki meri vpliva kakovost komponente, upoštevajo pa se še celostnost, razvojni stroški in po možnosti zanesljivost izvora komponente[139]. Za določitev licenčnine knjižnica postavi formulo, po kateri jo računa. Formula je opisana v pravilih knjižnice in dostopna vsem članom. Ob predaji komponente v knjižnico se na razvijalčev uporabniški račun prenese odkupnina, če je razvijalec komponento prodal (podelil izključno licenco), sicer pa se bodo na ta račun prilivala sredstva iz naslova licenčnin za to komponento.

[139] Izvor komponente je lahko knjižnica, ki je komponento "odkupila" ali jo dala razviti po naročilu, lahko pa je komercialni prodajalec – razvijalec komponente.

Pri sprejemanju različice neke komponente, ki je že v knjižnici, so postopki preverjanja ustreznosti komponente enaki, le da je ob sprejemu take komponente treba originalno komponento, iz katere je na novo sprejeta komponenta izpeljana, ustrezno dopolniti z informacijo o obstoju različice te komponente ali celo s povezavo na različico.

Da bi preprečili kopičenje različnih različic neke komponente, ki se morebiti med seboj niti ne razlikujejo veliko, je treba ob sprejemu vsake različice ugotoviti, v kolikšni meri se nova različica razlikuje od prejšnje[140]. Če so odstopanja majhna in je funkcionalnost temeljne komponente v različici popolnoma ohranjena, je smiselno obdržati samo različico oziroma temeljno komponento nadomestiti z njeno različico, seveda ob predpostavki, da je različica tudi nadgradnja temeljne komponente. Enako velja postopati tudi v primeru, ko neko različico nadomeščamo z novo različico iste temeljne komponente. Glede na to, da dobijo različice isto IDK kakor temeljna komponenta, je smiselno postaviti pogoj, da lahko različico neke komponente v knjižnico posreduje samo avtor te temeljne komponente, ne glede na to, ali je avtor (razvijalec komponente) temeljno komponento knjižnici predhodno prodal ali ji za njo podelil samo licenco[141]. Smiselno je tudi postaviti omejitev, da mora razvijalec različice, ki je knjižnici temeljno komponento "prodal", isto storiti tudi z različico, sicer se lahko pripeti, da je pod isto IDK lastništvo mešano, kar pa ni najbolj vzpodbudno pri odločanju, ali bomo tako komponento ponovno uporabili[142]. Kvečjemu bi bila sprejemljiva še možnost, da za različico razvijalec podeli knjižnici licenco, v kateri pa se obveže, da bo različico vzdrževal, ne glede na to, ali je za temeljno komponento s knjižnico podpisal pogodbo o vzdrževanju ali ne. Sicer pa se lastništvo na različicah določa po postopkih, opisanih v 3. poglavju.

Glede sprejemanja komponent si vsaka knjižnica postavi svoja pravila. Možno je, da se knjižnica odloči za brezpogojni sprejem vseh posredovanih komponent. V tem primeru se odločanje o ponovni uporabi komponent popolnoma prepusti posameznim uporabnikom

[140] V ta namen se lahko neposredno uporabijo algoritem in orodja za ugotavljanje, ali je neka komponenta izpeljana iz neke druge, lahko pa se ti algoritmi priredijo za določanje, kdaj je neka komponenta samo različica druge komponente. Če gre za različico, se ta namreč poveže z IDK originalne komponente in postane v bistvu njen sestavni del, medtem ko postane izpeljana komponenta nova komponenta knjižnice, ki se ji dodeli enolična IDK. V tem primeru se kvečjemu v opis izpeljane komponente med lastnosti napiše tudi dejstvo, da je bila izpeljana in iz katere komponente je bila izpeljana (opisne lastnosti – podobne komponente).
[141] To pravilo je smiselno določiti že v pravilih delovanja knjižnice.
[142] Navadno je z lastništvom povezano tudi vzdrževanje komponente, ki pa je za ponovnega uporabnika zelo pomembno.

knjižnice, ki se za neko komponento odločajo predvsem na temelju faktorja kakovosti za to komponento in njene spremljajoče dokumentacije. Problem, ki se lahko pri taki politiki delovanja knjižnice pojavi, je izjemno število komponent v knjižnici. Iskanje postane težavno, v knjižnici je zelo verjetno veliko število slabih komponent. Taka knjižnica postane kaj kmalu nezanimiva za ponovne uporabnike, saj se jim lahko pogosto pripeti, da porabijo kar nekaj časa za iskanje v množici komponent, na koncu pa ne najdejo ustrezne, ne po funkcionalnosti ne po kakovosti.

Priporočljivo je torej, da si knjižnica ob začetku svojega delovanja postavi jasna merila in pravila, kakšne vrste komponent bo sprejemala. Pri tem se je modro omejiti na komponente ene domene ali kvečjemu množice sorodnih domen, znotraj nje pa na komponente, za katere lahko postavi oceno o neki zadovoljivi stopnji ponovne uporabe. Taka knjižnica ima sicer manj komponent, so pa zato bolj kakovostne. Kakovost komponente je gotovo eden temeljnih faktorjev pri določanju ponovne uporabnosti komponente in knjižnice. Ponovni uporabniki radi posegajo po komponentah take knjižnice, saj jim je v interesu dobiti čim kakovostnejše komponente. Če tako knjižnico podpremo še z dobrima mehanizmoma klasifikacije in iskanja komponent, je uspeh knjižnice že skoraj zagotovljen.

Pomen ocene kakovosti ali postopkov certificiranja komponente je torej zelo velik. Ta ocena v prvi vrsti vpliva na sprejem komponente v knjižnico ali njeno zavrnitev. Odločilna je tudi za poznejšo ponovno uporabo te komponente s strani člana knjižnice ali za njeno zavrnitev. Prav tako pa v veliki meri vpliva tudi na višino licenčnine za to komponento in posredno tudi na pogoje in določila licence ali podlicence za to komponento. Zato je pomembno, da ima knjižnica skrbno opredeljena pravila in postopek certificiranja ter določanja faktorja ali indeksa kakovosti komponente. Celoten postopek sprejemanja komponente v knjižnico je prikazan na sliki 21.

5.4.2.2 Proces brskanja po knjižnici in iskanja komponent

Temelj iskanja neke komponente je brskanje po knjižnici in njenih komponentah. Pri tem uporabnik knjižnice pregleduje posamezne elemente dokumentacije, poročila o ocenah kakovosti, poročila o dostopih, poročila uporabnikov neke komponente in preostale sestavne

dele komponente, niso mu pa na voljo načrtovalski elementi in izvorna koda ali razvojna dokumentacija. Za uporabo knjižničnih virov, ki te transakcije omogočajo, lahko knjižnica zaračuna določeno pristojbino. Sicer pa transakcije, povezane z brskanjem po knjižnici, ne ustvarjajo pretoka denarja (plačevanje licenčnin), medtem ko ga postopki iskanja ali izbire neke komponente ustvarjajo, saj je z izbiro neke komponente obvezno povezano tudi pridobivanje licence ali podlicence zanjo in plačilo ustrezne licenčnine.

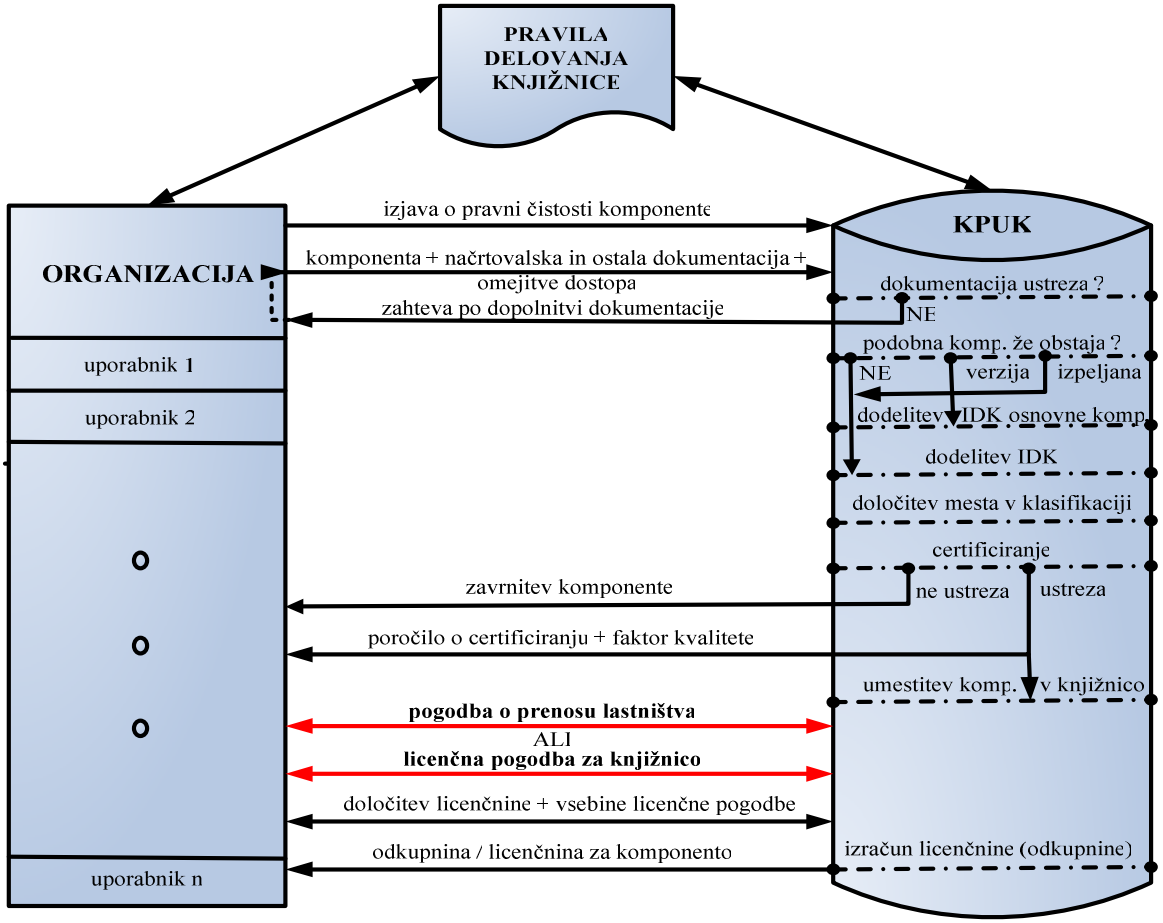
Član knjižnice navadno brska po knjižnici s ciljem, da bi našel komponento, ki jo potrebuje. Pri tem uporablja razna iskalna in odločitvena orodja, na voljo pa so mu lahko tudi razna orodja za vizualizacijo ali simulacijo delovanja komponent in prikaz njihove vsebine oziroma predstavitev njihove funkcionalnosti, če gre za neizvedljive (neizvršljive) komponente.

Knjižnica mora poskrbeti za ustrezne varnostne mehanizme in druge omejitve, s katerimi omeji procesa brskanja in iskanja. Uporabnik mora biti pred začetkom brskanja opozorjen na pravice in omejitve uporabe podatkov, do katerih bo imel med brskanjem dostop, ter na dejstvo, da kakršnakoli uporaba dokumentacijskih elementov komponente zahteva plačilo licenčnine, sicer so kršene pravice iz naslova intelektualne lastnine na komponenti. Kakor posledica kršitev lahko nastopi prekinitev članstva, ki jo sproži upravitelj knjižnice ali celo kazen, ki sledi iz zadevne veljavne zakonodaje. Pomembno je, da so skupaj z drugimi podatki o komponenti, do katerih ima uporabnik med brskanjem dostop, shranjeni tudi licenčni podatki za to komponento – določbe in pogoji licence ter višina licenčnine in drugih stroškov, povezanih s ponovno uporabo te komponente.

Ob prijavi uporabnika se samodejno preverijo njegove dostopne pravice in se mu v pregled ponudi seznam komponent, do katerih ima pravico dostopati. Ko med njimi najde želeno komponento, se sproži proces prevzema komponente iz knjižnice. V ta namen mora član posredovati informacijo o projektu (identifikacijsko številko projekta), znotraj katerega bo komponenta uporabljena ter se strinjati z določbami in pogoji iz licence[143]. Če je nujno, se ti med uporabnikom in knjižnico dorečejo ali uskladijo. Sledi plačilo licenčnine za komponento.

[143] Licenca je načeloma lahko podeljena samo za enkratno ponovno uporabo komponente ali za ponovno uporabo v samo enem izdelku ali pa knjižnica omogoči pridobitev več licenc hkrati. Predlagam, da bi vsaka licenca omogočala le enkratno ponovno uporabo, saj bi se knjižnica tako zavarovala proti nepooblaščenim uporabam licencirane komponente v drugih projektih istega ponovnega uporabnika.

Kopija komponente, skupaj z razvojno in drugo dokumentacijo, se po elektronski poti na dogovorjen, varen način (šifrirana) pošlje uporabniku – pridobitelju licence za komponento.



Slika 21: Postopek sprejemanja komponente v knjižnico.

Če želi uporabnik med iskanjem samo pregledati razvojno in drugo dokumentacijo komponente, ne želi pa ničesar uporabiti, se plačilu licenčnine izogne s podpisom ustrezne izjave. Če pa uporabnik že plača licenčnino in pozneje ugotovi, da te komponente ne potrebuje, lahko še vedno formalno zavrne uporabo najdene komponente in prekliče licenco. Stroški licenčnine, ki jo je vplačal, se mu povrnejo. Lahko pa knjižnica zadrži določen del licenčnine zaradi nastalih stroškov.

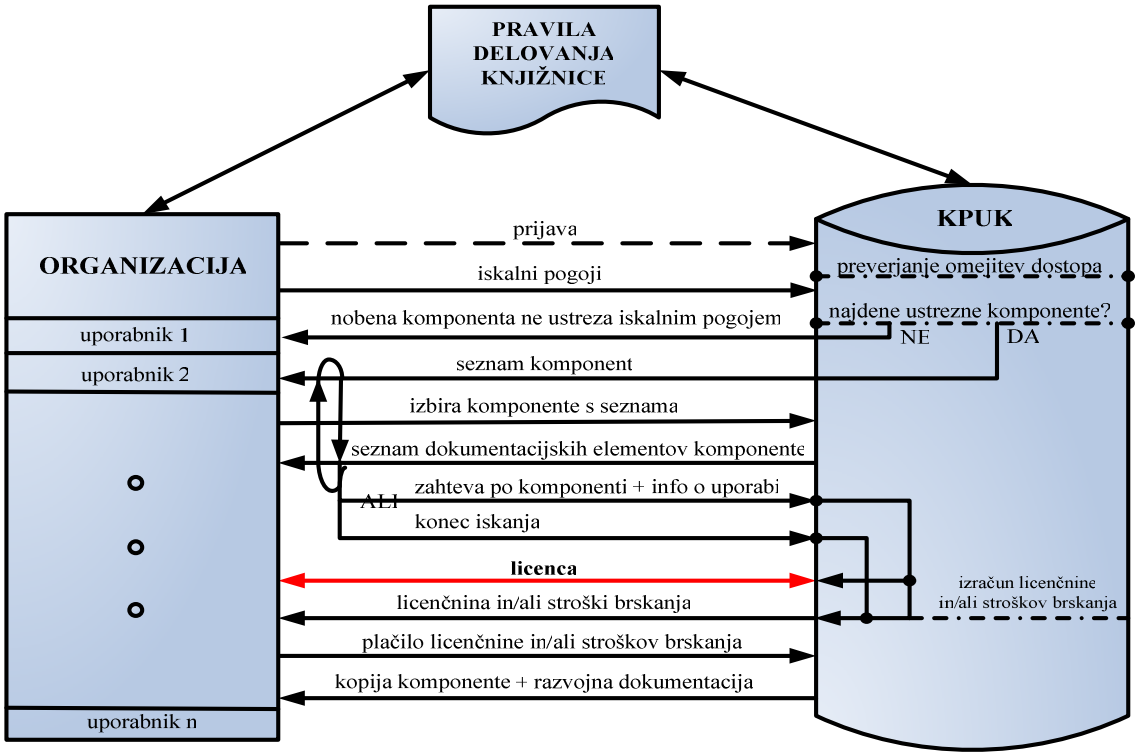
5.4.2.3 Proces odstranitve komponente

Do odstranitve komponente iz knjižnice pride lahko iz različnih razlogov, zahtevata pa jo bodisi knjižnica bodisi njeni uporabniki bodisi razvijalec komponente. Glavni razlogi so navadno v nepravilnostih, odkritih napakah in škodljivi logiki ter nenatančnostih, povezanih s komponento, ki se ugotovijo med pregledovanjem razvojne ali preostale spremljajoče dokumentacije, ko je bila komponenta že sprejeta v knjižnico, ali pa med njeno ponovno uporabo.

Upravitelj knjižnice komponento in njeno varnostno kopijo označi za odstranitev ter na elektronsko desko izobesi ustrezno obvestilo o odstranitvi komponente. Obvestilo razpošlje tudi vsem uporabnikom, ki so za to komponento plačali licenčnino. V obvestilu navede razloge, ki so privedli do odstranitve komponente, ter informacije o morebitnem vračilu licenčnine. O slednjem odloča programski svet knjižnice, katerega temeljne naloge so oblikovanje pravil delovanja knjižnice ter varnostnih mehanizmov knjižnice.

Odstranitev komponente iz knjižnice lahko privede do začasne ali stalne prekinitve oziroma preklica licence. Začasna prekinitev bi nastopila, če bi se knjižnica odločila, da bo razloge za odstranitev komponente odpravila, potem pa popravljeno komponento brezplačno posredovala vsem uporabnikom, ki so zanjo pridobili licenco in plačali licenčnino. Za čas prekinitve se veljavnost licence (samodejno) podaljša. Če je imetnikov licence za to komponento veliko in je licenčnina visoka, je knjižnici začasna prekinitev licence gotovo v interesu. Če pa knjižnica ugotovi, da bi bili stroški z odpravo napak previsoki ali morda, da komponenta sploh še ni licencirana ali je pridobiteljev licence za to komponento malo, je smiselno komponento odstraniti. Ker se lahko zgodi, da so določeni uporabniki komponento, ki je v postopku odstranitve iz knjižnice, že uporabili pri gradnji svoje aplikacije, je koristno, da se v licenci ali podlicenci za komponente določijo tudi ravnanja v tem primeru, še posebno, če so razlogi za odstranitev komponente ponovnemu uporabniku že povzročili (materialno in/ali moralno) škodo.[144]

[144] Posledice nepravilnega delovanja komponente so nepravilno delovanje aplikacije, v katero je bila komponenta vgrajena, nastanek stroškov za razvijalca aplikacije z odpravo napak ter nezadovoljstvo in morda celo nezaupanje uporabnikov aplikacije do njenih razvijalcev.



Slika 22: Proces brskanja po knjižnici in iskanja komponente.

5.4.2.4 Proces uporabniškega poročanja o komponentah

Zbiranje povratnih informacij od uporabnikov komponent je za knjižnico zelo pomembno, saj se na temelju teh informacij knjižnica odloča o svojem nadaljnjem razvoju in izboljšavah. Povratne informacije lahko pošlje katerikoli uporabnik, tisti, ki po knjižnici samo brska, ali tisti, ki dejansko komponente tudi ponovno uporabi oziroma zanje pridobi licence. Zaradi lažjega zbiranja, shranjevanja in obdelave teh informacij je dobro, če knjižnica vnaprej pripravi enotne obrazce ali vzorce[145], prek katerih lahko uporabniki po elektronski poti pošiljajo informacije.

Knjižnici je seveda v velikem interesu pridobiti čim več povratnih informacij o komponentah, zato mora svoje uporabnike ustrezno motivirati, da svoje pripombe, predloge in mnenja posredujejo knjižnici. Predlagam, da knjižnica v ta namen vzpostavi sistem denarnega

[145] Spletni obrazci.

nagrajevanja ali neke vrste finančne stimulacije uporabnikom, ki so knjižnične komponente ponovno uporabili in o svojih izkušnjah poročali knjižnici. O višini teh stimulacij ali morebiti popustov pri nadaljnjih ponovnih uporabah naj bi odločal programski svet knjižnice, kot temeljna merila pa naj bi vzel izčrpnost, celovitost in popolnost poročil.

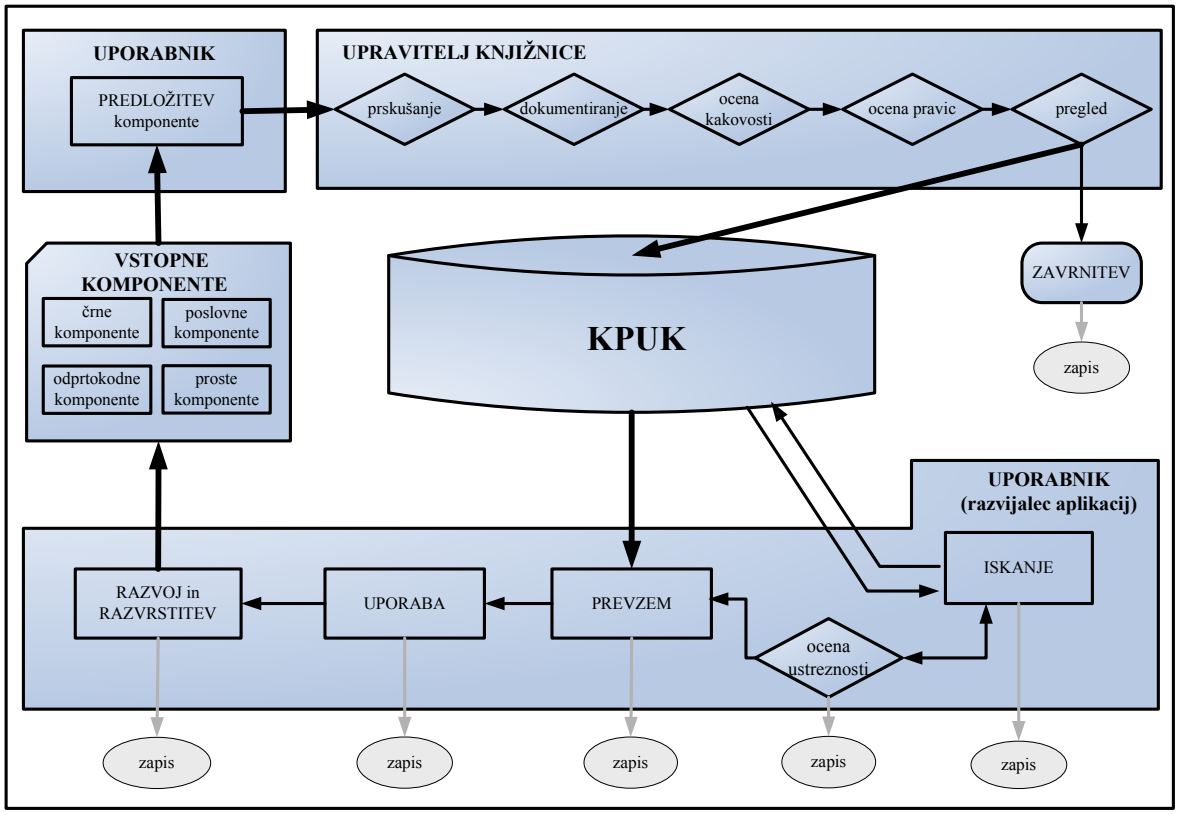
Če naj bo poročilo celostno in za knjižnico uporabno, mora vsebovati več elementov. Kot zelo pomembne ocenjujemo stroške, ki jih je imel uporabnik knjižnice s ponovno uporabo komponente, poročilo o morebitnih ugotovljenih pomanjkljivostih ali napakah komponente ali njene dokumentacije, poročilo o težavah pri ponovni uporabi komponente ter podatke o oceni komponente, ko je bila ponovno uporabljena. V teh poročilih gre za ovrednotenje komponente z vidika ponovnega uporabnika. Iz vseh teh poročil lahko knjižnica pridobi pomembne informacije o delovanju komponente med njenim integriranjem v novo aplikacijo ter njenim vedenjem v novem okolju, kar lahko posledično vpliva tudi na dvig ali spust licenčnine za komponento. Licenčnina je namreč v veliki meri odvisna od ocene kakovosti, poročila o pomanjkljivostih komponente pa lahko to oceno znižajo. Da bi se knjižnica izognila lažnim poročanjem uporabnikov v pridobitniške namene, je koristno, če vzpostavi nek mehanizem preverjanja resničnosti in verodostojnosti poročil ter uporabnike seznani, da bodo v primeru krivega pričanja ali posredovanja zavajajočih informacij ustrezno kaznovani. Koristno pa je tudi, če ima knjižnica izoblikovana merila o stimulativnem nagrajevanju posredovanja povratnih informacij ob ponovni uporabi komponente iz knjižnice. Ta merila so sestavni del pravil knjižnice.

Pridobljene in preverjene informacije o komponenti se povežejo s komponento v knjižnici in z njeno preostalo dokumentacijo, da so na voljo vsem drugim uporabnikom, ki bodo brskali in iskali po knjižnici. Povratna sporočila uporabnikov v povezavi z ustreznimi metrikami in statistikami uporabe omogočajo pridobitev ocene kakovosti knjižnice (ustreznost njene vsebine, kakovosti njenih komponent, enostavnost ponovne uporabe njenih komponent), njene ponovne uporabnosti, posredno je mogoče oceniti ali izmeriti tudi uspešnost ponovne uporabe (stroškovno učinkovitost programa ponovne uporabe).

5.4.2.5 Proces spremljanja stanja knjižnice in zbiranja povratnih informacij

Poleg informacij o komponentah knjižnica nujno potrebuje tudi informacije o svojem delovanju in storitvah, ki jih ponuja. Nekatera povratna poročila o spremljanju uporabe knjižnice se lahko izdelajo samodejno, za kar mora poskrbeti knjižnični sistem oziroma ustrezna orodja, preostala pa izdelajo in posredujejo člani knjižnice. Poročila lahko vsebujejo navedbo ali opis problemov, izkušenj, napotkov ali nasvetov, ki se nanašajo na orodja knjižnice, knjižnične postopke, sisteme poročanja ali upravljanje knjižnice.

Knjižnični sistem mora samodejno slediti uporabi knjižničnih virov in voditi dnevnik uporabe knjižnice. V dnevniku uporabe naj bi bile shranjene informacije o dostopih do komponent in informacije o uporabi knjižnice kakor to prikazuje slika 23.



Slika 23: Sledenje in zapisovanje uporab knjižnice in knjižničnih virov.

Informacije o dostopih do komponent se podajo s podatki o tem, kdo je do komponente dostopal, znotraj katerega projekta[146] naj bi bila komponenta ponovno uporabljena, čas dostopa do komponente, razlog dostopa do komponente[147], vsa plačila, povezana s komponento, ter statistika dostopov do komponent. Zapisi in statistike, povezani z dostopi do komponent, morajo biti na voljo vsem uporabnikom že med brskanjem po knjižnici.

Informacije o uporabi knjižnice pa se nanašajo na podatke o tem, kdo do knjižnice dostopa, kdaj in zakaj, do katerih komponent ter o vseh plačilih, povezanih z dostopi. Do teh informacij in statistik lahko dostopa samo osebje knjižnice. S tem, ko knjižnica vzpostavi sistem pridobivanja informacij o svojem stanju in delovanju, pokaže pripravljenost, da bo morebitne napake sproti odpravljala.

5.4.3 KLJUČNI FAKTORJI USPEHA KNJIŽNICE PONOVNO UPORABNIH KOMPONENT

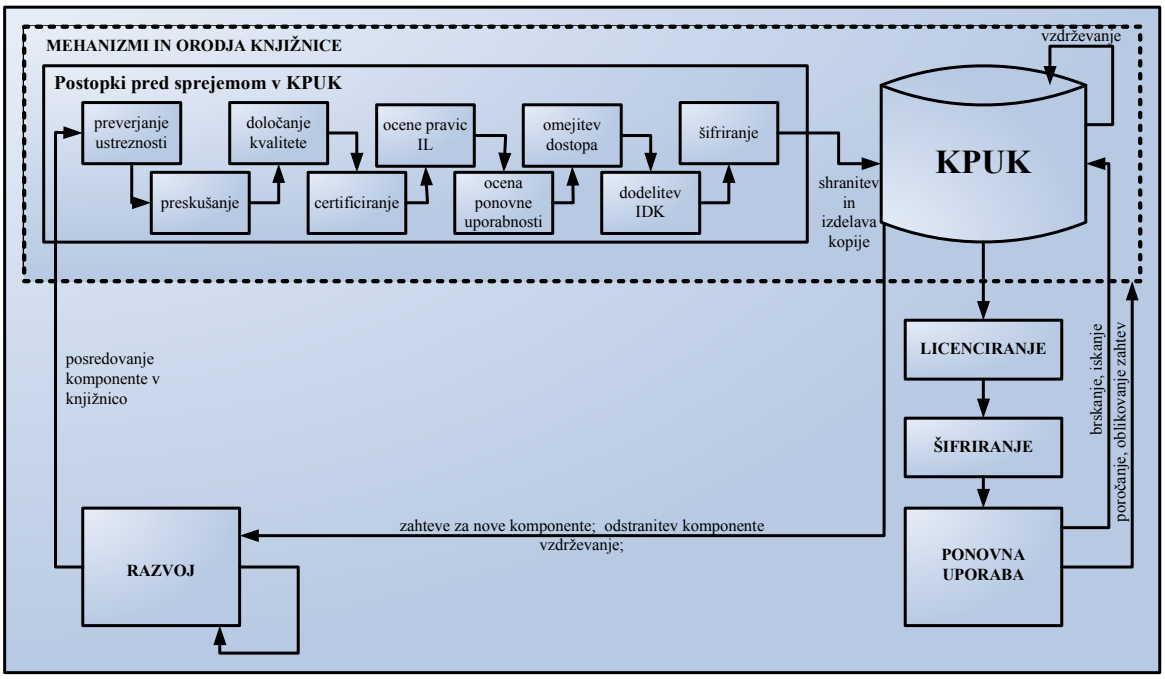
Uspešnost knjižnice, ki jo lahko opredelimo kot uspešnost njenega delovanja na eni ter uspešnost njene uporabe v smislu množične uporabe njenih virov na drugi strani, je odvisna od zelo veliko faktorjev.

- Knjižnica se mora omejiti na eno domeno ali nekaj zelo tesno povezanih, torej sorodnih domen.
- Kot temeljno za svojo strukturo mora uporabiti modele domene, ki so nastali kakor rezultat analize domene.
- Pri izbiranju komponent mora knjižnica upoštevati potrebe ponovnih uporabnikov, to so razvijalci aplikacij domene.
- Zgraditi mora take mehanizme in postopke sprejemanja komponent, ki bodo dovolili vstop v knjižnico samo komponentam, ki bodo zadoščale neki množici vstopnih pogojev, kot so opremljenost z dokumentacijo, jasno lastništvo, dovolj visok faktor kakovosti, relevantnost komponente za knjižnico, njena ponovna uporabnost (slika 24).

[146] Ali projektov, če je bilo za komponento danih več licenc.
[147] Razlog je lahko samo informativno brskanje ali pa pregledovanje vsebine, ki se zaključi z licenciranjem.

- Uporabnikom mora ponujati učinkovite in varne mehanizme brskanja, iskanja, pregledovanja vsebine komponent, njihove dokumentacije, pošiljanja povratnih informacij, licenciranja itn.
- Opremljena mora biti s čim večjo množico različnih orodij, ki povečujejo učinkovitost delovanja knjižnice, dostopnost do njenih virov in ponovno uporabo komponent.
- Temeljna pogodba med člani knjižnice in knjižnico morajo biti jasno in nedvoumna napisana pravila knjižnice, v katerih je natančno opisano njeno delovanje ter pravila obnašanja in ki so vsebinsko dobro strukturirana, da omogočajo visoko stopnjo preglednosti.
- Posebno pozornost mora knjižnica nameniti mehanizmom sledenja pravicam na komponentah, mehanizmom podeljevanja licenc, prenosa lastništva in vsega drugega, kar lahko še privede do kršitve pravic intelektualne lastnine.
- Posebno pozornost je pri postavitvi knjižnice treba nameniti tudi uporabniškemu vmesniku, s katerim se knjižnica predstavlja svojim uporabnikom in ki predstavlja vrata v knjižnico. Uporabniški vmesnik mora biti preprost, uporabniku pa mora tudi vzbuditi občutek uporabnosti in relevantnosti vsebin, ki so "skrite" za njim. Poleg tega, kako so informacije na zaslonu predstavljene, torej podoba vmesnika (barve, ozadje, postavitev in razporeditev objektov na zaslonu, delež grafike, predstavitev gumbov in ikon idr.), sta zelo pomembni preprostost in razumljivost uporabljene terminologije, ki mora biti konsistentna skozi celoten sistem, usklajena s pojmovnikom (besednjakom) ali slovarjem knjižnice. Glede na bogatost z vsebinami je treba posebno pozornost nameniti navigaciji po knjižničnem sistemu oziroma preprostemu premikanju po sistemu. Z naraščanjem količine informacij narašča namreč tudi kompleksnost strukture shranjevanja teh informacij, zato se lahko obiskovalec knjižnice kaj hitro izgubi. Posebne funkcije uporabniškega vmesnika lahko poskrbijo, da uporabnika stalno obveščajo o njegovem trenutnem "položaju" in prehojeni poti do tega. Od uporabniškega vmesnika je v veliki meri odvisna tudi hitrost iskanja informacij.
- Dostopnost do knjižnice mora biti čim večja. Če bo knjižnica dostopna prek spleta, potem uporabniki za njeno uporabo ne potrebujejo nobene posebne strojne in programske opreme, kar bistveno poveča in olajša možnost njene uporabe. Poleg tega pa je tak vmesnik tudi preprostejši, uporabnikom je okolje povsem znano in ga dobro obvladujejo.

- Potencialni uporabniki knjižnice morajo za knjižnico (iz)vedeti, zato mora knjižnica poskrbeti za ustrezno promocijo, oglaševanje, poskusne dostope, obveščanje razvijalcev o svojem obstoju.



Slika 24: Življenjski cikel ponovno uporabne komponente, ki ga mora knjižnica podpreti.

- Uporabniki morajo uporabo knjižnice čim prej usvojiti, zato mora knjižnica poskrbeti za ustrezno izobraževanje svojih sedanjih in prihodnjih članov.
- Pri gradnji knjižnice je treba vnaprej predvideti možnost njenega sodelovanja z drugimi knjižnicami oziroma njeno medknjižnično povezovanje, saj je to lahko ključnega pomena za njen obstoj, nadaljnjo rast in razvoj. Torej za temeljni podatkovni model knjižnice izberemo takega, ki ga je možno nadgraditi ali ki že podpira izmenjavo informacij med sodelujočimi knjižnicami.

5.5 MEDKNJIŽNIČNO POVEZOVANJE

Še pred dobrim desetletjem [60] so bile vse knjižnice ponovno uporabnih komponent internega značaja, to pomeni samostojne, omejene največkrat na projektno skupino ali kvečjemu razvojno organizacijo in se med seboj niso povezovale. Vsaka je uporabniku

omogočala dostop do svoje vsebine prek svojega uporabniškega vmesnika, ki je bil sestavni del knjižnice. Če so želeli uporabniki uporabljati več knjižnic, so se morali naučiti uporabljati več uporabniških vmesnikov. Katalogi, v katerih so bile zbrane in opisane komponente knjižnice, so se nanašali samo in izključno na komponente, ki jih je ta knjižnica tudi fizično hranila (v istem računalniškem sistemu kakor katalog). Uporabniški vmesnik in katalog sta bila edina smerokaza do izdelkov in storitev knjižnice.

Z razširjeno uporabo spleta, spletnih storitev in brskalnikov se je koncept knjižnic ponovno uporabnih komponent nekoliko spremenil. Pojavila se je potreba po medsebojnem sodelovanju teh knjižnic in s tem po standardizaciji postopkov, ki se izvajajo pri povezovanju. Leta 1991 je bila zaradi tega, da bi se postavili standardi, ki bi omogočali medsebojno povezavo in skupno delovanje različnih digitalnih knjižnic, ustanovljena skupina RIG (ang. *Reuse Interoperability Group*), ki danes vključuje številne predstavnike tako vladnih in akademskih knjižnic kakor tudi knjižnic v podporo industriji programske opreme [61]. S standardi naj bi se odpravile vse dotedanje pomanjkljivosti izoliranih internih knjižnic. Standardi naj bi izhajali iz naslednjih predpostavk:

- sredstvo za medsebojno povezovanje knjižnic naj bi postal splet,
- uporabniški vmesnik naj bi bil neodvisen od knjižničnega kataloga (za brskanje po knjižnicah se lahko uporablja katerikoli spletni brskalnik),
- katalogiziranje komponent naj bi bilo neodvisno od shranjevanja komponent (v katalogu so lahko tudi komponente, ki so fizično shranjene v drugem računalniku, v drugi knjižnici),
- zgraditi je treba abstraktni (univerzalni) podatkovni model, ki bi omogočal izmenjavo informacij[148],
- mreža med seboj povezanih knjižnic naj bi ustvarila spletno tržišče komponent, na katerem se srečujejo in sodelujejo uporabniki, dobavitelji, razvijalci komponent in aplikacij,
- postopki plačevanj, prenosov tržne programske opreme prek spleta in drugi postopki shranjevanja informacij za sledenje ponovnim uporabam knjižničnih komponent naj bi se izvajali rutinsko.

[148] HTML predpisuje samo model spletnih dokumentov, ne pa predstavitvene oblike dokumentov; do vsebine dokumentov je mogoče priti s poljubnim brskalnikom, grafičnim ali ne grafičnim.

Prednosti medsebojnega (so)delovanja knjižnic ponovno uporabnih komponent so številne, čutijo pa jih lahko uporabniki knjižnic in družbe nasploh. Uporabniki knjižnic lahko do izdelkov in storitev različnih knjižnic dostopajo prek enega vmesnika, knjižnicam ni treba shranjevati redundantnih vsebin, ampak lahko uporabnika do komponente, ki je shranjena v drugi knjižnici, preusmerijo s hiperpovezavo. Funkcije, ki jih zagotavljajo različne knjižnice, so na razpolago kateremukoli uporabniku, ustvarjajo tržne niše za podjetnike in omogočajo obstoj industrije ponovne uporabe.

Ključni element, ki uporabniku ene knjižnice omogoča dostop do vsebine druge knjižnice, je abstrakcija podatkov, ki naj bi se pri tem sodelovanju izmenjali. Ker nista pomembni niti konkretna oblika niti metoda predstavitve teh podatkov, mora biti abstraktni podatkovni model[149] neodvisen od izbire specifičnega protokola[150], kar pomeni, da je mogoče do virov različnih knjižnic dostopati prek različnih tipov omrežij in omrežnih protokolov ter neodvisno od konkretne predstavitve podatkov[151] [62]. Rezultat izvedbe takega abstraktnega modela ponuja uporabniku možnost, da do katalogov različnih knjižnic ponovno uporabnih komponent dostopa prek skupnega vmesnika, kakršen je spletni brskalnik.
Še en pomemben cilj, ki naj bi ga bilo s sodelovanjem knjižnic moč doseči, je zmanjšanje redundantnih pojavitev istih ali zelo podobnih programskih izdelkov znotraj različnih knjižnic. Če knjižnice lahko sodelujejo z izmenjavo kataloških zapisov in celo zrcalijo datoteke med seboj, potem lahko uporabnik katerekoli knjižnice dostopa do storitev in izdelkov, ki jih knjižnice ponujajo. Tako knjižnicam ni treba hraniti kopij istega izdelka. V tem primeru je za sodelujoče knjižnice bolje, da se vsaka osredotoči na eno (svojo) aplikacijsko domeno ali eno družino uporabnikov, za druge programske izdelke pa v katalogu preskrbi ustrezne povezave oziroma kazalce na knjižnice, ki jih hranijo.

Vsaki komponenti se ob sprejemu v knjižnico (četudi interno) dodeli enolična identifikacijska številka. Tudi v primeru povezovanja knjižnic med seboj se ponovno uporabnim komponentam dodeli enolična globalna oznaka. Tako ima vsaka komponenta, ne glede na svojo fizično lokacijo, enolično oznako. Vsaka sprememba komponente povzroči dodelitev

[149] Podatkovni model na abstrakten način opisuje informacije, ki naj bi se med knjižnicami izmenjale.
[150] Protokol opisuje način pogovarjanja med dvema omrežnima vozliščema pri izmenjavi podatkov.
[151] Uporabljen format opisuje, kako je primerek podatkovnega modela dejansko predstavljen.

novega identifikatorja. Na tak način je mogoče rešiti problem podvajanja in vzdrževati nadzor nad različicami istega izdelka med sodelujočimi knjižnicami.

Za medsebojno sodelovanje knjižnic pa ni pomemben samo abstraktni podatkovni model. Ta opisuje minimalno množico informacij, ki naj bi si jih knjižnice izmenjevale o komponentah, da bi bilo medsebojno (so)delovanje mogoče. Problem, ki ga je treba še rešiti, je različnost meril ocenjevanja, preskušanja in certificiranja komponent ter upravljanja s pravicami na izdelkih knjižnice, ki si jih vsaka knjižnica postavi zase in so odvisni od oziroma pogojeni s temeljnim poslanstvom posamezne knjižnice. Za učinkovito sodelovanje različnih knjižnic je treba poskrbeti tudi za vrsto drugih informacij, ki naj bi se izmenjevale, zato mora biti abstraktni podatkovni model tak, da se lahko nadgradi.

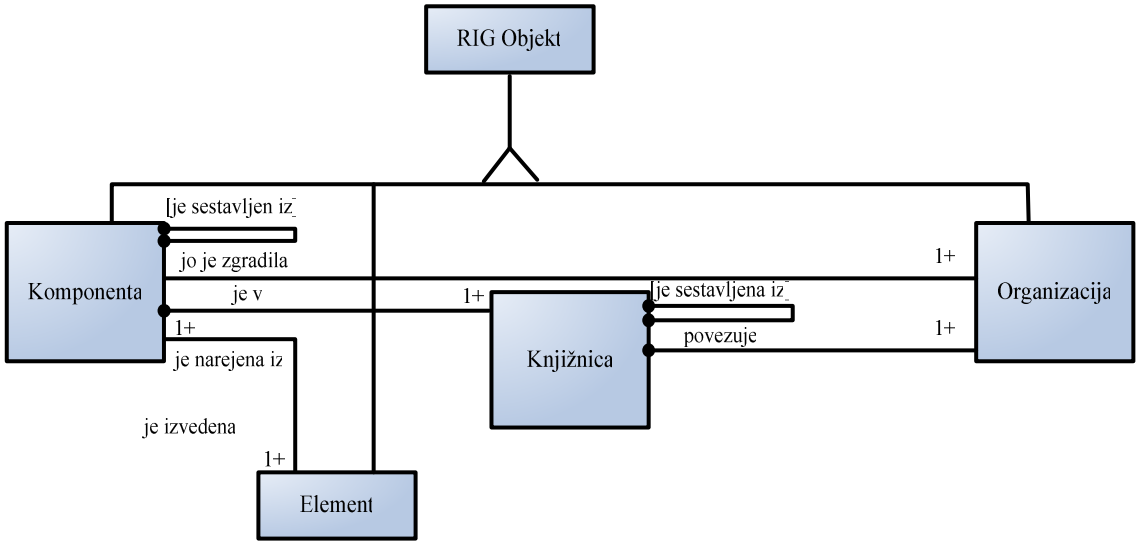
5.5.1 Temeljni podatkovni model za medknjižnično sodelovanje

BIDM (ang. *Basic Interoperability Data Model*) je bil leta 1995 sprejet, leta 2002 pa ponovno potrjen kot standard IEEE (1420.1)[152] za knjižnice ponovno uporabnih komponent. Model določa minimalno množico metapodatkov ali informacij o komponentah, ki naj bi si jih bile knjižnice ponovno uporabnih komponent sposobne izmenjevati med sodelovanjem. Izražen je v obliki razširjenega entitetno-relacijskega podatkovnega modela. Ponovno uporabne komponente (ponovno uporabne entitete), posamezni elementi, ki sestavljajo komponento (datoteke), knjižnice s komponentami in organizacije, ki razvijajo in vzdržujejo knjižnice ter komponente, so v modelu predstavljeni kot razredi. Na sliki 25 je podatkovni model BIDM grafično predstavljen s pomočjo Rumbaughjeve notacije[153] [62].

Model spremlja dokumentacija [62], v kateri so podrobno opisani vsi razredi, atributi in medsebojne povezave. Atribut, ki omogoča, da najdemo podatke o nekem objektu ali kar objekt sam, je enolični identifikator. Globalna enoličnost identifikatorja se nanaša na dejstvo,

[152] IEEE – Institute of Electrical and Electronic Engineers. [62]
[153] Rumbaugh J., Blaha M., Premerlani W, Eddy F., Lorensen W. Object-Oriented Modeling and Design. Prentice-Hall Inc. New Jersey. 1991

da različna objekta ne smeta imeti istega identifikatorja, četudi sta fizično v ločenih knjižnicah[154].



Slika 25: Shema temeljnega podatkovnega modela za medknjižnično sodelovanje.

5.5.1.1 Standardizirani razširitvi modela BIDM

Razširitve temeljnega podatkovnega modela omogočajo natančnejšo opredelitev določenih področij ali zahtev specializiranih knjižnic. Dve od področij, ki sta za vsako knjižnico izredno pomembni, sta področje certificiranja in ocenjevanja komponent ter področje pravic intelektualne lastnine na komponentah, zato sta ti dve področji podrobneje opisani v prilogi C s posebnima razširitvama temeljnega modela, ki tudi veljata za standarda.

5.5.1.2 Nadaljnje razširjanje temeljnega podatkovnega modela

Glede na to, da model BIDM ni obvezujoč za posamezne interne knjižnice, in glede na različne interese posameznih knjižnic, ki uporabljajo BIDM kakor temeljni podatkovni model, se lahko pojavijo različne tendence po razširjanju tega modela. Da bi se uredilo in poenotilo način razširjanja, da bi se razširjanje izkoristilo kot nadgradnja temeljnega podatkovnega

[154] Večina izvedb tega modela za dodeljevanje identifikatorjev uporablja kar naslove URL (Uniform Resource Location) [61].

modela oziroma za zapolnitev vrzeli v modelu, bi bilo smiselno definirati nov standard, ki bi urejal področje. Standard bi bil lahko neke vrste formalni meta model, ki bi določal način ali metodologijo razširjanja temeljnega modela in bi bil razumljiv (tudi) inteligentnim posrednikom, ki bi znali samodejno interpretirati in obdelati metapodatke iz temeljnega modela in njegovih razširitev.

Področje, ki je gotovo zanimivo in ga temeljni model ne definira, je izrazoslovje knjižnic. Zaželeno in koristno je namreč, da sodelujoče knjižnice za sezname ključnih besed, taksonomijo in slovarje uporabljajo poenoten besednjak, ki bistveno olajšajo uporabo knjižnic ter seveda ponovno uporabo komponent. Menimo, da je poenoteno poimenovanje ključno za medsebojno sodelovanje knjižnic, ob predpostavki seveda, da gre za medknjižnično povezovanje znotraj ene domene ali več sorodnih domen, kjer je tako poenotenje mogoče. Poimenovanje ali izrazoslovje mora biti usklajeno z izrazoslovjem domene, v katero knjižnice sodijo.

5.5.2 Možnosti medsebojnega povezovanja knjižnic

S povezovanjem konkretnih knjižnic ponovno uporabnih komponent zgradimo t.i. *virtualno ali navideznoresnično knjižnico*. Knjižnice so uporabnikom dostopne prek skupnega vmesnika – skupne ali izhodiščne spletne strani. Ta vedno ponuja temeljne informacije o knjižnicah, ki jih združuje, o njihovih komponentah, pa tudi pravila uporabe virtualne knjižnice. Dejansko predstavlja vstopno točko v spletno knjižnico. Stran lahko postavita ena od knjižnic tržišča ali neodvisni posrednik, ki za to knjižnicam zaračunava uporabnino in igra vlogo upravitelja spletnega tržišča.

Izhodiščna stran lahko odigra samo funkcijo vstopne točke do posameznih knjižnic[155], kar pomeni, da se uporabnik prek hiperpovezave poveže na spletno stran konkretne knjižnice. Pri iskanju in pregledovanju je omejen samo na to knjižnico. V bistvu predstavlja taka izhodiščna stran samo skupek informacij in povezav. Po drugi strani pa lahko izhodiščna stran

[155] Knjižnice dobijo ob vstopu v spletno tržišče svoje identifikacijske številke (IDKpuk), ki enolično določajo knjižnico. Pri morebitni izmenjavi komponent med knjižnicami se IDK kot predpona doda IDKpuk. Tako je komponenta ponovno enolično določena. Označevanje knjižnic z enoličnimi identifikatorji je koristno tudi pri zbiranju in shranjevanju informacij o transakcijah posameznih knjižnic.

uporabniku ponuja tudi katalog komponent ne glede na njihovo fizično mesto. V katalogu so lahko samo opisi komponent, ki so fizično shranjene na oddaljenih računalnikih in nanje kažejo samo kazalci. Prek izhodiščne spletne strani (skupnega vmesnika), ki ima vgrajen iskalnik po katalogih spletnih knjižnic, pa je mogoče tudi dostopati do opisanih komponent.

Sodelujoče knjižnice si torej lahko programsko opremo delijo na dveh nivojih:

1. na nivoju kataloga informacij, ki samo opisuje komponente knjižnice,
2. na nivoju dejanskih (programskih) datotek.

Prednosti takega neposrednega razširjanja ali razpečevanja komponent (slika 26) so predvsem hitrejše in zanesljivejše posredovanje komponent in storitev uporabnikom, poleg tega pa skupni vmesnik predstavlja tudi skupno stično točko za določene administrativne postopke, kakršen je npr. licenčni dogovor. Problemi, ki jih neposredno razširjanje prinaša, so povezani predvsem z odgovornostmi pri uveljavljanju pravnih omejitev na komponentah, z zaupanjem v prenos komponent (programskih datotek) in statistiko uporabe izhodiščne strani, ki je lahko temelj za izračun stroškov posamezne knjižnice za vzdrževanje skupnega vmesnika.

Viri virtualne knjižnice ter vzdrževanje teh virov so torej porazdeljeni po posameznih knjižnicah. Dejstvo, da so informacije o vzdrževanju tesno povezane z viri, predstavlja prednost (lokalni nadzor in lokalno osveževanje), poraja pa tudi probleme tako pri učinkovitosti kakor zanesljivosti dostopa oddaljenih uporabnikov do teh virov, predvsem v primerih, ko knjižnice ne izvajajo poenotenih, standardiziranih postopkov.[156]

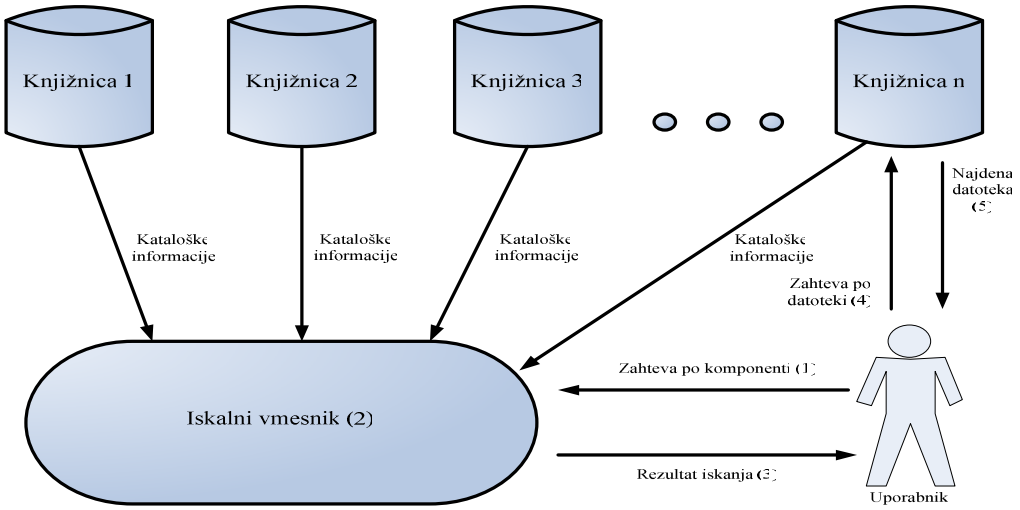
5.5.3 Izhodišča za gradnjo spletnega tržišča

V luči koncepta knjižnice ponovno uporabnih komponent, ki sem ga predstavila v tem poglavju, in glede na dejstvo, da je povezovanje v virtualne ali spletne knjižnice ali komponentna tržišča za sodobno knjižnico nujno, predlagam gradnjo arhitekture spletne knjižnice na naslednjih izhodiščih:

- v komponentna tržišča ali spletne knjižnice se združujejo samo knjižnice iz iste ali sorodnih domen;

[156] Probleme z dostopi je sicer možno reševati s podvajanjem in skrivanjem virov, vendar pa to poleg težav s konsistentnostjo in osveževanjem zrcalnih kopij povzroča tudi številne probleme s pravicami intelektualne lastnine.

- idealno bi bilo, če bi se knjižnice uskladile v vseh postopkih, ker pa je to v praksi zaradi različnih interesov neizvedljivo, menimo, da bi bila nujna uskladitev vsaj v zahtevah, ki se nanašajo na dokumentiranost komponent[157], dodeljevanje enoličnih identifikacijskih številk za komponente, postopke certificiranja in označevanja pravic na komponentah, varnostne vidike (varni prenosi), načela licenciranja in morebiti tudi izračun licenčnin ter izrazoslovje;



Slika 26: Primer arhitekture virtualne spletne knjižnice.

- vsaka knjižnica bi fizično hranila svoje komponente, jih vzdrževala, razpečevala (distribuirala) ter lahko delovala čisto ločeno oziroma neodvisno od spletne knjižnice ali spletnih knjižnic, v katere se povezuje[158];
- zgraditi bi morala katalog komponent, ki bi jih bila pripravljena ponujati na spletnem tržišču (smiselno je, da so to vse komponente, ki jih sama trži tudi sicer)[159];
- vzpostaviti bi morala sistem zaračunavanja komponent članom drugih knjižnic; v ta namen predlagam, da bi bile knjižnice, ki so vključene v spletno tržišče, včlanjene druga v drugo in imele odprte ustrezne račune, člani teh knjižnic pa bi nastopali v vlogi njihovih pooblaščenih uporabnikov; tako bi knjižnica za posredovanje komponente

[157] To pomeni v pogojih za sprejem komponente v knjižnico.
[158] Za povezovanje bi poskrbela tako, da bi svojo arhitekturo gradila na temelju modelov (npr. BIDM z razširitvami), ki predvidevajo medknjižnično povezovanje.
[159] Ažuren katalog mora biti stalno dostopen prek skupnega vmesnika.

ponovnemu uporabniku, ki ni njen član, ampak član neke knjižnice iz skupnega tržišča, licenčnino zaračunala kar tej knjižnici, torej bi knjižnice v tem primeru igrale posredniško vlogo, kar pomeni, da knjižnica lahko podeli licenco drugi knjižnici, hkrati pa nanjo prenese tudi pravico podlicenciranja; za ustrezno zavarovanje pred morebitnimi odgovornostmi v primeru kršitev pravic svojih članov do druge knjižnice ali na komponentah druge knjižnice morajo knjižnice poskrbeti z ustreznimi določbami v pogodbah o včlanitvi ali licenčnih pogodbah;

- posamezne knjižnice morajo podpisati poseben dogovor z vzdrževalcem skupnega vmesnika ali izhodiščne spletne strani, ki vse te knjižnice med seboj povezuje; najbolje je, da ta ali morda programski svet spletnega tržišča[160] določi pravila delovanja in sodelovanja knjižnic, ki so vključene v tržišče, ter da ta pravila predstavljajo sestavni del pogodbe o včlanitvi knjižnice v spletno tržišče;
- posamezne knjižnice so lahko včlanjene v eno ali več spletnih tržišč, vsako spletno tržišče pa morata sestavljati vsaj dve knjižnici.

5.5.4 Varnostni postopki

Temeljni predpogoj, da bodo uporabniki posegali po knjižnici ponovno uporabnih komponent ter njenih virih, je zaupanje vanje. Pri zaupanju gre po eni strani za zaupanje v kakovost komponent, ki jih knjižnica ponuja, in prepričanje, da te komponente v resnici ponujajo to, kar piše v njihovih specifikacijah in dokumentaciji, po drugi strani pa v gotovost, da sta komunikacija in prenos podatkov med njimi in knjižnico varna ter da uporabniki po podpisu licenčne pogodbe dobijo kopijo originalne komponente, katere funkcionalnost so preučevali. Avtorji ali razvijalci komponent želijo svoje komponente zaščititi, da jih kdo po umestitvi v knjižnico ne bi več spreminjal, če tega ne želijo, ali celo, da ne bi ukradel njihovega avtorstva.

Knjižnica mora poskrbeti za ustrezne varnostne mehanizme, ki ji bodo zagotavljali integriteto in njenim uporabnikom vlivali zaupanje. Med najosnovnejše in najpomembnejše sodijo:

[160] Programski svet spletnega tržišča sestavljajo predstavniki posameznih knjižnic tržišča, uporabnikov komponent ter dobaviteljev komponent.

- preverjanje identitete ali verodostojnosti uporabnika (avtentifikacija): postopek preverjanja identitete tistega, ki dostopa do knjižničnih virov;[161]
- preverjanje dostopov (avtorizacija): postopek presoje, ali ima nek uporabnik dovoljenje za dostop ali ne, s čimer preprečujemo nepooblaščene dostope;[162]
- preverjanje originalnosti komponent: postopek ugotavljanja nespremenjenosti komponent, s čimer preprečujemo nepooblaščeno spreminjanje komponent;[163]
- preverjanje identičnosti med komponentami[164];
- zagotavljanje tajnosti podatkov;
- preprečevanje kraje avtorstva komponente.

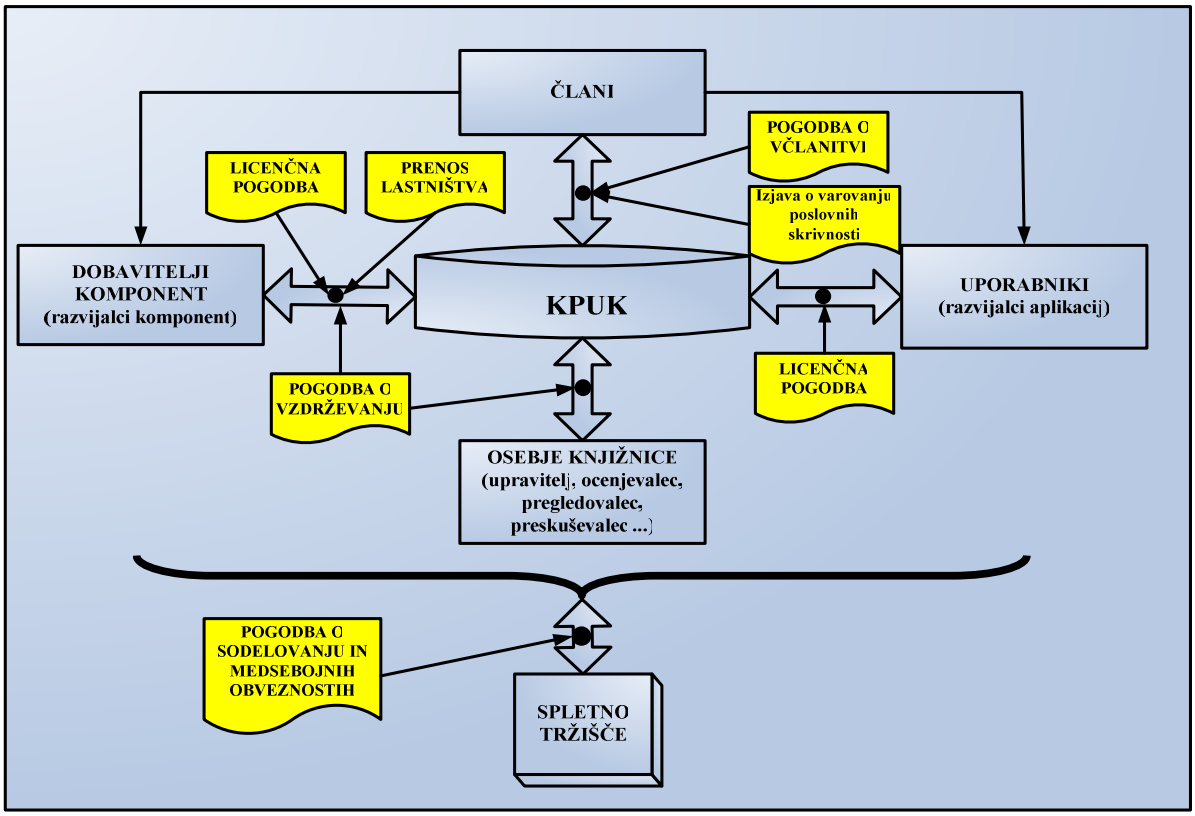
Vse te cilje je mogoče doseči z ustreznimi kombinacijami različnih kriptografskih tehnik in metod. Nekaj predlogov lahko najdemo v [69, 106, 142], v disertaciji pa se podrobnostim o kriptografskih metodah nisem posvečala, ker je tematika preobsežna, da bi jo ustrezno zajela v koncept te disertacije.

[161] Primeri: PGP, Kerberos, X.509, digitalno podpisovanje idr. [69, 142]
[162] Najpogosteje se izvaja s kombinacijo uporabniškega imena in gesla, z imenom domene, šifriranjem z javnim ključem, pregledovanjem seznama dovoljenih dostopov ipd.
[163] Asimetrične kriptografske metode, kombinirane z enoličnim identifikatorjem komponente.
[164] V pomoč upraviteljem knjižnice, da ne sprejemajo istih komponent večkrat in da za isto komponento ne plačajo licenčnine večkrat.

6 POGODBE IN LICENCE ZA PONOVNO UPORABO

Pogodbe so opredeljene kot sporazum, soglasje glede kakih določil ali obveznosti med pogodbenima strankama [148]. Predstavljajo pravni institut, ki omogoča opredelitev tistega, česar zakonodaja ne predpisuje in obravnava, ali natančneje opredeljujejo tisto, kar je v zakonodaji opredeljeno zgolj načelno in je prepuščeno podrobni opredelitvi v pogodbah.

Pogodbe so še toliko bolj pomembne na področjih, kjer ni veliko prakse ali je zelo veliko posebnosti. Eno od takih področij je prav gotovo prenos pravic ter ravnanje v okolju ponovne uporabe, ki se uravnava z različnimi oblikami pogodb. Najpogostejša oblika pogodb so licenčne pogodbe za posamezne programske komponente, ki jih knjižnica podeljuje ponovnim uporabnikom. Sledijo tiste licenčne pogodbe, s katerimi dobavitelji komponent na knjižnico prenašajo določene pravice, ki knjižnici omogočajo nadaljnje razpečevanje in trženje teh komponent. Glede na pomembnost vsebine knjižnice, do katere dostopajo posamezni ponovni uporabniki, je treba s posebno pogodbo o včlanitvi zagotoviti, da se bodo člani držali nekih pravil pri rokovanju z izdelki in storitvami knjižnice. V procesu ponovne uporabe prihaja do stika s poslovnimi skrivnostmi, tako tistimi, ki so zajete v funkcionalnosti komponente, kakor s poslovnimi skrivnostmi delovanja in organizacije knjižnice. Zato mora knjižnica od svojih članov zahtevati podpis pogodbe, s katero se člani obvežejo k varovanju poslovnih skrivnosti. Ker gre v bistvu za vnaprej sestavljeno pogodbo, kjer se posamezna določila ne oblikujejo s pogajanji, sem jo poimenovala kar izjava o varovanju poslovnih skrivnosti. Pa še ene vrste pogodba je zelo običajna za proces ponovne uporabe. To je t.i. pogodba o vzdrževanju, v kateri se knjižnica in dobavitelj komponent dogovorita, kdo in na kakšen način bo vzdrževal komponente, ko jih knjižnica sprejme. Med katerimi strankami nastaja v procesu ponovne uporabe pogodbeno razmerje in kakšno je to razmerje, je prikazano na sliki 27, v nadaljevanju tega poglavja pa so podrobneje opisane vsebine posameznih vrst teh pogodb.



Slika 27: Sklepanje pogodb za ponovno uporabo.

Ponovni uporabnik se torej v procesu pridobivanja ali iskanja ponovno uporabne komponente sooči s kar nekaj pogodbami in lahko tudi različnimi pravili, še posebno, če dostopa do več različnih knjižnic. Kaj hitro se ga lahko poloti zmedenost, toliko bolj, če je sistem podeljevanja licenc zamotan in zahteven, licence pa dolga, težko berljiva besedila. Zato sem poskušala v tem poglavju najprej ugotoviti posebnosti, ki jih prinaša ponovna uporaba v primerjavi s klasičnim razpečevanjem programskih sistemov, na temelju teh posebnosti ugotoviti, česa v določbah licence za ponovno uporabo ne smemo izpustiti, ter sestaviti model, po katerem je mogoče zgraditi vzorce ali družine ponovno uporabnih vzorcev licenčnih pogodb, ki bodo napisane čim krajše, čim bolj jedrnato in razumljivo. Hkrati pa sem podala tudi koncept sistema podeljevanja licenc, ki bi bil v čim večji meri avtomatiziran. S tem sem želela zaokrožiti model procesa ponovne uporabe, ki sem ga delno opisala znotraj poglavja o procesu ponovne uporabe, delno pa znotraj poglavja o modelu knjižnice ponovno uporabnih komponent.

6.1 LICENCE ZA PONOVNO UPORABNE PROGRAMSKE KOMPONENTE

Glede na doslej izpostavljene posebnosti razvoja na temelju ponovne uporabe ter vseh težav, ki iz teh posebnosti izhajajo, lahko sklepamo, da zahteva ponovna uporaba komponent zelo premišljeno in natančno opredeljeno strategijo podeljevanja licenc. Prodaja uporabniških računalniških programov ali programske opreme že vseskozi poteka izključno s podeljevanjem ali prodajo licenc. Torej, če želimo definirati čim bolj optimalen in učinkovit način podeljevanja licenc za ponovno uporabne komponente in vsebine teh licenc, moramo najprej analizirati licence za standardno, paketno ali aplikacijsko programsko opremo, ugotoviti, v čem se ponovno uporabne komponente od te programske opreme razlikujejo, ter na temelju tega sestaviti ogrodja ali vzorce licenc za ponovne uporabne komponente, tako da bodo upoštevane vse ugotovljene razlike. Dosedanjo prakso in načine podeljevanja licenc je treba samo ustrezno nadgraditi, in sicer tako, da se poskušamo v čim večji možni meri izogniti pravnim problemom, ki sem jih opisala v tretjem poglavju oziroma prilogi A, ter v čim večji meri ali celo v celoti podeljevanje licenc prenesti v elektronsko okolje.

Obstaja pa seveda nevarnost, da bi ob morebitnem vključevanju vseh mogočih izjem in opisovanju posebnih primerov uporabe postale licence in pogodbe nasploh še bolj neberljive[165], torej dolga besedila, polna specifičnih pravnih izrazov, ki so za pravno nepoučene razvijalce odbijajoča, včasih pa celo nerazumljiva. Za uspeh razvijalcev je namreč zelo pomembno, da poznajo prijeme in zvijače, ki se uporabljajo pri podeljevanju licenc za ponovno uporabne komponente, saj jim to znanje omogoča boljšo presojo tveganj in stroškov, povezanih s ponovno uporabo neke komponente, v končni fazi pa tudi uspešnosti njihovega izdelka, v katerem bodo (naj bi bile) komponente vgrajene.

Glede na dejstvo, da mora razvijalec aplikacij posegati po številnih ponovno uporabnih komponentah, morebiti tudi iz različnih knjižnic, bi take dolge in zahtevne licenčne pogodbe razvijalca gotovo odvračale od ponovne uporabe. Zato je torej še toliko bolj pomembno, da

[165] Licence so že za standardno programsko opremo običajno zelo dolge in jih le redkokdaj preberemo v celoti in tako natančno, kakor bi morali. Kakor lahko preberemo v [116] so licenčne pogodbe izjemno zapletene pogodbe z velikim številom določb ali klavzul.

knjižnica vzpostavi sistem podeljevanja licenc, ki sloni na kratkih, jedrnatih, razumljivih, preprosto in sistematično napisanih pogodbah, iz katerih bo vnaprej razvidno, katere so tiste posebnosti, ki jih je treba pri ponovni uporabi predmeta licence upoštevati, in predvsem, katera so tista dejanja, ki se jim je treba izogibati oziroma so prepovedana in pomenijo kršitev pravic. Pogodbe pa morajo biti zgrajene tako, da jih je mogoče uporabiti v vseh primerih, od splošnih pa tudi do precej specifičnih.

Znotraj procesa ponovne uporabe se srečujemo z dvema vrstama licenc ali dogovorov o prenosu pravic: pogodbe med razvijalci komponent oziroma dobavitelji komponent in knjižnico, ki predstavlja neke vrste razpečevalca za te komponente in pogodbe med knjižnico in razvijalci aplikacij oziroma ponovnimi uporabniki knjižničnih komponent. Glede na vsebino prenesenih pravic se ti dve vrsti licenc med seboj nekoliko razlikujeta, zato sta potrebni ločeni ogrodji ali modela za oblikovanje vzorcev teh licenc.

6.1.1 Licenca za ponovno uporabno komponento

V splošnem velja, da je licenca ali licenčna pogodba *mehanizem, s katerim imetnik pravic, ki nastopa v vlogi dajalca licence, pridobitelju licence izključno ali neizključno, v celoti ali delno odstopi pravico izkoriščanja ali uporabe predmeta licence, pridobitelj licence pa bo za odstop pravic dajalcu licence dal določeno plačilo*. Pri ponovni uporabi gre po eni strani za pridobivanje licence, ki jo ponovnemu uporabniku v imenu imetnika pravic podeli knjižnica, po drugi strani pa za pridobivanje elektronske kopije komponent od njenega avtorja ali lastnika z namenom, da se vključi v digitalno podatkovno bazo. Zato lahko rečemo, da *uporabnik digitalne knjižnice z licenco kupi dostop do elektronske kopije predmeta licence za določen čas pod določenimi pogoji*. Torej predstavlja licenca formalno dovoljenje za nekaj, kar bi bilo sicer prepovedano, oziroma pravno sredstvo, s katerim se nadzira uporaba izdelka, ki je predmet licence.

V primeru programskih komponent je dajalec licence imetnik intelektualnih pravic na programski komponenti. Najpogosteje je torej dajalec licence hkrati tudi razvijalec komponente, ki pa licence ne podeljuje neposredno (oziroma le redko), pač pa za to pooblasti

knjižnico, v katero shrani svojo komponento. Vsaka tretja stranka, ki neko programsko komponento uporablja brez licence, krši intelektualne pravice lastnika.

Licence ne nadomeščajo drugih oblik pravnega varstva, npr. avtorskega ali patentnega prava ali prava poslovne skrivnosti, ampak jih dopolnjujejo. Licenca lahko podeli vse pravice, ki so (avtorskopravno) varovane, ali pa veliko manj. Zato se morajo knjižnice pri sklepanju licenc z dobavitelji komponent ali njihovimi razvijalci zavedati, da licence lahko omejijo njihove pravice, pa tudi pravice uporabnikov knjižnice.

Vrste in oblike pravic intelektualne lastnine na programski opremi ter vsebina licence za standardno programsko opremo so natančneje opisani prilogi A[166].

Če analiziramo[167] standardne licence za programsko opremo in primerjamo lastnosti predmeta teh licenc s ponovno uporabnimi komponentami, ugotovimo, da se najočitnejše razlike med njima nanašajo na namen, velikost, kakovost in prilagodljivost. Ravno te razlike pa tudi v veliki meri vplivajo na vsebino licence oziroma njenih določb. Najbolj opazne in pomembne razlike nastanejo v določbah, ki se nanašajo na plačila (licenčnine), lastništvo, obseg prenesenih pravic, odgovornosti, jamstva in vzdrževanje, ki predstavljajo bistveni del licence za programsko opremo (programske licence). Licenca za ponovno uporabno komponento mora torej vsebovati vsaj določbe, ki se nanašajo na[168]:

- pravice, ki jih dajalec licence odstopi ali prenese na prejemnika licence,
- plačila dajalcu licence,
- lastništvo na predmetu licence in njegovih modifikacij (predelav),
- tveganja, odgovornosti in obveznosti posameznih pogodbenih strani ter
- tehnično podporo, vzdrževanje in jamstva za komponento, ki je predmet licence.

[166] Večina razvijalcev svoje programske komponente varuje avtorskopravno in/ali kakor poslovno skrivnost, zelo redko pa za varovanje intelektualnih pravic izkoristi patent. Glavni razlog so predvsem visoka cena za pridobitev patenta, poleg tega pa zelo uporabne (najpogosteje ponovno uporabljene) komponente ne zadoščajo merilom za izum in nesplošnost, ki morata biti izpolnjena za pridobitev patenta.

[167] Analiza je bila narejena na podlagi nekaterih monografij, ki so v celoti posvečene licenciranju programske opreme, npr. [84, 89], modelov licenc npr. http://www.licensingmodels.com ter nekaterih vzorcev konkretnih licenc za programsko opremo, ki sem jih pridobila bodisi v papirnati bodisi elektronski obliki iz različnih spletnih strani.

[168] Pri določanju nabora nujnih določb licence za ponovno uporabno komponento so upoštevane predvsem okoliščine, ki lahko pripeljejo do problemov pravne narave, opisanih v, podpoglavju 3.2. tretjega poglavja.

Z licenčno pogodbo pridobi pridobitelj licence izključno pravico izkoriščanja predmeta licence samo, če je to izrecno dogovorjeno. V vseh drugih primerih pa pridobi pridobitelj neizključno licenco[169]. Pri podeljevanju licenc za ponovno uporabne komponente bi moralo biti vsakemu dajalcu licence v interesu, da podeli le neizključno licenco, saj mu to omogoča, da v istem času in prostoru odstopi pravico do uporabe istega predmeta licence več različnim pridobiteljem, lahko pa predmet licence v tem času tudi sam izkorišča[170]. S tem pa je tudi doseženo bistvo ponovne uporabe ali cilj, ki naj bi ga z gradnjo ponovno uporabnih komponent dosegli.

6.1.2 Razlike v izhodiščih za standardne programske licence in licence za ponovno uporabne komponente

Razlike med klasičnimi programskimi izdelki, ki so predmet standardnih licenc, in ponovno uporabnimi komponentami so precejšnje, kar se odraža tudi v razlikah v strukturi in vsebini med licenčnimi pogodbami za klasične programske izdelke in licenčnimi pogodbami za ponovno uporabne komponente. Temeljna izhodišča, ki te razlike ustvarjajo, se nanašajo na primer na namen, velikost, kakovost in prilagodljivost predmeta licence, v katerih je tudi bistvena razlika med klasičnimi predmeti licence in ponovno uporabnimi komponentami.

1. Namen ponovno uporabne komponente je bistveno drugačen od namena standardne programske opreme. Večina licencirane standardne programske opreme je v obliki samostojnih aplikacij. Razvijalci navadno licence podelijo razpečevalcem svojih aplikacij, ki to programsko opremo prodajajo končnim uporabnikom *(»shrink-wrap«*, *»click-wrap«* licenčne pogodbe[171]). Ponovno uporabne komponente pa ne predstavljajo in niso sposobne delovati kot samostojne aplikacije, razen morda v redkih primerih.
2. Običajno se licence za manjše komponente ne podeljujejo, saj so stroški, povezani z dajanjem licence in pogajanjem za obseg izkoriščanja te komponente mnogokrat višji od stroškov razvoja manjše komponente na novo. Pogosto se razvijalci odločajo take manjše komponente ponujati kot prosto ali poskusno programsko opremo, saj so za ustvarjanje nekega bistvenega prihodka ali celo dobička premajhne.

[169] Obligacijski zakonik, 707. člen, str. 704. [116]
[170] Obligacijski zakonik, stran 704 (komentar k 707. členu). [116]
[171] Glej prilogo A, razdelek A.6.

3. Kakovost predmeta licence je pomemben faktor pri dajanju licence. Pridobitelj licence namreč pričakuje in zahteva visoko kakovost komponente, saj predpostavlja, da je dajalec licence komponento predhodno dodobra preskusil in iz nje odstranil vse morebitne napake ter odpravil pomanjkljivosti.
4. Pridobitelji licenc pričakujejo, da bodo komponente, ki so predmet licence, prilagodljive, saj naj bi bile uporabne v številnih kontekstih. Če naj razvijalec to prilagodljivost doseže, potem mora bodisi izdelati zelo natančen in popoln vmesnik bodisi pridobitelju licence omogočiti dostop do izvorne kode.

6.1.3 Bistveni elementi licence za ponovno uporabo programske komponente

Zahteve, ki jih poraja ponovna uporaba komponent, ter lastnosti teh komponent vplivajo tudi na vsebino določb licence ali licenčne pogodbe oziroma na oblikovanje licenčnih pogojev. Glede na to, da licence med knjižnico in ponovnim uporabnikom oziroma razvijalcem aplikacij prevladujejo po številu in raznovrstnosti, so v nadaljevanju našteti in opisani najpomembnejši elementi, ki jih je treba vključiti v določbe te licence in ki hkrati predstavljajo osrednji del licenčne pogodbe.

Predmet licence

To je najbolj bistvena določba v licenčni pogodbi, saj je v njej opredeljeno, katere pravice intelektualne lastnine prenaša dajalec licence na pridobitelja in v kolikšnem obsegu.

Večina programskih licenc, ki jih knjižnica podeli ponovnemu uporabniku, podeljuje v primeru kodnih oziroma izvršljivih komponent pravico do uporabe ali izkoriščanja objektne kode in omejene pravice do reprodukcije programske opreme, ki je predmet licence (to se nanaša predvsem na izdelavo ene varnostne ali arhivske kopije predmeta licence). Pridobitelj licence ali prihodnji ponovni uporabnik predmeta licence pa potrebuje in tudi zahteva veliko širši obseg pravic, saj želi komponente, za katere je pridobil licenco, vgraditi v svoj izdelek ali morebiti celo te komponente spremeniti ali jih prilagoditi in s tem ustvariti izpeljano delo. V primeru prilagajanja ali spreminjanja komponente mora imeti pridobitelj licence vpogled tudi

v izvorno kodo komponente. Morebitni prenos teh pravic na pridobitelja licence pa v bistvu pomeni, da mu mora razvijalec prepustiti nadzor nad programskimi komponentami, česar se razvijalci komponent ali dajalci licenc bojijo.

Zato morajo biti odstopljene pravice natančno opredeljene in uravnotežene med dajalcem in pridobiteljem licence. To uravnoteženost pa je mogoče doseči samo z ustreznimi omejitvami, vključenimi v tiste določbe licenčne pogodbe, ki se nanašajo na obseg dodeljenih pravic z licenco.

Namesto da bi dajalec licence pridobitelju omogočil integriranje in spreminjanje licencirane komponente brez omejitev, se v tej določbi navedejo točno določeni pogoji, pod katerimi se to lahko zgodi, na primer, da je licenca omejena na integriranje komponente v točno določen izdelek, za katerega se v opisu navede tudi ime in podobno. Ravno tako se v licenci lahko določi le manjši segment komponente, nad katerim lahko pridobitelj licence izvaja spremembe, ali pa se od pridobitelja licence zahteva, da za kakršnokoli spreminjanje predhodno pridobi pisno dovoljenje dajalca licence.
Pri določanju obsega izkoriščanja predmeta licence je treba opredeliti tudi morebitno možnost vgraditve predmeta licence v več različnih izdelkov pridobitelja licence, ki se bodo ločeno tržili. V tem primeru bi bilo po našem mnenju najbolje, da se za vsako ponovno uporabo iste komponente sklene ločena licenčna pogodba, saj je možno le tako ustvariti nadzor nad prometom s predmetom licence, morebitnimi spremembami predmeta licence, kršitvami pravic ipd. Vsi pogoji morajo biti zapisani jasno in nedvoumno.

Plačilo ali licenčnina

Izhodišča o namenu, velikosti in kakovosti programske komponente spremenijo tudi plačilne pogoje, navedene v standardni licenci. Te namreč določajo vnaprejšnje, fiksno plačilo celotne licenčnine, kar pridobitelju licence omogoča popolno izkoriščanje z licenco dodeljenih pravic takoj po plačilu.

Tak način plačevanja pa v primeru licence za ponovno uporabno komponento za pridobitelja licence ni najbolj ustrezen oziroma zanimiv, saj je vnaprej zelo težko natančno predvideti zaslužek s prodajo izdelka, v katerega bo vgrajena komponenta, ki je predmet licence. Če

vgrajena komponenta predstavlja precejšen in/ali celo najpomembnejši del programskega izdelka, katerega del je postala, bi lahko dajalec licence pridobitelju upravičeno zaračunal visoko licenčnino. V nasprotnem primeru pa predstavlja zelo visoka vnaprejšnja licenčnina za pridobitelja licence veliko tveganje, če izdelek, v katerega bo to komponento vgradil, ne bo komercialno uspešen. Zato je pri strukturiranju licenčnine za ponovno uporabno komponento primerneje, če ob sklenitvi licenčne pogodbe pridobitelj vnaprej plača nek delni znesek licenčnine, ki bo dajalcu licence vsaj delno povrnil finančne stroške, nastale s podeljevanjem licence, preostali del pa pridobitelj dajalcu licence izplačuje postopno kot dogovorjen odstotek prihodka od prodaje pridobiteljevega končnega izdelka, v katerega je bila ta komponenta vgrajena (torej glede na ustvarjen promet s predmetom licence na strani pridobitelja licence). Na tak način se pridobitelj lahko tudi zavaruje za primer, ko dajalec licence ne bi spoštoval določb o vzdrževanju ponovno uporabnih komponente po podelitvi licence zanjo. To je namreč za pridobitelja licence lahko usodno ali pa zanj predstavlja velika tveganja in tudi stroške. Natančneje se tovrstne medsebojne obveznosti opredelijo v pogodbi o vzdrževanju in v določbah licence, ki se nanašajo na jamstva pravic, povračila škode, odškodnine in podporo.

Določba o višini plačila, času, v katerem mora pridobitelj licence plačati dogovorjeni znesek in načinu izplačila dogovorjenega plačila, se lahko po potrebi tudi spremeni, in sicer v primeru, če je postalo dogovorjeno plačilo očitno nesorazmerno v primerjavi s prihodkom, ki ga je deležen pridobitelj licence z izkoriščanjem predmeta licence[172].

V kolikor je z licenčno pogodbo dovoljena večkratna uporaba istega predmeta licence v različnih izdelkih pridobitelja licence, je treba na ustrezen način strukturirati tudi plačilo ter v njem upoštevati predviden promet z vsemi izdelki.

Ustrezno plačilo za ponovno uporabno komponento, ki je predmet licence, je pomemben motivacijski faktor tako za razvijalce komponent kakor za njihove uporabnike. Razvijalcem je v interesu, da zgradijo čim kakovostnejšo komponento, z jasno funkcionalnostjo in čim širšo ponovno uporabnostjo, saj imajo tako zagotovilo, da bo veliko ponovnih uporabnikov posegalo po njihovi komponenti in bo zaslužek s tako komponento velik. Po drugi strani pa

[172] Obligacijski zakonik, 720. člen, str. 712. [116]

uporabniki komponent radi posegajo po takih komponentah in zanje tudi plačajo ustrezno licenčnino, saj vedo, da so pridobili kakovostno komponento, za katero so prihranili razvojni čas in stroške. V vsakem primeru predstavlja licenčnina sredstvo za zaščito pravic in zagotavljanje stimulacije razvijalcem komponent na eni strani ter sredstvo za zagotavljanje ponovne uporabnosti komponente na drugi strani.

Lastništvo

Izhodišča o delovanju in prilagodljivosti ponovno uporabnih komponent v veliki meri spremenijo tudi določbo o lastništvu. V standardnih programskih licencah za samostojne aplikacije dajalec licence odstopi samo nekatere pravice do uporabe ali izkoriščanja in omejene pravice do reprodukcije programske opreme, zadrži pa vse ostale pravice, lastništvo in interese na programski opremi, ki je predmet licence[173].

Določba o lastništvu v licenci za ponovno uporabno komponento mora poleg lastništva na ponovno uporabnih komponentah obravnavati tudi lastništvo na programskem izdelku pridobitelja licence, v katerega je ali bo vgrajena komponenta (lastništvo izpeljanega dela), ter lastništvo na spremembah, ki so bile ali bodo nad to komponento narejene, vključno z lastništvom na različicah.

Dajalec licence mora ohraniti lastništvo na komponenti, da jo bo lahko prodal (podelil licenco za njeno uporabo) tudi drugim strankam, kajti to je nenazadnje bistvo in cilj ponovne uporabe, poleg tega pa si mora zagotoviti tudi možnost, da bo to komponento tudi sam ponovno uporabljal, če jo bo potreboval. Ravno tako mora pridobitelj licence, ki komponento vgradi v svoj izdelek, ostati ali postati lastnik svojega izdelka, da ga bo lahko uporabljal, prodajal in nasploh z njim razpolagal, seveda v skladu z licenco ali licencami za komponento/e, ki jo/jih je vgradil v svoj izdelek.

Strukturiranje že tako občutljive določbe o lastništvu postane še težavnejše, ko je pridobitelju licence z licenco dovoljeno tudi spreminjanje ali prilagajanje predmeta licence – komponente, kar pomeni, da ima pridobitelj licence neposredni dostop do izvorne kode, ki jo lahko uporablja in spreminja. Po eni strani bi lahko lastništvo na spremembah pridobil kar dajalec

[173] Obligacijski zakonik, 704. člen, str. 701-703. [116]

licence, saj pridobitelj licence po izteku licence teh sprememb brez komponente ne more uporabljati. Po drugi strani pa lahko pridobitelj licence nad komponento izvede zelo specializirane in zanj specifične spremembe, ki za dajalca licence niso zanimive, zato dajalec licence nima interesa pridobiti lastništva nad temi spremembami.

Tudi pri oblikovanju določb o lastništvu je torej treba upoštevati interese obeh strani. Če bi pridobitelj licence nameraval izvesti take spremembe, ki se izkažejo za dajalca licence zelo zanimive, bi seveda dajalec licence skušal postati lastnik teh sprememb. Lahko pa se dajalec licence odloči, da pridobitelju licence ne dovoli izvajati modifikacij nad komponento, ampak jih izvede sam na temelju specifikacij zahtev, ki mu jih posreduje pridobitelj licence, nato pa prejemniku licence podeli novo licenco za modificirano komponento. Na tak način si dajalec licence zagotovi absolutno lastništvo na komponenti in na modifikacijah le-te. Če pa ima pridobitelj licence namen nad komponento izvesti take spremembe, ki za dajalca licence niso zanimive, lahko dajalec licence odstopi lastniške pravice na teh spremembah.

Pogodbene obveznosti oziroma odgovornosti

Med standardnimi programskimi licencami za samostojne aplikacije in licencami za ponovno uporabne komponente ni bistvenih razlik v določbi o prevzemu, oprostitvi in omejitvah odgovornosti. V določbah o oprostitvi odgovornosti je navadno opredeljeno, da bo dajalec licence zagovarjal in branil pridobitelja licence, če bi ga tretja stranka sodno preganjala zaradi domnevne kršitve pravic intelektualne lastnine med uporabo programske opreme, ki je predmet licence. Določbe o omejitvah odgovornosti na splošno omejujejo odgovornost posameznih strani, vključenih v licenčni dogovor.

Tako kakor pri drugih določbah licence si morata dajalec in pridobitelj licence porazdeliti tudi odgovornosti in tveganja. Dajalec licence lahko pridobitelja licence oprosti v primeru kršitve intelektualnih pravic na komponenti, ki je predmet licence, če je ta kršitev nastala kot posledica pooblaščene uporabe komponente. Posledično lahko pridobitelj licence prevzame stroške zagovarjanja zaradi kršitev, če te kršitve izhajajo iz kombiniranja komponente, ki je predmet licence, z njegovo lastno programsko opremo ali iz modifikacij predmeta licence, ali pa so nastale zaradi napačne uporabe ali zlorabe predmeta licence.

Navedene standardne omejitve določil o prevzemu odgovornosti z manjšimi spremembami lahko veljajo tudi pri licencah za ponovno uporabne komponente. Znotraj te določbe se lahko obe pogodbeni stranki odrečeta odgovornostim, povezanim s škodami, ki jih ni mogoče vnaprej predvideti ali se v določbi zapiše najvišja stopnja odgovornosti dajalca licence. Slednje se nam zdijo v okolju ponovne uporabe programskih komponent zelo primerne, še posebno, ko pridobitelj licence komponento, ki je predmet licence, vgradi v zelo tvegano aplikacijo.

Jamstvo in podpora

Značilnosti programskih komponent, njihova vgraditev v programsko opremo tretje stranke, velikost, kakovost in prilagodljivost prožijo tudi močne zahteve po tehnični podpori na eni strani ter jamstvu za njihovo delovanje in kakovost na drugi strani. V standardnih licenčnih dogovorih je obdobje jamstva zelo kratko, pa tudi zahtev po podpori ni veliko. V tem je tudi bistvena razlika med standardno programsko licenco in licenco za ponovno uporabno komponento.

Pridobitelju licence je seveda v interesu pridobiti licenco za (ponovno) uporabo ali izkoriščanje neke komponente, zato je zanj še toliko bolj pomembno, da so v licenčni pogodbi tudi določbe o jamstvu, podpori in vzdrževanju komponent. Razvijalec aplikacije pridobiva licenco z namenom, da bo neko komponento, za katero je prepričan, da je zelo kakovostna in zelo prilagodljiva, vgradil v svoj programski sistem in bo to zanj predstavljajo nižje stroške, kakor da bi komponento sam na novo razvil. Ta določba predstavlja torej za pridobitelja licence določena zagotovila o kakovosti in ponovni uporabnosti predmeta licence. Zato bi moralo biti dajalcem licence v interesu, da z licenco podelijo tudi močno, vendar časovno kratko jamstvo, po preteku tega obdobja pa ponudijo tehnično podporo po standardnih tržnih cenah. Na primer, dajalec licence lahko jamči za enoletno brezhibno delovanje komponente po specifikaciji. V tem času bo pridobitelj licence domnevno vgradil komponento v svoj izdelek. V času trajanja jamstva bo dajalcu licence omogočil reševanje nepravilnosti oziroma odpravljanje napak, ki bodo nastale kot posledica kombiniranja komponente s programsko opremo tretje stranke. Če v komponenti obstajajo večje napake, potem bi se morale pojaviti v tem obdobju. Stroški odpravljanja napak se izvzamejo iz licenčnine, ki jo je dolžan pridobitelj plačati dajalcu licence. Po izteku obdobja jamstva je smiselno, da dajalec licence odpravi tudi

vse druge napake, ki se morebiti še pojavijo, vendar mora v tem primeru pridobitelj licence storitev plačati ločeno, oziroma se morata pogodbeni stranki dogovoriti za znesek vzdrževalnine, ki se navadno (vsaj pri klasičnih licencah) zaračunava letno v določenem odstotku celotne licenčnine. V tej določbi se določijo tudi vse druge podrobnosti, ki se nanašajo na probleme vzdrževanja in posledično plačila za vzdrževanje ali povračilo škode.

Kombinacija močnega, vendar kratkotrajnega jamstva in široke tehnične podpore v zameno za vzdrževalnino bi lahko uravnala interese tako dajalca kakor pridobitelja licence. Kratkoročno jamstvo pridobitelja licence motivira, da komponento, ki je predmet licence, čim prej vgradi v svoj programski izdelek, sočasno pa dajalec licence z jamstvom tudi potrjuje kakovost komponente, ki je predmet licence. V času trajanja jamstva lahko dajalec licence komponento dodatno preskusi in tudi izboljša. Obdobje vzdrževanja in tehnične podpore, ki sledi preteku jamstva, pridobitelju licence zagotavlja, da mu komponente ne bo treba podrobno preučevati. Potreba po natančnem preučevanju vsake komponente pred njeno ponovno uporabo zaradi ugotavljanja njene brezhibnosti bi izničila temeljni namen dodeljevanja licenc za komponente. Nenazadnje lahko obdobje vzdrževanja in zagotavljanja tehnične podpore dajalec licence izkoristi za odstranitev morebitnih napak in nepravilnosti iz komponente, s čimer poveča kakovost, zanesljivost in ponovno uporabnost komponente, za kar sicer ne bi imel dovolj motivacije.

Nerazkritje

Pri prilagajanju določb o zaupnosti informacij iz standardnih programskih licenc za licence za ponovno uporabne komponente je treba narediti le nekaj manjših sprememb. V standardnih licencah je v teh določbah navedeno, da bosta obe pogodbeni strani v določenem časovnem obdobju zaupne informacije, ki si jih izmenjujeta v času dogovarjanja ali pogajanja, ščitili in jih ne bosta razkrili.

Dajalec in pridobitelj licence gojita glede zaupnosti nasprotujoče si interese. Dajalci licenc si, v nasprotju s pridobitelji, želijo močna, zelo obvezujoča določila o zaupnosti, saj morajo zaščititi intelektualne pravice na komponenti, ki je predmet licence, posebno še, če gre za poslovno skrivnost. Glede na to, da je tveganje dajalca licence pred razkritjem poslovne skrivnosti veliko večje od napora, ki ga mora pridobitelj licence vložiti v vzdrževanje

zaupnosti informacij, povezanih s to komponento (omejitev dostopa do komponente, zagotavljanje, da bodo delavci ali pooblaščeni uporabniki pridobitelja licence s pisno izjavo zagotovili varovanje zaupnosti), naj pridobitelji licence ne bi nasprotovali vključitvi tovrstnega določila v licenčni dogovor. Dajalec licence lahko od pridobitelja tudi zahteva, da komponento pred shranjevanjem ali razpečevanjem ustrezno zakodira in s tem prepreči nepooblaščeno in nezaželeno razkritje zaupnih informacij o komponenti.

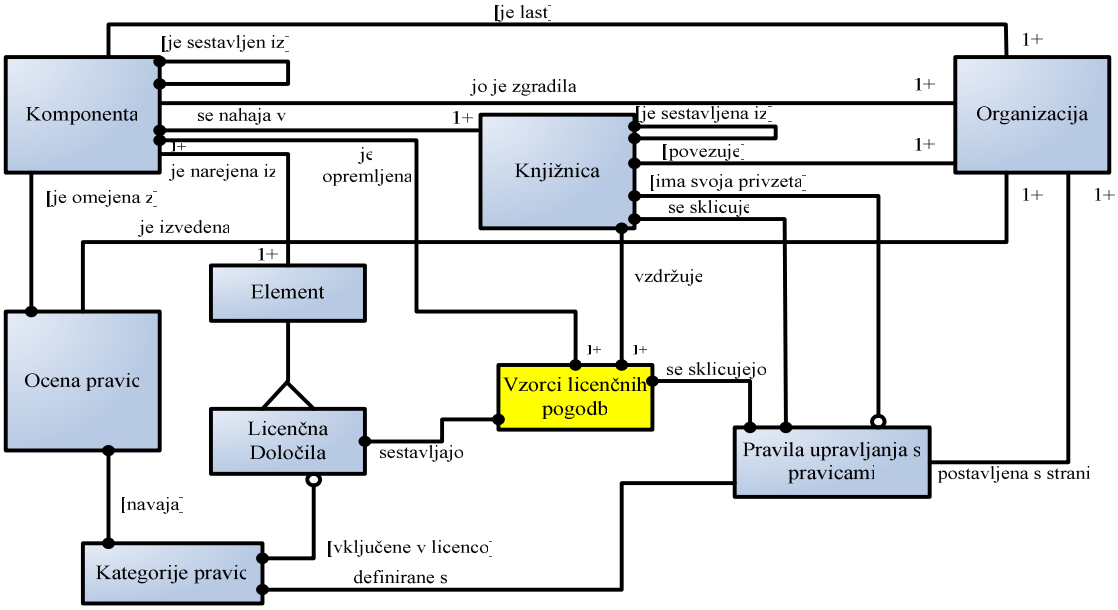
Glede na neprimernost obstoječe zakonodaje s področja varovanja pravic intelektualne lastnine na ponovno uporabni programski opremi, (p)ostajajo licenčne pogodbe izjemno pomembno sredstvo za uravnavanje medsebojnih pravic in obveznosti med dajalcem licence in njenim pridobiteljem ali njenimi pridobitelji. Bistvo, ki naj bi ga z licenčno pogodbo dosegli, je uravnotežena porazdelitev pravic in obveznosti med obema pogodbenima stranema, saj je samo v tem primeru dosežen predpogoj za ponovno uporabo. Dajalec licence mora biti seveda motiviran, da za svoje ponovno uporabne komponente podeli licence, pridobitelju licence pa ne smejo biti dani prestrogi pogoji in zahteve za ponovno uporabo neke komponente, sicer za ponovno uporabo ni motiviran. Ravno tako ne sme biti licenčnina previsoka, sicer se bo potencialni pridobitelj licence odločil za razvoj komponente na novo. Z določbami licenčne pogodbe je treba opredeliti vse naštete pogoje in obveznosti pogodbenih strank. Licenčne pogodbe morajo torej izražati namere obeh pogodbenih strank, iz njih morajo biti razvidne prilagodljivost, kakovost in interoperativnost programskih komponent, varovano mora biti lastništvo dajalca licence na komponenti, skratka, oblikovane morajo biti zelo previdno in morajo ščititi interese obeh strani. Pri oblikovanju teh pogodb pa je treba upoštevati, da njihov namen ni samo še bolj zbirokratizirati postopke ponovne uporabe in otežiti ponovno uporabo, ampak nasprotno, da gre s sklepanjem licenčne pogodbe za spodbujanje in podpiranje ponovne uporabe. Zato je v okviru sistema knjižnice ponovno uporabnih komponent smiselno razmisliti tudi o standardu za oblikovanje licenčnih pogodb za ponovno uporabne komponente, ki bo upošteval vse do sedaj navedene vidike in pomanjkljivosti iz prakse. Sistem dodeljevanja in oblikovanja licenčnih pogodb je treba v čim večji meri avtomatizirati (npr. on-line pogodbe).

6.1.4 Licenčni pogoji kot sestavni del dokumentacije komponente

Čeprav je licenčna pogodba rezultat pogajanja med strankama, bi bilo smiselno, da bi bilo sestavni del dokumentacije, ki spremlja knjižnično komponento, tudi ogrodje licence za to komponento. V njem bi dajalec licence specificiral svoje temeljne zahteve in pogoje, pod katerimi je pripravljen za komponento podeliti licenco, knjižnica pa bi glede na obseg dodeljenih pravic komponente klasificirala v enega od razredov. Pri pregledovanju knjižničnih komponent in iskanju najprimernejše ponovno uporabne komponente ali odločanju o pridobitvi neke komponente za ponovno uporabo imajo podatki iz tega ogrodja licence lahko zelo pomembno, če že ne odločilne vloge. Tako bi se lahko ponovni uporabnik že ob pregledu dokumentacije komponente, ki bi mu bila zanimiva za ponovno uporabo, odločil, da odstopi od njene ponovne uporabe, če mu pogoji ne bi ustrezali. Obstaja sicer določeno tveganje, da bi ob prestrogih pogojih ponovni uporabniki prehitro odstopili od ponovne uporabe, zato mora knjižnica v postopku sprejemanja komponente natančno pregledati tudi te pogoje in glede na svojo strategijo po potrebi izpogajati blažje, ali morebiti tudi strožje licenčne pogoje. Tako knjižnici kakor dobaviteljem komponent je namreč v interesu, da se odločanje o ponovni uporabi ne ustavi že na tem prvem koraku pri pregledovanju licenčnih pogojev. Če knjižnica svoja merila (vsaj v grobem) opredeli v pravilih, lahko to za dobavitelje komponent ali njihove razvijalce predstavlja dobro iztočnico pri opredelitvi temeljnega nabora licenčnih pogojev.

Ogrodje licence bi moralo predstavljati obvezni sestavni del dokumentacije, ki mora spremljati ponovno uporabno komponento ob njenem sprejemu v knjižnico. Če dajalec licence predvidi več različnih možnosti za ponovno uporabo neke komponente ali je pripravljen v različnih primerih dodeliti licenco za isto komponento pod različnimi pogoji, lahko pripravi ogrodja za več licenc. Dokončno oblikovanje ali usklajevanje vsebine licenčnih določb poteka, ko se potencialni ponovni uporabnik v bistvu že odloči za neko komponento in načeloma že sprejme vse pogoje in obveznosti iz spremljajočega predloga ali ogrodja licenčne pogodbe. Na tak način bi se dalo postopek dodeljevanja ali pridobivanja licenc za ponovno uporabne komponente pohitriti, kar predstavlja za ponovne uporabnike bistveno pridobitev, ker lahko po funkcionalnosti in ustreznosti izbrano komponento toliko prej uporabijo. S tem pa pridobijo čas in denar.

V podporo temu konceptu sem podatkovni model za pravice intelektualne lastnine (opisan v prilogi C) razširila z razredom Vzorci licenčnih pogodb (slika 28), ki združujejo družine vzorcev pogodb, sestavljenih iz licenčnih določil na temelju pravil upravljanja s pravicami, ki so sestavni del pravil knjižnice, na katera se sklicujejo posamezne določbe v vzorcih. Vsaki komponenti je prirejen en ali več vzorcev.



Slika 28: Razširjen podatkovni model za pravice intelektualne lastnine.

6.1.5 Programsko podprto sklepanje licenčnih pogodb

Druga, z vidika učinkovitosti procesa podeljevanja licenc za komponente še boljša možnost, se nam zdi vzdrževanje samostojne baze ponovno uporabnih vzorcev licenčnih pogodb. Tudi licenčne pogodbe lahko namreč obravnavamo kot ponovno uporabne komponente, pri čemer kot parametre za prilagajanje postavimo tiste določbe, ki so navadno zelo odvisne od posamezne komponente ali družine komponent: višina licenčnine, možnost spreminjanja, vzdrževanje, rok veljavnosti ipd. Dobavitelj komponente nato z navedbo identifikacijskih številk vzorcev licenčnih pogodb v dokumentaciji komponente poveže množico vzorcev licenčnih pogodb, ki bi se lahko uporabile pri licenciranju, s konkretno komponento (glej sliko 29). Ob pregledovanju dokumentacije se s klikom na številko licence odpre besedilo, ki ga lahko ponovni uporabnik pregleda. V tem primeru so ponovnemu uporabniku še bolj

transparentno posredovane vse morebitne omejitve uporabe komponente in druge zanj pomembne informacije. Če ponovni uporabnik ne uspe med vzorci izbrati zase primernega, nadaljuje pogajanja s knjižnico. Kot iztočnico ali osnovo za pogajanja izbere vzorec licenčne pogodbe, ki je najbliže njegovim zahtevam in potrebam.

Z vzdrževanjem take podatkovne baze sočasno tudi standardiziramo postopek sklepanja in vsebino licenčnih pogodb. Po nekaj pridobljenih licencah so ponovni uporabniki že dodobra seznanjeni z vzorci pogodb, pa tudi z vsebino standardnih določb, zato se lahko pri nadaljnjem licenciranju osredotočajo le na parametre licenčne pogodbe, ki bodo veljali za konkretno komponento, kar v veliki meri poenostavi in pohitri sicer zelo administrativen in dolgotrajen postopek pridobivanja licence za komponento.

Seveda je tako licenciranje smiselno le v primeru večjih komponent, kjer je treba prilagajati komponente specifičnim zahtevam in so tudi licenčnine relativno visoke. Za manjše, črne komponente, predlagamo nabor vnaprej pripravljenih licenčnih pogodb v stilu *"click-wrap"* licenc, ki jih ponovni uporabnik podpiše enostavno s klikom na gumb *"Se strinjam"* ali *"Sprejemam licenčne pogoje"*. Vsako drugačno licenciranje takih komponent bi bilo tako za knjižnico kakor za ponovnega uporabnika cenovno in časovno nesprejemljivo.

Za vsako licenčno pogodbo lahko knjižnica vzporedno hrani seznam komponent, ki so bile predmet take pogodbe. Licence je možno razvrstiti v razrede tudi glede na vsebine nekaterih njenih tipičnih določb kakor npr. izključne in neizključne licence, prenosljive ali neprenosljive licence, licence, ki vključujejo ali ne vključujejo možnosti spreminjanja predmeta licence ali izdelave izpeljanih del iz predmeta licence, licence, za katere je licenčnino treba poravnati v enkratnem znesku, in tiste, kjer se višina licenčnine oblikuje sproti ob prodaji programskega izdelka pridobitelja licence ipd. Ob ustrezni klasifikaciji licenčnih pogodb je možno to sklepno dejanje pred ponovno uporabo neke knjižnične komponente bistveno pohitriti.

Dodatno je seveda smiselno oblikovanje licenčnih pogodb, torej kombiniranje posameznih določil, pogojev in obveznosti, programsko podpreti. Tako na eni strani omogočimo izdelavo novih vzorcev ponovno uporabnih pogodb, na drugi strani pa izpogajanje specifičnih pogojev

pri posameznih določbah. Seveda mora knjižnica vzporedno poskrbeti tudi za ustrezen mehanizem certificiranja teh vzorcev pogodb, pregledovanje sovpadanja in podobnosti med pogodbami, in se tako izogniti nepregledni množici več ali manj podobnih pogodb, med katerimi bi bili prisiljeni brskati dobavitelji komponent ob posredovanju komponente v knjižnico. Temelja pri izbiri sta seznam vnaprej napisanih določb in seznam posameznih obveznosti, pogojev in določil znotraj posamezne določbe. Za določbo, ki vključuje posebnosti in izjeme, lahko program omogoči vnos poljubnih pogojev in določil, pri drugih določbah pa ponudimo množico pogojev in določil, ki je ni mogoče spreminjati, ampak samo izbirati, ali pri določilih, kjer je to treba, dopisati čas, znesek in podobno, torej parametre, ki so v celoti odvisni od predmeta licence in njegovih konkretnih lastnosti ali od zahtev in potreb pogodbenih strank. Ker se določeni pogoji ali določila med seboj izključujejo, mora biti v program vgrajena logika, ki poskrbi, da zamrzne tiste opcije, ki so zaradi izbire nekega drugega določila izključene. Tako v veliki meri tudi zmanjšamo možnost napak ali kontroverznosti ali neskladij, ki se lahko pojavijo znotraj licenčne pogodbe.

Poleg izbiranja in kombiniranja posameznih določb pogodbe ter vrednosti elementov teh določb mora program omogočati tudi *mehanizem pogajanja*. Ponovni uporabnik namreč dobi v procesu pridobivanja komponent že izpolnjeno licenco s pogoji, ki jih je določil licencodajalec. Pri vsakem elementu, ki se ne izpolni samodejno, ampak ga izbere ali vpiše licencodajalec, mora imeti ponovni uporabnik možnost, da izrazi nestrinjanje z njim, ali da navede svoje zahteve. Zato mu program pri teh elementih ponudi možnost, da potrdi sprejem pogojev iz licence ali svoje nestrinjanje izrazi tako, da z istega seznama možnosti, kakor jih je imel licencodajalec, izbere tisto, ki mu bolj ustreza, ali da v primeru ročnega vnosa v polje vpiše svoje vrednosti. Če se ponovni uporabnik strinja z vsemi pogoji in je torej potrdil vse določbe pogodbe, lahko pride do elektronskega podpisa licence. V nasprotnem primeru licencodajalec pregleda elemente, v katerih se ponovni uporabnik ne strinja, in se odloči, ali sprejme njegove pogoje ali ponudi svoje spremenjene. Če sprejme pogoje licencojemalca, lahko pride do takojšnjega podpisa pogodbe, sicer pa licenco z novimi pogoji ponovno pošlje licencojemalcu v pregled. Postopek se konča bodisi s sprejetjem pogojev z obeh strani bodisi z odstopom licencojemalca od komponente.

Da lahko sploh sestavljamo vzorce licenčnih pogodb, potrebujemo neko ogrodje, v katerem je predpisano, kateri so tisti elementi, ki jih posamezna določba mora vsebovati ali jih lahko vsebuje. Tak model ogrodja je opredeljen v nadaljevanju.

6.1.6 Model ogrodja licenčne pogodbe za ponovno uporabne komponente

V nadaljevanju predstavljeni model ogrodja licenčne pogodbe zajema najpomembnejše elemente posameznih določb in pogojev, ki morajo biti vključeni v licenčni pogodbi, katere predmet je ponovno uporabna komponenta. Z njim sem želela poudariti oziroma izpostaviti vsebinske zahteve z vidika ponovne uporabe, ki morajo biti v določbah izražene, da se izognemo čim večjemu številu poznejših težav, ni pa bil naš namen oblikovanje natančnega pravnega besedila posameznih določb, ampak samo iztočnic za slednje.

Glede na to, da zagovarjamo čim krajše, čim bolj berljive, razumljive in nedvoumne licenčne pogodbe, predlagamo, da se vse, kar ne sodi neposredno v konkretni licenčni dogovor in velja za vse licenčne dogovore, zapiše v pravilih knjižnice, v licenci pa se pri teh določbah navedejo pravila kot referenca. Tipična taka določba je opredelitev pojmov, ki se uporabljajo v licenci. Predlagamo, da se v določbi o definicijah ti pojmi samo naštejejo, do njihovih opisov, ki so v pravilih knjižnice, pa uporabnik pride že prek (hiper)povezave ali se mu definicija izpiše v pravokotniku, ki se na zaslonu izriše, ko se z miško zaustavimo na pojmu. S tem je hkrati tudi že predpisan standard za terminologijo, ki se uporablja v povezavi s knjižnico in ponovnimi uporabami, kar je tudi zelo pomemben dosežek. Tudi izdelke in storitve knjižnice je po našem mnenju koristno opisati v pravilih, kajti pravila so prvi dokument in osebna izkaznica vsake knjižnice, s katero se srečajo čisto vsi potencialni kandidati za vpis v knjižnico. S podpisom pogodbe o včlanitvi hkrati podpišejo strinjanje s pravili in namero, da bodo ta pravila spoštovali. Tudi podrobnosti o izvedbi licenciranja je koristno zapisati v pravilih, tako da ne prihaja do nepotrebnih težav in zadreg, s tem pa do nepravilnosti, ko se pričnejo izvajati določeni samodejni mehanizmi licenciranja. Model je uporaben tako v primeru licenčne pogodbe med knjižnico in ponovnim uporabnikom kakor tudi med razvijalcem ali posrednikom komponente in knjižnico, zato sta v nadaljevanju predstavljeni obe različici. Pri posamezni določbi so navedene vse možnosti, ki lahko nastopijo v danem primeru, ni pa nujno, da bodo v vsaki licenčni pogodbi tudi vse potrebne,

saj so nekatere med seboj izključujejo, druge so spet odvisne od izbir pri predhodnih določbah. Če licencodajalec ne dovoli spreminjanja predmeta licence, ni smiselno, da v določbi o lastništvu navajamo, čigavo je lastništvo nad spremembami predmeta licence ipd.

Model naj bi bil izhodišče, na podlagi katerega bi se sestavili vzorci licenčnih pogodb za posamezne družine komponent. Družine bi se tvorile predvsem kot presek mesta znotraj klasifikacijske sheme in pogojev za licenciranje, ki jih določi dobavitelj komponente ob posredovanju komponente v knjižnico, po njih pa bi knjižnica razvrščala komponente v posamezne razrede (npr. črne, sive in bele komponente).

Za čim večjo učinkovitost knjižnice in tudi ponovne uporabe mora biti mehanizem licenciranja čim bolj avtomatiziran, dogovarjanje licenčnih pogojev pa lahko le izjemoma zahteva več iteracij. V nadaljevanju predlagan model predstavlja temeljno ogrodje, iz katerega so lahko izpeljani različni vzorci ponovno uporabnih licenčnih pogodb, ki se shranijo v podatkovno bazo.

Pri oblikovanju modela so bili uporabljeni naslednja sintaksa in pravila zapisa:

- Vsaka določba je označena z zaporedno številko in naslovom določbe. Zaporedje določb, podano v modelu, ne določa in ne predpisuje končnega zaporedja določb v pogodbi. Oznaka A se nanaša na določbe licence med knjižnico in ponovnim uporabnikom, oznaka B pa na določbe licence med dobaviteljem komponente in knjižnico.
- Posamezni elementi ali pogoji posamezne določbe sledijo v obliki seznama. Cilj, ki ga mora posamezen element doseči, je naveden v oklepajih (< >), izpisan pa je s poševnimi črkami.
- Ob koncu seznama elementov posamezne določbe je še navedeno, na kakšen način je vnesena vrednost za posamezen element:
  samodejno (vnese jo že program za oblikovanje licenc glede na trenutne vrednosti),
  ročno (v ustrezno polje je treba vpisati besedilo ali vrednost),
  vnaprejšnje besedilo (ko gre za besedilo, ki se pojavlja nespremenjeno v vseh licencah te knjižnice),
  izbiranje predloga (ko je na voljo omejena množica vrednosti za nek element, so te vrednosti podane kot izbirni seznam; najprej izbere vrednost licencodajalec,

licencojemalec pa izbiro bodisi potrdi bodisi zavrne tako, da možnosti iz istega seznama izbire drugo vrednost[174]) ali

kombinacija (ko gre delno za samodejen delno za ročen vnos).

6.1.7 Primer ogrodja licence za ponovno uporabno komponento, ki jo knjižnica podeljuje posameznemu ponovnemu uporabniku (A)

A1. Podatki o pogodbi ali oznake pogodbe

a) <*številka licenčne pogodbe*>[175]

b) < *datum*>

Vnos: a) samodejno[176], b) samodejno.

A2. Pogodbeni stranki

a) <*podatki o dajalcu licence ali knjižnici*>

b) <*podatki o pridobitelju licence, vključno z IŠČ (identifikacijsko številko člana)*>

c) <*navedba vrste licence: neizključna / izključna*>

Vnos: a) in b) samodejno, c) izbiranje predloga[177].

A3. Namen licence

a) <*poda se kratek zapis ciljev pogodbenih strank ter okoliščine sestave licence* >[178]

Vnos: a) vnaprejšnje besedilo.

---

[174] Da ima licencodajalec nadzor nad tem, kaj je sam predlagal ob začetku pogajanja, kaj pa predlaga ponovni uporabnik, naj bi se ob zavrnitvi pogoja izpisal poleg nov seznam z istimi možnostmi, med katerimi bi zbiral ponovni uporabnik. Število teh iteracij je smiselno omejiti. Če se licencodajalec in licencojemalec ne uspeta v določenem številu (npr. tri) iteracij dogovoriti, se postopek podeljevanja licence prekine.

[175] Številka pogodbe naj bi bila sestavljena iz treh delov: enega, ki se nanaša na identifikacijsko številko člana knjižnice, ki nastopa v vlogi licencojemalca, drugega, ki se nanaša na identifikacijsko številko komponente, tretjega pa na datum (ali leto) sklenitve licenčne pogodbe. Na tak način je možno zelo hitro in preprosto izvesti pregled vseh licenc nekega člana (npr. ob izstopu ali izpisu člana iz knjižnice) ali pregled vseh ponovnih uporabnikov neke komponente ali pa preveriti veljavnosti licenc za določene komponente (npr. ob zamenjavi komponente s popravljeno ali nadgrajeno komponento, izključitvijo določene komponente iz knjižnice ipd., ko je treba o tem obvestiti vse ponovne uporabnike neke komponente).

[176] Številka pogodbe se izpolni v trenutku, ko ponovni uporabnik po ogledu predmeta licence sproži postopek pridobivanja licence. Takrat se njegova identifikacijska številka, številka predmeta licence in trenutni datum sestavijo v eno številko, ki predstavlja enolično številko licenčne pogodbe.

[177] Vrednosti iz seznama se izključujeta, zato je lahko izbrana le ena.

[178] Opis okoliščin ali namena licence je zelo pomemben pri interpretaciji pogodbe na sodišču, v kolikor pride do spora med strankama. Ker so okoliščine in namen podelitve licence knjižnice pri vseh licencah enake, je ta določba lahko standardna in enaka pri vseh licencah. V kolikor bi pa nastopile res specifične okoliščine, se te lahko navedejo v določbi A22 – Posebnosti in izjeme.

A4. Sodišče, izbira prava

a) <*navedba prava, ki naj se uporabi pri interpretaciji licenčne pogodbe in sodišča, ki je pristojno za reševanje morebitnega spora*>

Vnos: a) vnaprejšnje besedilo.

A5. Definicije

a) <*naštejejo se posamezni termini, ki se uporabljajo v pogodbi, z referenco ali povezavo na pravila, kjer so podrobno opisani*>[179]

Vnos: a) vnaprejšnje besedilo.

A6. Predmet licence ali dogovor

a) <*identifikacijska številka komponente IŠK*>[180]

b) <*oznaka komponente po klasifikacijski shemi*>

c) <*opis posameznih elementov predmeta licence*[181] *s pripadajočimi cenami*>

Vnos: a) in b) samodejno, c) kombinacija[182].

A7. Podeljene pravice

a) <*seznam aktivnosti, ki jih lahko pridobitelj licence izvaja na predmetu licence oziroma seznam dovoljenih uporab predmeta licence*>[183]

b) <*navedba prenosljivosti ali neprenosljivosti licence*>

[179] Pravno pisanje zahteva uporabo zelo natančnega izrazoslovja. Velikokrat se za nek pojem, ki ga označuje večbesedna zveza, v pogodbi uporabi poseben krajši izraz, ki ga je treba v tem delu natančno razložiti. Navadno se med definicijami nahajajo tudi pravila licenciranja. Po mojem mnenju je tako pravila licenciranja kakor termine smiselno standardizirati in jih vključiti v pravila knjižnice.

[180] Predlagam, da se za vsako ponovno uporabno komponento oblikuje svoja licenčna pogodba, saj je sledenje ponovnim uporabam komponent na tak način olajšano. Ena licenčna pogodba za več komponent naj bi se uporabljala le izjemoma, ko gre za množico ponovno uporabnih komponent, nad katerimi spremembe niso dovoljene in ki se bodo uporabile znotraj istega programskega projekta pridobitelja licence pod istimi pogoji.

[181] Sestavni del predmeta licence so lahko računalniški programi, vsa dokumentacija in vsi pripadajoči podatki, potrebni za praktično uporabo predmeta licence (Obligacijski zakonik, čl. 708, str. 705-706) ter vsa navodila in obvestila, ki jih pridobitelj licence potrebuje za uspešno izkoriščanje predmeta licence (Obligacijski zakonik, čl. 709, str. 706). [116]

[182] Pripadajoči elementi so našteti v sami dokumentaciji komponente, zato se samodejno prepišejo, cene pa se vpišejo ročno.

[183] Predvsem je pomembno, da se določi, ali je spreminjanje predmeta licence dovoljeno ali ne in do kolikšne mere ali, kdaj lahko rečemo, da gre samo za prilagajanje, kdaj pa za spreminjanje ali celo izdelavo različice. Razliko med prilagajanjem in spreminjanjem definiram s tem, da pri prilagajanju ne sme priti do spreminjanja ali dodajanja funkcionalnosti komponenti, medtem ko pri spreminjanju lahko. Spet pa je pomembno, da se opredeli, kolikšno spreminjanje privede do nastanka nove različice. Da se izognemo poplavi komponent s podobno funkcionalnostjo v knjižnici, je po mojem mnenju priporočljivo, da knjižnica že v pravilih definira, kdaj gre za nastanek različice.

c) <*definicija pooblaščenih uporabnikov predmeta licence in njihovih pravic*>

d) <*določba o možnosti podeljevanja podlicenc*>

e) <*določba o izdelavi izpeljanih del iz predmeta licence*>[184]

Vnos: a) izbiranje predloga[185], b) izbiranje predloga[186], c) kombinacija[187], d) izbiranje predloga[188], e) izbira predloga[189].

A8. Omejitve uporabe

a) <*seznam aktivnosti, ki jih pridobitelj licence ne sme izvajati nad predmetom licence in ki se štejejo za kršitev pravic*>[190]

b) <*omejitev uporabe predmeta licence na en ali več razvojnih* >[191]

Vnos: a) izbiranje predloga ali kombinacija[192], b) izbiranje predloga in kombinacija[193].

A9. Lastništvo[194]

a) < *navedba lastništva na predmetu licence*>

[184] Licenca mora vključevati neizključno pravico do izdelave izpeljanih del iz licenciranega izdelka. Razvijalec predmeta licence in razvijalec izpeljanega dela si razdelita licenčnino glede na procent prispevka v izpeljanem delu. Odstotek določi knjižnica po formuli, ki je priložena kot referenca. Da se knjižnica izogne težavam s sledenjem uporabi posamezne komponente, ki je bila spremenjena, lahko določi, da postanejo tudi spremenjene komponente last knjižnice, ponovni uporabniki pa so lahko ustrezno finančno nagrajeni za posredovane spremenjene komponente. Enako velja za različice komponent.

[185] Iz seznama lahko izberemo več vrednosti – več aktivnosti ali dovoljenih uporab.

[186] Vrednosti iz seznama se izključujejo, zato je lahko izbrana le ena.

[187] Pooblaščeni uporabniki so vsi tisti, ki so trenutno na seznamu pooblaščenih uporabnikov organizacije, ki je član knjižnice in nastopa v vlogi ponovnega uporabnika, zato jih lahko sistem samodejno vnese. Obseg pravic, ki se prenašajo na pooblaščene uporabnike je zelo verjetno isti za vse, če pa ni tako, se te pravice vnesejo ročno za vsakega pooblaščenega uporabnika. Kaj se zgodi v primeru, da organizacija nekemu uporabniku vzame pooblastila ali mu jih vzame knjižnica zaradi kršitev in nespoštovanja pravil, preden poteče veljavnost licence, ki jo je pridobil skupaj s preostalimi pooblaščenimi uporabniki, se definira v samih pravilih.

[188] Seznam možnosti je kratek: podeljevanje podlicenc je dovoljeno ali ni dovoljeno. Če dopuščamo možnost, da je dovoljeno samo v določenih okoliščinah ali da samo v določenih okoliščinah ni dovoljeno, potem je treba dovoliti kombiniran vnos, saj potrebujemo še polje, v katerega zapišemo, za kakšne okoliščine gre.

[189] Izpeljava del se navadno dovoli, saj sicer ne moremo govoriti o ponovni uporabi.

[190] Gre za omejitve, ki jih postavi dajalec licence. Pri ponovno uporabnih komponentah se lahko pojavijo omejitve kakor npr. omejitve reproduciranja predmeta licence, nadaljnje distribucije, prodaje, posojanja, dajanja v najem ali podeljevanja podlicenc za komponento ipd. Dajalec licence lahko tudi prepove vsakršno spreminjanje in prilagajanje predmeta licence ter izdelavo različice komponente.

[191] Za uporabo v vsakem drugem projektu, ki je zunaj dovoljenega obsega, je potrebna nova pogodba ali aneks k obstoječi licenčni pogodbi.

[192] Načeloma bi moral biti seznam aktivnosti, ki jih na predmetu licence ne smemo izvajati, dokaj standarden, lahko pa se pripeti, da je treba dodati tudi katero zelo specifično aktivnost ali samo pri kateri od omejitev dodatno opredeliti, kdaj ta omejitev velja.

[193] Izbiramo med en projekt in več projektov. V vsakem primeru je treba poleg vsake možnosti zapisati podatke o projektu ali projektih, znotraj katerih je uporaba dovoljena.

[194] Lastniška pravica na komponenti lahko pripada razvijalcu komponente, knjižnici, razvijalcu aplikacije ali ponovnemu uporabniku komponente ali gre za solastništvo več strank. V slednjem primeru se v licenci navedejo solastniški deleži, ki so tudi temelj za delitev licenčnin iz naslova licenciranja teh komponent.

b) < *navedba lastništva na programskem izdelku, v katerega je predmet licence vgrajen - lastništvo izpeljanega dela*>

c) < *navedba lastništva na spremembah predmeta licence*>

d) < *navedba lastništva na različicah predmeta licence*>[195]

e) < *navedba lastništva na programskem izdelku, v katerega je vgrajena različica predmeta licence* >

Vnos: a) izbiranje predloga ali kombinacija[196], b) izbiranje predloga ali kombinacija[197], c) izbiranje predloga ali kombinacija[198], d) izbiranje predloga ali kombinacija[199], e) izbiranje predloga ali kombinacija[200].

A10. Dostava in dostop

a) <*datum dostave ali predaje predmeta licence*>

b) <*sredstvo, prek katerega bo predmet licence dostavljen pridobitelju licence*>

c) <*način dostopa*>[201]

Vnos: a) ročno , b) in c) izbiranje predloga.

A11. Podelitev licence

a) <*datum začetka veljavnosti licence*>[202]

b) <*opredelitev začetka veljavnosti licence*>[203]

c) <*določitev načina podpisa licence*>[204]

Vnos: a) samodejno[205], b) in c) izbiranje predloga.

---

[195] V kolikor so spremembe tolikšne, da že lahko govorimo o različici komponente, je treba določiti, čigava last bo različica. Lahko gre tudi za solastništvo med lastnikom komponente in razvijalcem, ki je različico izdelal.

[196] Število potencialnih lastnikov je minimalno, v primeru solastništva pa je treba opredeliti, v kolikšnem obsegu je kdo lastnik ali delež lastništva na komponenti vsakega posameznika.

[197] Če v elementu A7 e) izpeljava del ni mogoča, tudi tu ni treba ničesar opredeliti. Sicer pa velja, da je seznam možnih lastnikov zelo omejen in ročno dodamo samo solastniške deleže, če je treba.

[198] Velja isto kakor pri prejšnji opombi, le da je ta element nepotreben, ko med aktivnostmi pri A7 a) ni navedeno spreminjanje predmeta licence.

[199] Velja isto kakor pri prejšnji opombi.

[200] Vnos bo mogoč le, če je izdelava različice možna (v A7 a) je med dovoljenimi aktivnostmi tudi izdelava različic). Tudi v tem primeru je najverjetnejše, da pride do solastništva, zato je potrebna tudi možnost polja za ročni vnos solastniških deležev.

[201] Za nemoteno ponovno uporabo predmeta licence mora biti podeljen stalen dostop do predmeta licence, kar pomeni, da knjižnica kakor dajalec licence prejemniku licence posreduje elektronsko kopijo predmeta licence, ki jo ta shrani pri sebi in ima do nje stalen dostop v času veljavnosti licence.

[202] Če ni naveden, se šteje datum podpisa licenčne pogodbe.

[203] Začetek veljavnosti lahko nastopi, ko licenčno pogodbo podpiše prejemnik licence ali ko je predmet licence v celoti izročen prejemniku licence.

[204] Kot veljaven način za podpis licenčne pogodbe se šteje lahko elektronski podpis (z avtentifikacijsko številko) ali ročni ali osebni podpis (formular) ali oboje.

A12. Rok in prenehanje veljavnosti

a) <*rok veljavnosti licence*[206] *ali datum, ko licenca preneha veljati*>[207]

b) <*navedba okoliščin, ki lahko privedejo do prenehanja veljavnosti licence ter posledic prenehanja veljavnosti*>[208]

c) <*navedba o veljavnosti določenih določb licence po prenehanju veljavnosti licence*>

d) <*navedba časovnega roka za poravnavo obveznosti ob prekinitvi veljavnosti licence*>

Vnos: a) ročno, b) in c) vnaprejšnje besedilo[209], d) ročno.

A13. Licenčnina in način plačila

a) <*višina licenčnine in drugih stroškov v zvezi s pridobitvijo licence* >

b) <*kaj obsega licenčnina*>

c) <*česa licenčnina ne vključuje*>

d) <*način plačila*>[210]

Vnos: a) ročno, b) vnaprejšnje besedilo ali izbiranje predloga[211], c) isto kakor b), d) izbiranje predloga ali kombinacija[212].

A14. Obveznosti pridobitelja licence

a) <*spoštovanje določb licenčne pogodbe*>

b) <*izogibanje kršitvam pravic dajalca licence*>

---

[205] V trenutku, ko je licenca podpisana z obeh strani, se lahko vnese tekoči datum v to polje samodejno.

[206] Rok veljavnosti ali obdobje veljavnosti se nanaša na časovno obdobje, v katerem lahko pridobitelj licence uporablja predmet licence v skladu z določbami licenčne pogodbe.

[207] Rok veljavnosti je poljuben in se stranki zanj dogovorita. Vedno ga je mogoče podaljšati, lahko pa se veljavnost licence tudi prekine pred pretekom, če pride do spora med strankama ali kršenja licenčnih določil.

[208] Licencodajalec oziroma knjižnica lahko delno ali v celoti prekine licenčno pogodbo, če pridobitelj licence krši zahteve, določbe in pogoje iz te pogodbe (povzročanje materialne in moralne škode knjižnici, uporaba komponent in storitev knjižnice v nezakonite namene, poseganje v zasebnost in pravice preostalih članov knjižnice ipd.). Ob prekinitvi pogodbe mora pridobitelj licence:

- odpovedati se vsem pravicam uporabe predmeta licence,
- vrniti ali potrditi uničenje vseh kopij predmeta licence,
- licencodajalcu poravnati vse nastale stroške zaradi prekinitve licenčne pogodbe, če je do prekinitve prišlo zaradi njegovega kršenja določil pogodbe,
- licencodajalcu posredovati prejete licenčnine za morebitne nepooblaščene distribucije predmeta licence, ki jih je izvedel v času veljavnosti licence.

Če je krivda za prekinitev pogodbe na strani pridobitelja licence, mu dajalec licence ni dolžan vračati licenčnine, v nasprotnem primeru pa ja.

[209] Morebitne izjeme se dodajo v določbi A22 ali pa se tu opredeli vnos kakor vnaprejšnje besedilo in ročno.

[210] Možnost plačila je lahko v enkratnem znesku ali določen odstotek ob podpisu pogodbe, preostali del do določenega datuma ali kaj podobnega.

[211] Pri izbiranju s seznama se možnosti ne izključujejo.

[212] Pri zelo specifičnih predmetih licence in/ali visokih licenčninah in/ali pričakovanih visokih zaslužkih s prodajo izpeljanega dela iz predmeta licence je treba dovoliti možnost vnosa specifičnega načina plačila.

c) <*odpoved uporabi predmeta licence v namene, ki so zunaj uporabe, določene z licenčno pogodbo, brez dovoljenja ali pooblastila dajalca licence*>

d) <*sodelovanje z dajalcem licence v primeru ugotovitve kršitev nepooblaščenega uporabnika*>

Vnos: povsod je lahko vnaprejšnje besedilo.

A15. Varovanje predmeta licence

a) <*varovanje predmeta licence pred nepooblaščenimi uporabami ali uporabniki*>

b) <*obveščanje dajalca licence v primeru, ko je varnost predmeta licence ogrožena*>

c) <*omejevanje ali zmanjšanje posledic prelomitve varnostnih ukrepov, če pride do njih*>

Vnos: povsod je lahko vnaprejšnje besedilo.

A16. Nerazkritje[213]

a) <*seznam zaupnih informacij, ki jih stranki ne smeta razkriti, in okoliščin, v katerih se te informacije lahko razkrijejo*>

Vnos: a) izbiranje predloga.

A17. Jamstvo pravic (garancija)[214] in povračila škode[215] ali odškodnine

a) <*jamstva dajalca licence*>[216]

b) <*primeri, ko je pridobitelj licence dolžan povrniti škodo dajalcu licence*>[217]

c) <*primeri, ko je dajalec licence dolžan povrniti škodo pridobitelju licence*>[218]

[213] Po mojem mnenju gre lahko ta klavzula v pravila knjižnice, saj lahko velja ista klavzula za vse knjižnične komponente.

[214] Jamstvo ali garancija je neke vrste izjava ali potrditev, da neko dejstvo velja, in zagotovilo za izpolnitev neke obljube ali dejanja [148].

[215] Povračilo škode je tesno povezano z jamstvom, saj do njega pride, ko zagotovila iz jamstva niso bila izpolnjena ali je zaradi napak, nedelovanja ali napačnega delovanja prišlo do stroškov ali morda celo izgub pri plasiranju programskega izdelka, katerega sestavni del je tudi predmet licence, na trgu. Določene smernice o tem, kaj vključiti v škodo in kdaj lahko katera pogodbena stran zahteva povračilo škode, se lahko vključijo že v pravila knjižnice.

[216] Knjižnica, ki nastopa v vlogi dajalca licence, mora jamčiti, da bo predmet licence deloval tako, kakor je opisano v spremljajoči dokumentaciji, da je predmet licence ustrezen, da ima pooblastila za podelitev licence ali podlicence, da bo obdržala pravice intelektualne lastnine na predmetu licence v času trajanja licence ipd.

[217] To je vedno, ko je dajalcu licence s svojimi dejanji povzročil materialno ali druge vrsto škodo (npr. uporaba komponent in storitev knjižnice v nezakonite namene, poseganje v zasebnost in pravice preostalih članov knjižnice ipd.).

[218] Četudi gre vsaka komponenta pred vključitvijo v knjižnico skozi postopke certificiranja, preverjanja ponovne uporabnosti in ocenjevanja kakovosti se lahko pripeti, da pozneje, ob konkretni ponovni uporabi ponovni uporabnik, to je pridobitelj licence, ugotovi njeno nepravilno delovanje. V tem primeru bo seveda zahteval, da mu knjižnica kot dajalec licence povrne nastalo škodo vsaj za izgubljeni čas. Pridobitelj licence pa se mora zavarovati pred škodo, ki bi nastala kot posledica kršitve pravic intelektualne lastnine na predmetu licence s

d) <*postopki, ki jih je treba izpeljati pred oddajo odškodninskega zahtevka*>

e) <*navedba stroškov, ki jih je treba vračunati v znesek odškodnine*>[219]

f) <*navedba primerov, ki ne opravičujejo zahteve za povračilo škode*>

Vnos: a), b), c) , d) in f) vnaprejšnja besedila[220], e) izbiranje predloga ali vnaprejšnje besedilo.

A18. Vračilo licenčnine

a) <*pridobitelj licence drugače razume določbe in pogoje iz pogodbe kot dajalec licence*>

b) <*pridobitelj licence odstopa od uporabe predmeta licence*>[221]

Vnos: a) in b) vnaprejšnja besedila.

A19. Višja sila[222]

a) <*kršitev, ki je posledica višje sile, ne more biti vzrok za prekinitev licence*>

Vnos: a) vnaprejšnje besedilo.

A20. Poravnava v primeru spora

a) <*določitev načina poravnave spora*>[223]

Vnos: izbiranje predloga.

A21. Dodatki k pogodbi

a) <*definicije in navedbe podrobnosti, ki se jih v glavnem delu pogodbe ne navaja, da ne bi z njimi zakrili bistva pogodbe*>[224]

Vnos: ročno.

---

strani tretje osebe, pred tem, da je edini odgovoren za morebitni neuspeh ali nedelovanje programskega izdelka, v katerega je vgrajen predmet licence, pred morebitnim nezakonitim ravnanjem s predmetom licence s strani knjižnice ipd.

[219] V škodne stroške je treba všteti izgube, škode, stroške tožbe, licenčnino itn. Ti stroški lahko drastično presegajo znesek, ki ga mora pridobitelj licence plačati za pravice na predmetu licence, ki so mu podeljene z licenco.

[220] Izjeme opišemo v A22 ali pa pri teh elementih omogočimo tudi ročni vnos.

[221] Če pridobitelj licence po določenem času ugotovi, da predmeta licence ne bo mogel uporabiti, lahko načeloma odstopi od licence. V tem primeru mora pisno utemeljiti zavrnitev predmeta licence in vseh kopij, ki so morebiti nastale po podpisu licenčne pogodbe, ter zagotoviti, da predmet licence ni bil uporabljen v nobenem projektu ali izdelku.

[222] Z izrazom višja sila se v pravu poimenujejo pogoji, ki so zunaj nadzora pogodbenih strani. Mednje sodijo npr. vojne, sovražna dejanja, stavke, poplave, prekinitve električnega napajanja, uničenje mrežnih povezav, uničenje komunikacijskih povezav, državne omejitve (zavrnitev ali preklic izvozne ali druge licence ipd).

[223] Možni načini poravnave spora so na sodišču, s pomočjo arbitraže ali strokovnjaka – eksperta.

[224] Tu so lahko navedeni posamezni elementi predmeta licence, datumi, dostave – načini in oblike, prenosni mediji, lokacije, na katerih se lahko predmet licence uporablja ipd.

A22. Posebnosti in izjeme

a) <*seznam morebitnih posebnosti in izjem, povezanih z naravo predmeta licence, ki so zunaj nabora standardnih določb in pogojev*>

Vnos: ročno.

6.1.8 Primer ogrodja licence, ki jo dobavitelj ponovno uporabne komponente podeli knjižnici (B)

B1. Podatki o pogodbi ali oznake pogodbe

a) <*številka licenčne pogodbe*>[225]

b) < *datum*>

Vnos: a) in b) samodejno.[226]

B2. Pogodbeni stranki

a) <*podatki o dajalcu licence – razvijalcu komponente ali njenemu dobavitelju*>

b) <*podatki o knjižnici*>[227]

c) <*navedba neizključnosti licence*>[228]

Vnos: a) in b) samodejno, c) izbiranje predloga.

B3. Namen licence

a) <*poda se kratek zapis ciljev pogodbenih strank ter okoliščin sestave licence*>

Vnos: a) vnaprejšnje besedilo[229].

B4. Sodišče, izbira prava

a) <*navedba prava, ki naj se uporabi pri interpretaciji licenčne pogodbe in sodišča, ki je pristojno za reševanje morebitnega spora*>

Vnos: vnaprejšnje besedilo.

[225] Isto kot v opombi 214.
[226] Velja isto kakor pri A1 (opomba 215).
[227] V primeru, da se knjižnice združujejo v komponentna tržišča, dobi vsaka knjižnica svojo identifikacijsko številko.
[228] Licenca mora biti obvezno neizključna, saj mora razvijalcu ali dobavitelju komponente ostati odprta možnost nadaljnje ponovne uporabe te komponente in nadaljnjega trženja te komponente.
[229] Velja isto kakor pri A3 (opomba 217).

B5. Definicije

a) <*naštejejo se posamezni termini, ki se uporabljajo v pogodbi, z referenco na pravila, kjer so podrobno opredeljeni*>

Vnos: a) vnaprejšnje besedilo.

B6. Predmet licence ali dogovor

a) <*identifikacijska številka komponente IŠK*>[230]

b) <*oznaka komponente po klasifikacijski shemi* >

c) <*opis posameznih komponent in njihovih elementov s pripadajočimi cenami*>

Vnos: a) in b) samodejno, c) kombinacija[231].

B7. Podeljene pravice

a) <*določba o podeljevanju podlicenc*>[232]

b) <*seznam drugih aktivnosti, ki jih lahko pridobitelj licence izvaja na predmetu licence ali seznam dovoljenih uporab predmeta licence*>

c) <*navedba prenosljivosti ali neprenosljivosti licence*>

d) <*seznam pravic, ki jih lahko podeli s podlicenco*>

Vnos: c) izbiranje predloga[233], a), b) in d) izbiranje predloga ali kombinacija[234].

B8. Omejitve uporabe

a) <*seznam aktivnosti, ki jih pridobitelj licence ne sme izvajati nad predmetom licence in ki se štejejo za kršitev pravic*>[235]

Vnos: izbiranje predloga ali kombinacija[236].

[230] Predlagam, da se za vsako ponovno uporabno komponento oblikuje svoja licenčna pogodba ali kvečjemu ena licenčna pogodba za več komponent, ki se knjižnici posredujejo pod enakimi (licenčnimi) pogoji.
[231] Isto kakor pri A6 c) (opomba 221).
[232] Temeljna vloga knjižnice je ravno v tem, da s podlicencami trži komponente, ki so pod njenim okriljem. Torej je nujno, da je knjižnici podeljena pravica podlicenciranja. Če je knjižnica del virtualnega komponentnega tržišča, je treba natančno določiti, ali lahko podeljuje podlicence samo končnim ponovnim uporabnikom ali tudi preostalim knjižnicam za nadaljnje podlicenciranje. Slednje je tudi tesno povezano s prenosljivostjo ali neprenosljivostjo licence.
[233] Isto kakor pri A7 a) (opomba 224).
[234] Če je treba navajati specifične okoliščine, potrebujemo možnost ročnega vnosa.
[235] Gre za omejitve, ki jih postavi dajalec licence. Pri ponovno uporabnih komponentah se lahko pojavijo omejitve, na primer omejitve reproduciranja predmeta licence, nadaljnje distribucije, prodaje, posojanja, dajanja v najem ali podeljevanje podlicenc za komponento ipd. Dajalec licence lahko tudi prepove vsakršno spreminjanje in prilagajanje predmeta licence in izdelavo različice komponente.
[236] Isto kakor A8 a) (opomba 231).

B9. Lastništvo[237]

a) <*navedba lastništva na predmetu licence*>

b) < *navedba lastništva na izdelku, v katerega je vgrajen predmet licence*>[238]

c) < *navedba lastništva na spremembah predmeta licence*>[239]

d) < *navedba lastništva na različicah predmeta licence*>[240]

Vnos: a) izbira predloga ali kombinacija[241], b) izbira predloga ali kombinacija[242], c) izbira predloga ali kombinacija[243], d) izbira predloga ali kombinacija[244].

B10. Dostava in dostop

a) <*datum dostave ali predaje predmeta licence*>

b) <*sredstvo, prek katerega bo predmet licence dostavljen pridobitelju licence*>[245]

Vnos: a) ročno, b) izbira predloga.

B11. Podelitev licence

a) <*datum začetka veljavnosti licence*>[246]

b) <*opredelitev začetka veljavnosti licence* >

c) <*določitev načina podpisa licence* >

Vnos: a) samodejno[247], b) in c) izbiranje predloga[248].

[237] Lastniška pravica na komponenti lahko pripada razvijalcu komponente ali knjižnici ali gre za solastništvo obeh. V slednjem primeru se v licenci navedejo solastniški deleži, ki so tudi osnova za delitev licenčnin iz naslova licenciranja teh komponent.

[238] Dobavitelj komponente ali njen razvijalec lahko ob posredovanju komponente v knjižnico vnaprej določi, ali zahteva kakršnokoli lastniško pravico na izdelku, izpeljanem iz njegove komponente.

[239] Isto kakor pri prejšnji opombi velja tudi za lastniško pravico na spremembah, ki jih ponovni uporabnik izvede nad komponento. Če je sprememb veliko, potem je smiselno govoriti že o različici komponente, če pa jih je ravno toliko, da je komponenta po teh spremembah prilagojena za ponovno uporabo, potem nima smisla "drobiti" lastništva.

[240] Če so spremembe tolikšne, da že lahko govorimo o različici komponente, je treba določiti, čigava last bo različica.

[241] Isto kakor A9 a) (opomba 235).

[242] Isto kakor A9 b) (opomba 236).

[243] Isto kakor A9 c) (opomba 237), le da morajo biti spremembe omogočene znotraj elementa B7 a).

[244] Isto kakor A9 d) opomba 238), le da mora biti izdelava različic vključena med dovoljene aktivnosti pod B7 a).

[245] Knjižnica se lahko odloči za fizično shranjevanje le enega primerka predmeta licence ter varnostne kopije, ob vsaki podeljeni podlicenci pa od dobavitelja komponente zahteva, da dostavi ustrezno kopijo neposredno ponovnemu uporabniku, ki bo zanjo plačal licenčnino. Tako se sicer izogne vzpostavljanju dodatnih varnostnim mehanizmov za varen prenos podatkov in dejansko odigra le posredniško vlogo, lahko pa pride do zakasnitev ali napak pri realizaciji licenčne pogodbe (prekoračen rok dobave, potrebna so vzporedna obveščanja knjižnici, kdaj je bil predmet licence dostavljen ipd.), kar gotovo ne prispeva k ugledu knjižnice. Zato predlagam, da se knjižnici podeli pravica izdelave kopije komponente vsakič, ko pride do sklenitve licenčne pogodbe. V primeru, da dobavitelj posreduje knjižnici kriptirano komponento, se v tej določbi opiše način in postopek kriptiranja.

[246] Če ni naveden, se šteje datum podpisa licenčne pogodbe.

B12. Rok in prenehanje veljavnosti

a) <*rok veljavnosti licence*[249] *ali datum, ko licenca preneha veljati*>[250]

b) <*navedba okoliščin, ki lahko privedejo do prenehanja veljavnosti licence ter posledic prenehanja veljavnosti*>[251]

c) <*navedba o veljavnosti določenih določb licence po prenehanju veljavnosti licence*>

d) <*navedba časovnega roka za poravnavo obveznosti ob prekinitvi veljavnosti licence*>

Vnos: a) ročno, b) in c) vnaprejšnje besedilo[252], d) ročno.

B13. Licenčnina in način plačila

a) <*višina licenčnine in drugih stroškov v zvezi s pridobitvijo licence, ki jih mora plačati pridobitelj licence*>

b) <*kaj obsega licenčnina*>

c) <*česa licenčnina ne vključuje*>

d) <*način plačila: v enkratnem znesku ali določen odstotek ob podpisu pogodbe, preostali del do določenega datuma*>

Vnos: a) ročno, b) vnaprejšnje besedilo ali izbiranje predloga[253], c) isto kakor b), d) izbiranje predloga ali kombinacija[254].

---

[247] Isto kakor vnos A11 a) (opomba 244).
[248] Isto kakor A11 b) in c) (opombi 242 in 243).
[249] Rok veljavnosti ali obdobje veljavnosti se nanaša na časovno obdobje, v katerem lahko pridobitelj licence uporablja predmet licence v skladu z določbami licenčne pogodbe.
[250] Rok veljavnosti je poljuben in se stranki zanj dogovorita. Vedno ga je mogoče podaljšati, lahko pa se veljavnost licence tudi prekine pred pretekom, če pride do spora med strankama ali kršenja licenčnih določb.
[251] Licencodajalec lahko delno ali v celoti prekine licenčno pogodbo, če pridobitelj licence krši zahteve, določbe in pogoje iz te pogodbe. Pridobitelj licence ravno tako, le da se mora pridobitelj licence zavarovati tudi v primeru, ko po sprejemu komponente v knjižnico ugotovi nepravilnosti v njenem delovanju in kršitve pravic intelektualne lastnine na tej komponenti (lažno avtorstvo, zatajitev patenta ali postopka pridobivanja patenta za komponento ali kaj podobnega). Če dogovor prekine pridobitelj licence, mora le-ta:
- odpovedati se vsem pravicam nadaljnje distribucije predmeta licence,
- vrniti ali potrditi uničenje vseh kopij predmeta licence,
- obvestiti vse prejemnike licence za komponento (ponovne uporabnike te komponente), ki je knjižnica ne bo več licencirala in
- ustrezen prenos medsebojnih obveznosti neposredno na dobavitelja komponente ali licencodajalca iz te pogodbe ali pa zamrznitev nadaljnje distribucije ter izključitev komponente iz knjižnice po preteku veljavnosti najkasnejše licenčne pogodbe za to komponento,
- licencodajalcu poravnati vse nastale stroške v primeru neutemeljene prekinitve licenčne pogodbe ali prekinitve, katere povod ni kršitev s strani licencodajalca.

Če se pogodba prekine zaradi kršitev njenih določb s strani pridobitelja licence, ni povračila denarja, ne glede na to, katera stran je pogodbo prekinila.
[252] Morebitne izjeme se opredelijo v B22.
[253] Isto kakor A13 b) (opomba 250).
[254] Isto kakor A13 d) (opomba 251).

B14. Obveznosti pridobitelja licence

a) <*spoštovanje določb licenčne pogodbe*>

b) <*izogibanje kršitvam pravic dajalca licence*>

c) <*odpoved uporabi predmeta licence v namene, ki so zunaj uporabe, določene z licenčno pogodbo, brez dovoljenja ali pooblastila dajalca licence*>

d) <*sodelovanje z dajalcem licence v primeru ugotovitve kršitev s strani nepooblaščenega uporabnika*>

Vnos: a), b), c) in d) vnaprejšnje besedilo.

B15. Varovanje predmeta licence

a) <*varovanje predmeta licence pred nepooblaščenimi uporabami ali uporabniki*>

b) <*obveščanje dajalca licence (knjižnice), če je varnost predmeta licence ogrožena*>

c) <*omejevanje ali zmanjšanje posledic preloma varnostnih ukrepov, v kolikor do njih pride*>

Vnos: a), b) in c) vnaprejšnje besedilo.

B16. Nerazkritje[255]

a) <*seznam zaupnih informacij, ki jih stranki ne smeta razkriti in okoliščin, v katerih se te informacije lahko razkrijejo*>

Vnos: a) izbiranje predloga.

B17. Jamstvo pravic (garancija)[256] in povračila škode[257] ali odškodnine

a) <*jamstva dajalca licence*>[258]

b) <*primeri, ko je pridobitelj licence dolžan povrniti škodo dajalcu licence*>[259]

[255] Klavzula gre lahko v pravila knjižnice, saj lahko velja ista klavzula za vse knjižnične komponente.
[256] Jamstvo ali garancija je neke vrste izjava ali potrditev, da neko dejstvo velja ali zagotovilo za izpolnitev neke obljube ali dejanja [148].
[257] Povračilo škode je tesno povezano z jamstvom, saj do njega pride, ko zagotovila iz jamstva niso bila izpolnjena ali je zaradi napak, nedelovanja ali napačnega delovanja prišlo do stroškov ali morda celo izgub v poslovanju knjižnice (povračila škode in licenčnin ponovnim uporabnikom takih komponent). Določene smernice o tem, kaj vključiti v škodo in kdaj lahko katera pogodbena stran zahteva povračilo škode, se lahko vključijo že v pravila knjižnice.
[258] Posrednik ali razvijalec komponente mora jamčiti, da bo predmet licence deloval tako, kakor je opisano v spremljajoči dokumentaciji, da je predmet licence ustrezen, da niso kršene nobene pravice intelektualne lastnine na predmetu licence ipd.
[259] Pridobitelj licence je dolžan škodo povrniti samo v primeru, ko ni spoštoval določb licenčne pogodbe, ko je namerno preprečeval ponovno uporabo predmeta licence svojim ponovnim uporabnikom, ko je v zvezi s

c) <*primeri, ko je dajalec licence dolžan povrniti škodo pridobitelju licence*>[260]

d) <*postopki, ki jih je treba izpeljati pred oddajo odškodninskega zahtevka*>

e) <*navedba stroškov, ki jih je treba vračunati v znesek odškodnine*>[261]

f) <*navedba primerov, ki ne opravičujejo zahteve za povračilo škode*>

Vnos: a), b), c) , d) in f) vnaprejšnja besedila[262], e) izbiranje predloga ali vnaprejšnje besedilo[263].

B18. Vračilo licenčnine

a) <*pridobitelj licence drugače razume določbe in pogoje iz pogodbe kakor dajalec licence*>

b) <*pridobitelj licence odstopa od trženja predmeta licence*>[264]

Vnos: a) in b) vnaprejšnje besedilo.

B19. Višja sila[265]

a) <*kršitev, ki je posledica višje sile, ne more biti vzrok za prekinitev licenčnega dogovora*>

Vnos: a) vnaprejšnje besedilo.

B20. Poravnava v primeru spora

a) <*določitev načina poravnave spora*>[266]

Vnos: a) izbiranje predloga.

B21. Dodatki k pogodbi

a) <*definicije in navedbe podrobnosti, ki se jih v glavnem delu pogodbe ne navaja, da ne bi z njimi zakrili bistva pogodbe*>[267] Vnos: ročno.

---

predmetom licence objavljal neresnične, zavajajoče in škodljive informacije ali kakorkoli drugače povzročil škodo dajalcu licence.

[260] Ob ugotovitvi knjižnice, da so v komponenti hujše nepravilnosti, in licencodajalčevem nespoštovanju določb o odpravi teh nepravilnosti, zavajajočih in nepravilnih spremljajočih informacijah, lažnih podatkih o avtorstvu ali lastništvu na predmetu licence ipd., s čimer ustvari licencodajalec knjižnici določene stroške, je te stroške ter nastalo škodo dolžan povrniti, skupaj z že prejetimi licenčninami, ki jih mora knjižnica vrniti ponovnim uporabnikom, s katerimi se prekinejo licenčne pogodbe.

[261] V škodne stroške je treba všteti izgube, škode, stroške tožbe, licenčnino itn. Ti stroški lahko drastično presegajo znesek, ki ga mora pridobitelj licence plačati za pravice na predmetu licence, ki so mu podeljene z licenco.

[262] Isto kakor A17 a), b), c), d) in f) (opomba 259).

[263] Nekateri tipični primeri se lahko navedejo v seznamu, pustimo pa možnost vnosa specifičnih primerov.

[264] Če pridobitelj licence – knjižnica po določenem času ugotovi, da ne bo uspela uspešno tržiti predmeta licence, lahko prekine pogodbo brez obveznosti plačila odškodnine, ampak samo vrne morebitno zaračunano licenčnino.

[265] Isto kakor A19 (opomba 261).

[266] Isto kakor A20 (opomba 262).

B22. Posebnosti in izjeme

a) *<seznam morebitnih posebnosti in izjem, povezanih z naravo predmeta licence, ki so zunaj nabora standardnih določb in pogojev>*

Vnos: a) ročno.

6.2 POGODBA O VČLANITVI

Včlanitev v knjižnico je predpogoj tako za dobavitelje komponent kakor za ponovne uporabnike komponent, če želijo sodelovati s knjižnico. Namen tega dogovora je na eni strani obvezati upravitelja knjižnice, da bo odgovorno skrbel za učinkovito delovanje ter vzdrževanje knjižnice, na drugi strani pa obvezati člane, da bodo pri uporabi knjižnice in njenih storitev spoštovali pravila knjižnice. Pravila knjižnice morajo biti sestavni del pogodbe o včlanitvi, saj natančno opredeljujejo način in koncept delovanja knjižnice, postopke in mehanizme knjižnice ter obveznosti tako knjižnice do članov kakor tudi članov do knjižnice.

Postopek sklepanja pogodbe o včlanitvi je relativno preprosto avtomatizirati, saj gre v bistvu za vnaprej pripravljeno pogodbo, ki jo potencialni član lahko bodisi podpiše bodisi ne podpiše (t.i. "*click-wrap"* pogodbo). Samo v primeru določbe *1 – Podatki o članu* ter določbe *4 – Nivo dostopa do virov knjižnice* gre za vpis konkretnih podatkov pri vsakokratnem vpisu novega člana. V določbi 1 se podatki pridobijo iz zahtevka za včlanitev v knjižnico, ki ga potencialni član pošlje knjižnici, v drugem primeru pa knjižnica dodeli vsakemu članu in njegovim pooblaščenim uporabnikom nivo dostopa[268]. Besedilo vseh drugih določb pa se ne spreminja glede na to, kdo se včlanjuje.

Temeljni elementi takega dogovora morajo zajemati naslednje določbe:

1 - Podatki o članu

a) *<podatki o članu – organizaciji in vseh njenih pooblaščenih uporabnikih>*

[267] Tu so lahko navedeni posamezni elementi predmeta licence, datumi, dostave – načini in oblike, prenosni mediji, lokacije, na katerih se lahko predmet licence uporablja ipd.
[268] Glej 5. poglavje.

2 - Definicije

a) <*definirajo se posameznimi termini, ki se uporabljajo v pogodbi, in se navede, da so termini opredeljeni v pravilih*>

3 - Sodišče, izbira prava

a) <*navedba prava, ki naj se uporabi pri interpretaciji pogodbe, in sodišča, ki je pristojno za reševanje morebitnega spora med članom in knjižnico*>

b) <*določitev načina poravnave spora*>

4 - Nivo dostopa do virov knjižnice

a) <*dodeljen začetni nivo dostopa za konkretnega člana in njegove pooblaščene uporabnike*>[269]

5 - Obseg storitev in izdelkov knjižnice

a) <*opis izdelkov in storitev, ki jih knjižnica ponuja, če pa so ti opredeljeni v pravilih, se jih samo našteje ali se kakor referenca zapišejo pravila knjižnice*>[270]

6 - Odgovornosti člana

a) <*posredovanje verodostojnih podatkov*>[271]

b) <*spoštovanje in upoštevanje pravil knjižnice, ki so temeljni in sestavni del dogovora o včlanitvi, ter spremljanje sprememb teh pravil*>[272]

c) <*pridobitev ali namestitev vse potrebne opreme za dostop do knjižnice in njenih storitev*>

d) <*redno plačevanje članarine/uporabnine in vseh drugih finančnih obveznosti*>

e) <*zakonita in varna uporaba knjižničnega materiala in storitev*>

f) <*varovanje poslovnih skrivnosti, povezanih z delovanjem knjižnice*>

[269] Nivo dostopa se pozneje lahko tudi spremeni, glede na "obnašanje" člana. Odločitev sprejme programski svet knjižnice.

[270] Med izdelke in storitve sodijo geslo in identifikacijska številka člana, elektronska podatkovna baza s komponentami, programska oprema za iskanje in dostop do elementov knjižnice, sistem za podporo finančnim transakcijam (sistem obračunavanj), sistem pomoči uporabnikom, sistem certificiranja komponent, sistem kvalificiranja komponent, sistem vzdrževanja podatkovnih baz s podatki o ponovnih uporabah, sistem (avtomatskega) elektronskega licenciranja, mehanizem spremljanja delovanja knjižnice in povratnega obveščanja, elektronska oglasna deska, sistem elektronske pošte, sistem vzdrževanja elementov in storitev knjižnice itn.

[271] Predvsem pomembno je, da dobavitelji komponent posredujejo verodostojne podatke o samem delovanju komponent ter avtorstvu ali lastništvu komponent.

[272] Knjižnica ima pravico v vsakem trenutku spremeniti svoja pravila, mora pa o tem obvestiti vse svoje člane.

g) <*obveščanje knjižnice o morebitnih kršitvah, nepooblaščenih dostopih in drugih nepravilnostih, za katere izve ali jim je priča*>

7 - Odgovornosti knjižnice

a) <*zagotavljanje učinkovitega, kakovostnega, varnega, sodobnega delovanja in vzdrževanja knjižnice*>

b) <*zagotavljanje zakonite uporabe knjižničnega materiala in storitev*>

c) <*varovanje osebnih podatkov članov*>

d) <*zagotavljanje obveščenosti o vseh spremembah*[273]>

e) <*zagotavljanje ažurnega obračunavanja in posredovanja licenčnin, stimulacij, finančnih nagrad, povračil stroškov in drugih denarnih zneskov, do katerih so upravičeni člani knjižnice*>

8 - Prenehanje veljavnosti pogodbe

a) <*navedba pravice vsakega člana, da lahko kadarkoli prekine svoje članstvo*>

b) <*obveze, ki jih mora izpolniti član ob samodejni prekinitvi članstva*>[274]

c) <*obveze, ki jih mora knjižnica izpolniti do člana, ki se izpisuje*>[275]

d) <*pogoji, pod katerimi lahko knjižnica zahteva izpis člana*>[276]

e) <*obveze, ki jih mora izpolniti član, ko izpis zahteva knjižnica*>[277]

[273] Spremembe so lahko povezane z licencami, s komponentami, dobavitelji, ponovnimi uporabniki, sistemom ali pravili delovanja knjižnice, z napakami, težavami idr.

[274] Če gre za dobavitelja komponent, se mora s knjižnico dogovoriti o tem, da ostanejo njegove komponente v knjižnici do datuma trajanja zadnje licence, ki je bila podeljena za njegove komponente, dolžan pa je še nek garancijski rok (npr. 1 leto) ponujati morebitno vzdrževanje in pomoč uporabnikom, ki so plačali licenčnino za njegove komponente in so do take pomoči upravičeni, če se s knjižnico ne dogovori za prevzem obveznosti vzdrževanja s podpisom ustrezne pogodbe o vzdrževanju. Če gre za ponovnega uporabnika, se z njim prekine sklepanje nadaljnjih licenčnih pogodb in se mu onemogoči dostop do vseh virov knjižnice, dolžan pa je poravnati vse finančne obveznosti. Članstvo se torej začasno zamrzne, dokončno pa se prekine potem, ko poteče rok veljavnosti zadnji licenčni pogodbi, ki jo je sklenil s knjižnico.

[275] V primeru, da član izstopa zaradi nespoštovanja pravil in določb pogodbe s strani same knjižnice (nekakovostni izdelki in storitve ipd.), mora knjižnica članu poravnati škodo, ki je nastala zaradi njene malomarnosti, ter vrniti uporabnino oziroma članarino, ki jo je plačal ob vpisu.

[276] Knjižnica lahko prekine pogodbo, če član knjižnici povzroča materialno škodo, uporablja izdelke in storitve knjižnice v nelegalne namene, poskuša posegati ali posega v zasebnost ali pravice kogarkoli drugega, ruši integriteto in ugled knjižnice ipd.

[277] Velja isto kakor pri samodejnem izpisu člana, le da lahko knjižnica v primeru hujše kršitve ponovnega uporabnika tudi takoj prekine vse veljavne licenčne pogodbe in zahteva vrnitev komponent, morda celo povračilo škode ter izjavo, da ne bo uporabljal ali kakorkoli izkoriščal ničesar, kar je povezanega s komponentami, za katere je v času članstva pridobil licence. V primeru hujših kršitev dobavitelja komponent se lahko prekinejo vsi finančni prilivi na njegov račun, poleg tega pa se mu zaračuna še nastala škoda.

9 - Višja sila

a) *<kršitve ali škode, ki so posledica višje sile, ne morejo biti vzrok za prekinitev pogodbe>*[278]

6.3 POGODBA O VZDRŽEVANJU

Knjižnice so lahko najbolj učinkovite takrat, ko hranijo preskušene in vzdrževane komponente. Za preskušanje komponent in njihovo certificiranje knjižnica poskrbi že pred vključitvijo komponente v knjižnico, prav pa bi bilo, da bi se takrat definiralo tudi njihovo vzdrževanje. Namreč, veliko prihrankov v razvoju programskih sistemov na temelju ponovne uporabe izhaja ravno iz dejstva, da razvijalcem ali ponovnim uporabnikom ni potrebno vzdrževati ponovno uporabljenih komponent.

Najbolj razumljivo bi bilo, da bi se knjižnica za vzdrževanje dogovorila kar z dobavitelji ali razvijalci komponent. S podpisom licenčne pogodbe za komponento, ki jo dobavitelj posreduje v knjižnico, ali vzporedno pogodbe o vzdrževanju se dobavitelj obveže, da bo skrbel za pravočasno posodabljanje in po potrebi (glede na zahteve ali pripombe ponovnih uporabnikov) nadgrajevanje komponente, odstranitev morebitnih napak in nepravilnosti (v obdobju garancije brezplačno) ter morebiti izvedel konkretne prilagoditve za posamezne ponovne uporabnike, v kolikor dobavitelj ne želi podeliti ponovnemu uporabniku pravice do spreminjanja in prilagajanja njegove komponente. Knjižnica je v tem primeru kar v veliki meri odvisna od dobavitelja ali razvijalca. Zato se mora ustrezno zavarovati pred njegovim morebitnim nekorektnim obnašanjem in nespoštovanjem pogodbenih določil ter poskrbeti za ustrezne ukrepe. Za knjižnico to predstavlja veliko tveganje. Slabe izkušnje, ki bi jih imeli ponovni uporabniki zaradi slabega vzdrževanja komponent, bi senco najprej vrgle na knjižnico kot na neposrednega dobavitelja ponovno uporabne komponente, kar lahko pomeni tudi dovolj močan razlog za izpis ponovnega uporabnika iz knjižnice.

Zato se knjižnica lahko odloči, da bo sama vzdrževala komponente, ki jih bo hranila. Če se seveda dobavitelji komponent s tem strinjajo, morajo podpisati ustrezen dogovor, pogodbo o

[278] Isto kakor pri A19 (opomba 261).

vzdrževanju. To pa hkrati za knjižnico pomeni velik dodaten strošek in zmanjšan dobiček od "prodaje" svojih komponent. Zato so dobavitelji prisiljeni ceno za komponento pred sprejemom v knjižnico ustrezno znižati, hkrati pa mora knjižnica zahtevati tudi vnaprejšnji dogovor o deležu licenčnine, ki jo bodo ponovni porabniki te komponente posredovali knjižnici in ga bo knjižnica zadržala za vzdrževanje.

Načeloma bi bilo možno tudi, da se za vzdrževanje knjižnica dogovori s posameznimi ponovnimi uporabniki komponent, torej se v tem primeru način izvedbe in plačila vzdrževanja dogovorijo v licenčni pogodbi med knjižnico in ponovnim uporabnikom ali pa se med njima sklene pogodba o vzdrževanju, ko nastopijo okoliščine, ki zahtevajo vzdrževanje licencirane komponente.

Vsi ti medsebojni odnosi in obveznosti, ki so kakorkoli povezani z vzdrževanjem ponovno uporabnih komponent, se opredelijo v pogodbi o vzdrževanju.

1 - Podatki o vzdrževalcu

a) <*podatki o članu, ki s knjižnico sklepa dogovor o vzdrževanju*>[279]

2 - Definicije

a) <*definirajo se posameznimi termini, ki se uporabljajo v pogodbi in se navede, da so termini opredeljeni v pravilih*>

3 - Sodišče, izbira prava

a) <*navedba prava, ki naj se uporabi pri interpretaciji pogodbe in sodišča, ki je pristojno za reševanje morebitnega spora med pogodbenima stranema*>

b) <*določitev načina poravnave spora*>

4 - Obseg aktivnosti vzdrževanja

a) <*podatki o komponenti (IDK), nad katero se bo izvedlo vzdrževanje*>

b) <*opis storitev ali aktivnosti, ki so predmet pogodbe; lahko se samo naštejejo in se kot referenca zapišejo pravila knjižnice, v katerih so podrobneje opisane*>

[279] V vsakem primeru mora biti vzdrževalec član knjižnice in predhodno podpisati izjavo o varovanju poslovne skrivnosti.

5 - Rok za izvedbo vzdrževalnih del

a) *<določitev roka za zaključek vzdrževalnih del>*[280]

6 - Plačilo vzdrževalnine

a) *<določitev višine vzdrževalnine>*

b) *<določitev načina plačila>*[281]

7 - Odgovornosti vzdrževalca

a) *<posredovanje verodostojnih podatkov>*

c) *<spoštovanje in upoštevanje pravil knjižnice, ki so temeljni in sestavni del dogovora o vzdrževanju>*

d) *<spoštovanje rokov za izvedbo vzdrževalnih del>*

e) *<obveščanje knjižnice o morebitnih kršitvah in nepravilnostih, ki so se zgodile na komponenti med njeno ponovno uporabo ali v času njenega vzdrževanja>*

8 - Odgovornosti knjižnice

a) *<zagotavljanje pravočasnih in verodostojnih informacij, ki so potrebne za kakovostno in čim hitrejšo izvedbo vzdrževalnih del>*

b) *<zagotavljanje obveščenosti preostalih članov knjižnice o vseh spremembah, povezanih z vzdrževanimi komponentami, npr. dodatna funkcionalnost, nadgradnja, odstranitev komponente iz knjižnice idr.>*

c) *<zagotavljanje ažurnega obračunavanja in posredovanja denarnih nadomestil za vzdrževalna dela zunaj garancijskega obdobja>*

9 - Prenehanje veljavnosti pogodbe

a) *<navedba pravice vzdrževalca, da lahko kadarkoli prekine pogodbo>*

b) *<obveze, ki jih mora izpolniti vzdrževalec ob samodejni prekinitvi pogodbe>*[282]

c) *<obveze, ki jih mora knjižnica izpolniti do vzdrževalca, s katerim prekinja pogodbo>*[283]

[280] Če obsega vzdrževanje več različnih aktivnosti, ki lahko različno dolgo trajajo, je treba rok navesti za vsako aktivnost posebej.

[281] Vzdrževalnina se lahko izplača v celoti po zaključenih vzdrževalnih delih, lahko se delno plača vnaprej, preostali del po zaključenem vzdrževanju, lahko se obračuna kakor letno pavšalno plačilo ipd.

[282] Če vzdrževalec prekinja pogodbo brez posebnega razloga, gotovo pa ne po krivdi knjižnice, nima knjižnica do njega nobenih obveznosti, razen da mu izplača do tedaj zasluženo vzdrževalnino. Če pa pogodbo prekinja zaradi nekorektnosti knjižnice, lahko vzdrževalec od knjižnice zahteva plačilo morebitne odškodnine, ki je nastala kot posledica nekorektnega obnašanja knjižnice.

d) <*pogoji, pod katerimi lahko knjižnica zahteva prekinitev pogodbe*>[284]

e) <*obveze, ki jih mora izpolniti vzdrževalec, ko prekinitev zahteva knjižnica*>[285]

10 - Višja sila

a) <*kršitve ali škode, ki so posledica višje sile, ne morejo biti vzrok za prekinitev pogodbe*>[286]

6.4 POGODBE IN LICENCE V MEDKNJIŽNIČNEM OKOLJU

Odpiranje knjižnice navzven in povezovanje z drugimi knjižnicami zahteva vzpostavitev še dodatnih pravnih varoval, ki jih je smiselno realizirati v obliki specifičnih pogodb.

Med konceptom delovanja samostojne knjižnice, ki sem ga predstavila v 5. poglavju, in konceptom medknjižničnega povezovanja obstaja precejšnja analogija. Ko se knjižnica "seli" v medknjižnično okolje, igra vlogo člana, ki se vpisuje v večjo knjižnico, njeni pooblaščeni uporabniki pa so vsi njeni člani. Ko želi član neke knjižnice ponovno uporabiti neko komponento, ki je v drugi knjižnici, njegova matična knjižnica (knjižnica, v katero je včlanjen) posreduje pri pridobitvi licence za to komponento, sama pa jo naprej podeli svojemu članu. Za uporabnika mora biti pridobitev licence za komponento enako zahtevna, to je čim bolj preprosta, ne glede na to, v kateri knjižnici spletnega tržišča se komponenta nahaja.

6.4.1 Licenčna pogodba med dvema knjižnicama

Licence, ki si jih knjižnice podeljujejo med seboj, morajo torej obvezno omogočati podlicenciranje. To je tudi največja razlika med licenčnimi pogodbami v okolju samostojne

[283] Če član izstopa zaradi nespoštovanja pravil in določb pogodbe s strani knjižnice, mora knjižnica članu poravnati škodo, ki mu je nastala zaradi njene malomarnosti ter poleg članarine, ki jo je plačal ob vpisu, vrniti tudi stroške, nastale z do tedaj opravljenim vzdrževanjem.

[284] Knjižnica lahko prekine pogodbo, če vzdrževalec ne spoštuje rokov in ostalih obveznosti iz pogodbe in s tem povzroča škodo tako knjižnici kot ponovnim uporabnikom komponente.

[285] Vzdrževalec je dolžan prenesti na knjižnico pravico do vzdrževanja komponente in s tem omogočiti nadaljnjo ponovno uporabo komponente, ali pa sprožiti postopek odstranitve komponente iz knjižnice, če je hkrati njen dobavitelj. Dolžan pa je knjižnici in članom knjižnice, ki so imeli zaradi neizpolnjenega vzdrževanja komponente, za katero so plačali licenčnino, stroške, te stroške povrniti.

[286] Isto kakor pri A19 (opomba 261) .

knjižnice in licenčnimi pogodbami v okolju virtualnega tržišča komponent. Naslednja razlika se nanaša na to, kdo sta pogodbeni strani. Če sta pogodbeni stranki knjižnica in njen član, potem gre za navadno licenciranje, opisano z ogrodjem A. Če pa je ponovni uporabnik iz druge knjižnice, sta pogodbeni strani obe knjižnici.

Nekoliko zahtevnejši problem nastane pri opredelitvi določbe, ki se nanaša na sodišče in izbiro prava v primeru spora. Knjižnice, ki se med seboj povezujejo, so namreč lahko iz različnih držav. Pravne ureditve intelektualne lastnine pa se od države do države razlikujejo. Zato je treba že v pravilih knjižničnega medsebojnega sodelovanja definirati metodologijo določanja prava in sodišča, ki je pristojno za reševanje morebitnih sporov med knjižnicami. Predlagamo, da bi veljala zakonodaja tiste države, v kateri je strežnik, ki ponuja izhodiščno spletno stran za dostop do posameznih knjižnic tržišča oziroma kataloge komponent, ki jih tržišče ponuja (država ponudnika te storitve ali knjižnice - člana tržišča, ki to stran vzdržuje). Ker je lahko na skupnem tržišču navzočih veliko knjižnic iz različnih držav, lahko ponovne uporabnike različni sistemi zmedejo ali celo odvračajo. Zato je poenotenje sistemov tudi v tem primeru najoptimalnejša rešitev za preprečevanje ali vsaj zmanjšanje števila kršitev pravic na komponentah.

6.4.2 Pogodba o sodelovanju in medsebojnih obveznostih

Stična točka medsebojno povezanih knjižnic je izhodiščna spletna stran, ki igra vlogo portala. Za skupno izhodiščno stran in z njo povezane aktivnosti skrbi bodisi neodvisni ponudnik teh storitev bodisi ena od knjižnic. Pred oblikovanjem spletnega komponentnega tržišča je treba postaviti pravila medsebojnega sodelovanja ter definirati medsebojne obveznosti. V bistvu gre za to, da mora vsaka knjižnica, ki se vključuje v komponentno tržišče, podpisati neke vrste pogodbo o včlanitvi ali pogodbo o sodelovanju in medsebojnih obveznostih, katere sestavni del so pravila komponentnega tržišča, podobno kakor so pravila knjižnice sestavni del pogodbe o včlanitvi v okolju samostojne knjižnice.

Razlike med pogodbo o včlanitvi v samostojno knjižnico (ogrodje C) in včlanitvi v komponentno tržišče:

- V vseh določbah, kjer je omenjeno spoštovanje pravil, se upoštevajo pravila komponentnega tržišča.
- V določbi 1 in vseh drugih določbah se član zamenja s knjižnico, knjižnica pa s komponentnim tržiščem. Pooblaščeni uporabniki so člani knjižnice, ki vstopa v tržišče.
- V določbi 3 se definira sodišče in izbira prava, ki se uporablja v primeru spora med knjižnico in tržiščem. Predlagamo, da je za to merodajno pravo države, v katero sodi ponudnik skupne spletne izhodiščne strani in ki vzdržuje seznam vseh knjižnic ter katalog ponujenih komponent.
- V določbi 4 se nivo dostopa opredeli za posamezno knjižnico.
- V določbi 5 se opišejo izdelki in storitve, ki jih spletno komponentno tržišče ponuja ponovnim uporabnikom.

Pri povezovanju knjižnic med seboj je treba zagotoviti enolično označevanje posameznih knjižnic tržišča. Oznaka te knjižnice se v primeru licenciranja komponent razvijalcem, ki niso neposredni člani neke knjižnice, doda kot predpona že obstoječim enoličnim oznakam komponent (IŠK). Na ta način je omogočeno tudi sledenje posameznim komponentam posamezne knjižnice. Pakiranje ali opremljanje komponent z identifikacijsko številko knjižnice se lahko izvede znotraj postopka kodiranja ali šifriranja komponente tik pred razpečevanjem oziroma pošiljanjem ponovnemu uporabniku.

Ker je komponente možno posredovati samo v eno knjižnico spletnega komponentnega tržišča, so postopki enaki kakor v okolju samostojne knjižnice. Zato se lahko za licence uporabi ogrodje B.

7 ZAKLJUČEK

Ob pregledu literature, bodisi monografij, znanstvenih člankov, doktorskih disertacij in druge s področja razvoja programskih sistemov nasploh, še toliko bolj očitno pa poslovnih informacijskih sistemov ugotovimo, da si avtorji kot glavni cilj postavljajo najti različne rešitve problema krize programske opreme, ki nastaja kot posledica zahtev in potreb po hitrem spreminjanju, prilagajanju in razvoju programske opreme za čim manjše stroške. Navadno se rešitve iščejo ali oblikujejo v različnih novih pristopih, metodah in modelih ali razširitvah in optimizacijah obstoječih. Pričujoča disertacija tako po cilju kakor po rezultatih tudi sodi v omenjeno skupino del.

Modeli, ogrodja, procesi in postopki, ki jih v tem delu predstavljam, temeljijo na ponovni uporabi kot modernem perspektivnem procesu, ki omogoča povečanje produktivnosti, kakovosti, zmogljivosti in funkcionalnosti iz ponovno uporabnih komponent razvite programske opreme, znižanje razvojnih stroškov ter povečanje učinkovitosti razvoja in bodočega vzdrževanja tako razvitih programskih sistemov. Naštete pridobitve izhajajo iz dejstva, da so ponovno uporabne komponente načrtovane za večkratno uporabo v različnih okoljih, primerno preskušene in predvsem opremljene z ustrezno dokumentacijo, ki lajša ponovno uporabo. Sistematična ponovna uporaba zahteva natančno načrtovan in organiziran pristop k razvoju, pred njeno vzpostavitvijo pa je treba rešiti kopico različnih problemov: tehničnih, tehnoloških, upravljavskih, ekonomskih in tudi pravnih.

V dosedanjih raziskavah so bile v dovolj veliki meri obravnavane vse vrste problemov, zaostajala pa je obravnava pravnih problemov, ki se najpogosteje izražajo v obliki kršenja pravic intelektualne lastnine. Po mojem mnenju predstavljajo nerešeni pravni problemi in dileme, ki izhajajo iz nezadostne in neustrezne pravne zaščite programske opreme ter neustrezno obravnavanega koncepta ponovne uporabe v obstoječih pravnih ureditvah s področja varstva pravic intelektualne lastnine na programski opremi, bistven element za

izogibanje ponovni uporabi "tujih komponent". Ker pa tega problema oziroma te družine problemov ni mogoče reševati samo z ustrezno spremembo predmetne zakonodaje, sem v pričujočem delu ustrezno nadgradila celoten proces ponovne uporabe z različnimi mehanizmi, ki bi v čim večji meri preprečevali nastajanje tovrstnih kršitev in tako spodbujali razvijalce bodisi k poseganju po ponovno uporabnih komponentah bodisi k razvoju ponovno uporabnih komponent.

V ta namen v delu:

- Definiram osnovne koncepte in strategije ponovne uporabe, identificiram udeležence v procesu ponovne uporabe, njihove vloge in medsebojne odvisnosti ter povezave, prav tako pa tudi probleme, s katerimi se soočajo ter predpogoje za uspešno ponovno uporabo.
- Izdelam primerjavo med strategijami ponovne uporabe ter podam odločitveni model za izbiro najustreznejše strategije glede na stopnjo zrelosti ponovne uporabe v organizaciji ter spremljajoče stroške.
- Da bi sploh lahko celostno obravnavali pravne probleme pri ponovni uporabi, najprej podam sistematičen pregled oblik pravnega varstva programske opreme in ugotovimo, da se to varstvo vse preveč omejuje na ožji pojem računalniškega programa ter da so osnovni koncepti, ki izhajajo iz določenih (najpogostejših) oblik varstva, kot so omejitev reprodukcije in izpeljava del, milo rečeno, skregani s konceptom ponovne uporabe.
- Podam temeljit pregled vrst ali družin problemov pravne narave, povezanih s ponovno uporabo komponent ter usmeritve, ki naj bi pravnim strokovnjakom pomagale pri oblikovanju sprememb obstoječe pravne ureditve pravic intelektualne lastnine na programski opremi (upoštevaje koncepte ponovne uporabe) ali nove *sui generis* ureditve tega področja, po možnosti na mednarodnem nivoju.
- Podam model procesa ponovne uporabe, ki vključuje tako produkcijo kakor večkratno (u)porabo teh komponent, njihovo vzdrževanje (predvsem vzdrževanje ponovne uporabnosti kot najpomembnejše lastnosti) in upravljanje. Aktivnosti in opravila procesa so podana splošno, neodvisno od konkretne razvojne metodologije in tehnike, na temelju standardnega nabora razvojnih aktivnosti [145, 146]. Poseben poudarek je namenjen dodajanju pravnih aktivnosti, s katerimi lahko preprečimo ali vsaj v čim večji meri omejimo nastajanje kršitev pravic.

- Osrednji del disertacije namenim knjižnici ponovno uporabnih komponent kot osrednjemu elementu procesa ponovne uporabe in veznemu členu med produkcijo ter porabo komponent. Knjižnica naj bi združevala varne, preskušene, kakovostne ponovno uporabne komponente domene, njena struktura naj bi slonela na modelih domene, podpirala naj bi sledenje uporabe komponent in izdelave različic komponent, osveževanje spremljajoče dokumentacije, zbiranje povratnih informacij o komponentah in storitvah knjižnice, zbiranje potreb in zahtev ponovnih uporabnikov, programsko podprto sklepanje licenčnih pogodb za komponente, povezovanje v komponentna tržišča oziroma povezovanje z drugimi knjižnicami domene, varno distribucijo komponent ter vse potrebne postopke za sprejem komponent (preskušanje, ugotavljanje ponovne uporabnosti ter ustreznosti, certificiranje, označevanje lastništva in pravic na komponenti idr.). Postopki in opravila so definirani tako, da se je možno v čim večji meri izogniti kršitvam pravic na komponentah.
- Podrobno definiram vsebino ponovno uporabne komponente oziroma njene spremljajoče dokumentacije, ki je ključna za hitro in učinkovito ponovno uporabo komponente.
- Vsi medsebojni odnosi med posameznimi udeleženci v procesu ponovne uporabe ter opredelitev pravic na posameznih komponentah so uravnani s pogodbami, zato podam pregled najpomembnejših vrst pogodb za ponovno uporabo ter predlagam njihove vsebine. Vsebine podam v obliki standardiziranih ponovno uporabnih ogrodij. Pogodbe, posebno licence, bi se shranjevale v posebni podatkovni bazi, kjer bi bile na voljo posameznim razvijalcem, ki bi svoje komponente povezali z ustreznimi vzorci licenc ter tako pripravili podlago za nadaljnje sklepanje pogodb s posameznimi ponovnimi uporabniki, ki ga v njihovem imenu izvaja knjižnica. Pri oblikovanju vzorcev, ogrodij ali predlog teh pogodb stremim k čim večji učinkovitosti in standardiziranosti. Vse, kar je lahko opredeljeno v pravilih knjižnice kot osnovnemu sestavnemu delu pogodbe o včlanitvi, se v posameznih pogodbah ne ponavlja, tako da so listine krajše, bolj berljive in preglednejše. Definicije posameznih pojmov so podane globalno, v pravilih knjižnice, in so tako na nek način tudi standardizirane. Podam tudi predlog oziroma algoritem za programsko podprto sklepanje pogodb. Slednje omogoča vnaprejšnjo določitev standardnih naborov izbir pri posameznih določilih. Ponovni uporabniki lahko predloge licenčnih pogodb pregledajo že med brskanjem po knjižnici (ko pregledujejo posamezne komponente in se odločajo o izbiri). Tako se lahko v primeru za njih zelo neugodnih

licenčnih pogojev odločijo, da neke komponente ne bodo ponovno uporabili. Z vnaprej pripravljenimi ponovno uporabnimi licencami prihranijo ponovni uporabniki veliko časa, saj se izognejo navadno dolgotrajnejšemu postopku sklepanja licenčnih pogodb. Za enostavnejše komponente ali kar za vse črne komponente, pri katerih ni dovoljeno nobeno spreminjanje in prilagajanje, predlagam licenčne pogodbe "*click-wrap*".

7.1 NADALJNJE DELO

Področji, ki ju obravnavam v pričujočem delu, torej intelektualna lastnina programske opreme in ponovna uporaba, sta zelo obširni. Glede na dosedanjo (ne)obravnavanost pravnih vidikov ponovne uporabe je pričujoče delo dobro izhodišče za številne konkretne rešitve, ki bi jih bilo mogoče podati in vgraditi v razvojno okolje na temelju ponovne uporabe. Poleg tega pa predstavljajo tudi razmišljanja in ugotovitve o neustreznosti pravne ureditve pravic na programski opremi dobra izhodišča pravnim strokovnjakom pri spreminjanju obstoječe ali pisanju nove pravne ureditve (*sui generis*) za področje programske opreme in digitalno intelektualno lastnino. Možnosti za nadaljnje delo so torej številne. Med najpomembnejše štejemo:

- oblikovanje baze znanja o komponentah domene in o (ponovnih) uporabah komponent ter storitev knjižnice, ki bi predstavljala osnovo za hitrejše in lažje odločanje oziroma izbiranje komponente iz množice kandidatnih komponent;
- izdelava konkretne knjižnice (repozitorija) na temelju podanega modela;
- izdelava konkretne programske rešitve za sklepanje licenčnih in drugih pogodb preko interneta;
- izdelava podatkovne baze vzorcev licenčnih pogodb po podanih predlogah, kjer bi bile tudi licence podobno klasificirane kot komponente (odprtokodne komponente, proste komponente, črne komponente, poslovne komponente), saj je vrsta licenčne pogodbe najprej odvisna od vrste komponente in načina njene ponovne uporabe;
- preučitev možnosti in smiselnosti zaklepanja komponent, za katere je bila podeljena licenca za spreminjanje ali izdelavo različice, s čimer bi preprečili nastajanje več različic hkrati;

- standardizacija izrazoslovja, ki je za uspešno medsebojno povezovanje knjižnic nujna, vsaj znotraj domene;
- standardizacija postopkov medsebojnega povezovanja knjižnic;
- vključitev znanj o pravnem varstvu programske opreme in pravicah intelektualne lastnine iz tega naslova v izobraževalni proces računalnikarjev in informatikov (že na nivoju srednje šole);
- sodelovanje razvijalcev (strokovnjakov računalništva in informatike) pri posodabljanju obstoječega pravnega sistema s področja intelektualne lastnine na programski opremi in/ali pisanju nove, svojevrstne ureditve za to področje.

PRILOGA A:
OBLIKE VAROVANJA INTELEKTUALNE LASTNINE NA RAČUNALNIŠKIH PROGRAMIH

Med najstarejše oblike varstva intelektualne lastnine na računalniških programih sodi poslovna skrivnost, saj se je uporabljala vse od zgodnejših obdobij razvoja računalniške industrije. S pojavom in razširitvijo mikroračunalnikov ter programske opreme za množično trženje (ang. *mass-marketed software*) v srednjih in poznih 70-ih se je kot primarna ali prevladujoča oblika varovanja pravic na računalniških programih uveljavilo avtorskopravno varovanje, ki se je kot tako obdržalo vse do danes. Obe obliki varovanja intelektualne lastnine spremljata še patentnopravno varovanje (predvsem v ZDA) in pravo blagovnih znamk. Slednji danes postajata vse pomembnejši in bosta, po predvidevanjih nekaterih avtorjev [79], v naslednjem desetletju še pomembnejši. Seveda pa so za obdobje novih, hitro razvijajočih se tehnologij računalništva in informatike vedno bolj pomembne tudi druge oblike varstva intelektualnih pravic, ki so prikrojene ali se jih da prikrojiti trenutnim zahtevam in potrebam po varstvu intelektualnih pravic na računalniški programski opremi.

Vseskozi pa se pojavlja vprašanje, kateri način je najprimernejši oziroma najpopolnejši. Po Samuelsonovi [78], McManisu [82] in drugih je vzrok za porajanje tovrstnih vprašanj predvsem v številnih posebnih lastnostih računalniških programov, ki pravnim strokovnjakom pri sprejemanju odločitev povzročajo precejšnje težave, tudi in predvsem v razumevanju teh lastnosti.

Najpomembnejši del ali bistvo varstva pri programih je tisto, kar programi naredijo oziroma počno, torej njihovo delovanje, ne tekst, napisan v nekem višjem programskem jeziku, ki predstavlja izvorno kodo tega programa. Tekst in delovanje programa sta neodvisna, saj se kaj hitro lahko zgodi, da nekdo, ki originalne izvorne kode nekega programa sploh nikoli ni videl, na preprost način napiše funkcionalno enak oziroma podoben program. Programi so nenazadnje stroji – entitete, ki proizvajajo neke rezultate, rezultat programa pa je ravno njegovo delovanje. Kot sredstvo za dosego tega delovanja se uporablja tekst (izvorna in objektna ali strojna koda). Od tu tudi osnovna dilema, ali naj se računalniški programi ščitijo s patenti kot "stroji" ali naj se varujejo kot pisana dela po avtorskem pravu.
Računalniški programi delujejo torej kot del stroja (v objektni kodi), pa tudi kot sredstvo komuniciranja z ljudmi (v izvorni kodi), torej bi lahko bili varovani po eni strani kot patenti, po drugi pa avtorskopravno. Istočasno jih je mogoče javno distribuirati v objektni kodi in vzdrževati kot skrivnost v izvorni obliki, zato jih je mogoče varovati tudi z mehanizmom poslovne skrivnosti. Nekateri programi so celo vgrajeni v polprevodniške čipe,

katerih načrti so posebej varovani s svojevrstno pravno ureditvijo za polprevodniške elemente[287]. Zaslonske prikaze, ki jih ustvarijo računalniški programi, je možno varovati kot literarna ali umetniška dela po avtorskem pravu, lahko pa celo po pravu blagovne znamke, če posedujejo ustrezne lastnosti.

Računalniški programi se torej od drugih predmetov varstva po pravu intelektualne lastnine razlikujejo, saj so v bistvu neke vrste križanci med strojnim, pisnim in umetniškim delom.

A.1 Avtorskopravno varovanje

Avtorske pravice oziroma avtorsko pravo je še vedno ključno in najpomembnejše pravno sredstvo za obvladovanje pretoka idej, znanja in produktov znanja, med katere sodi tudi programska oprema. Zato se večina vprašanj v zvezi z varovanjem računalniških programov rešuje v okviru avtorskega prava[288]. Po Zakonu o avtorski in sorodnih pravicah (v nadaljevanju ZASP)[289] [133] sodijo med avtorska dela, definirana kot "individualne stvaritve s področja književnosti, znanosti in umetnosti" tudi računalniški programi in pripravljalno gradivo zanje (načrtovalski material)[290]. Računalniški programi so varovani z avtorsko pravico kot pisana (književna, literarna) dela[291]. Pri tem je varovan izraz programa (izrazna oblika, ki je zaznavna), ne pa ideje in načela[292], na podlagi katerih je bil računalniški program izdelan. Pogoj za avtorskopravno varovanje računalniškega programa je, da gre za lastno intelektualno stvaritev njegovega avtorja – izvirno delo (biti mora kreativna invencija ali kreacija), temu pa večina računalniških programov ustreza. Vendar je originalnost, kot eden od pogojev za avtorskopravno varovanje računalniških programov, kontroverzen pogoj (npr. postavi se vprašanje originalnosti, ko gre za dela, ki jih ustvari računalnik[293]). Standardi oziroma merila originalnosti so znotraj EU veliko višji, kakor jih določa zakon o avtorskem pravu ZDA (ang. *Copyright Act* iz leta 1976 oziroma njegova sprememba iz leta 1998, *Digital Millenium Copyright Act*) [78]. Ne zadostuje namreč, da je delo originalna stvaritev njegovega avtorja, da avtor torej samega dela ali idej, ki so njegova osnova, ni kopiral od

[287] Zakon o varstvu topografije polprevodniških vezij (Uradni list RS, št. 21/95) [140].
[288] V Evropi se je s sprejetjem Direktive o pravni zaščiti računalniških programov (ang. *Council Directive on the legal protection of computer programs*) z dne 14.5.1991 na področju varstva računalniških programov izoblikovalo usklajeno evropsko pravo, saj je večina evropskih držav v svoji zakonodaji privzela določbe te direktive. Pogoj harmonizacije pravnih ureditev pa so morale izpolniti tudi nove članice, ki so vključno s Slovenijo pristopile v EU s 1. majem 2004. V večini držav EU so zakonom, ki urejajo avtorske pravice, dodane še določbe, ki urejajo sorodne pravice. Gre za pravice, ki so povezane z avtorskopravno varovanimi deli oziroma avtorskimi pravicami.
[289] Slovenski ZASP vsebuje poseben oddelek določb (2. oddelek 4. poglavja, čl. 111-117), ki urejajo (po vzoru Direktive EU o varovanju računalniških programov) področje računalniških programov ter poseben oddelek določb (6. oddelek 5. poglavja, čl. 141a – 141g), ki urejajo pravice izdelovalcev podatkovnih baz (po vzoru Direktive EU o varstvu podatkovnih baz). Tako pozna slovenski ZASP samo manjša odstopanja od besedil predmetnih Direktiv. Spremenjen in dopolnjen ZASP [133] pa je usklajen tudi z zelo pomembno Direktivo ES o informacijski družbi, ki rešuje vprašanja digitalnega okolja in hkrati ureja pravila novih pogodb WIPO v ES. S to direktivo se usklajujejo pravice do reproduciranja in distribuiranja, priobčitev javnosti, pravno varstvo naprav, ki preprečujejo kopiranje in sistem upravljanja pravic.
[290] 111. člen ZASP [133].
[291] 2. člen ZASP [133].
[292] 9. člen ZASP [133].
[293] O problemu avtorstva v teh primerih lahko več preberemo v [78].

drugega, ampak mora biti delo lastna intelektualna stvaritev njegovega avtorja. Kot pogoj za varovanje znotraj EU pa ni treba, da je delo zapisano (ali shranjeno) na nekem fiksnem, opredmetenem mediju, izraženo na človeku razumljiv način. Dejstvo pa je, da je pri vseh računalniških programih individualnost mogoče samo domnevati[294], kar lahko predstavlja problem. Ti problemi ustvarjajo dileme o upravičenosti avtorskopravnega varovanja računalniških programov.

A.1.1 Predmet varstva po avtorskem pravu

Po avtorskem pravu sta pri računalniški programski opremi varovani tako izvorna kakor objektna ali strojna ali izvršljiva koda, varovani so literarni elementi v programski kodi, pa tudi neliterarni elementi (zaslonski prikazi), pod varstvo sodijo tudi funkcionalni elementi in protokoli ter nenazadnje rezultati programa (uporabniški vmesnik). Varstvo torej uživajo:

- Sestavni deli in naslov programa: sestavni deli avtorskega dela so ravno tako avtorskopravno varovani, če so sami zase individualne stvaritve, torej avtorska dela. Naslov programa pa je varovan na dva načina: po avtorskem pravu, če je naslov sam po sebi individualna intelektualna stvaritev[295] ali po konkurenčnem pravu, če gre za delo enake vrste in bi ponovna uporaba enakega naslova ustvarila zmedo glede izvora dela[296]. Pri računalniških programih navadno ne govorimo o naslovu, ampak imenu programa. V praksi se imena računalniških programov največkrat varujejo z blagovno znamko [136][297].
- Pripravljalno gradivo (ang. *preparatory design material*) kot osnutek programa oziroma drugo gradivo iz posameznih faz nastajanja računalniškega programa (diagrami, programske specifikacije ipd.): zadosten pogoj za avtorskopravno varovanje tega gradiva je, da je neposredno namenjeno (prihodnjemu) konkretnemu programu[298].
- Algoritmi so bistvo računalniškega programa in so praviloma avtorskopravno varovani, če ne gre za splošne, standardne algoritme, ampak implementacijske (konkreten opis postopkov in navodil v programu). O vrsti algoritma odločata stopnja neabstraktnosti in nesplošnosti algoritma. [103]
- Vmesniki (ang. *interface*) so "deli programa, ki omogočajo logično in po potrebi fizično medsebojno povezavo in interakcijo med posameznimi deli programske opreme in/ ali strojne opreme, tako da ti delujejo, torej omogočajo interoperabilnost med različnimi deli programske in strojne opreme (tudi različnih proizvajalcev" [103] Varovani so enako kakor programi, pod določenimi pogoji pa je mogoč vpogled v informacije, ki jih vsebujejo, bodisi z opazovanjem[299] bodisi z dekompilacijo[300].
- Uporabniški vmesniki (ang. *user interface*): določajo način upravljanja s programom in predstavitve ter posredovanja podatkov na zaslonu. Primarno se varujejo s pravom konkurence, avtorskopravno pa samo prek samega programa (kot njegov sestavni del).

[294] Komentar ZASP [104], str. 268.
[295] 1. odstavek 6. člena ZASP [133].
[296] 2. odstavek 6. člena ZASP [133].
[297] ZIL – Zakon o industrijski lastnini [136], čl. 17 – 24.
[298] 111. in 116. člen ZASP [133].
[299] 3. odstavek 114. člena ZASP [133].
[300] 115. člen ZASP [133].

A.1.2 Izključne avtorske pravice

Avtorske pravice vedno nastanejo z nastankom avtorskega dela, torej računalniškega programa. Avtorska pravica je monopol avtorja nad izkoriščanjem njegovega avtorskega dela in nad določenimi oblikami razpolaganja s primerki avtorskega dela [83, 91]. Avtorju zagotavlja na eni strani premoženjske koristi od izkoriščanja njegovega dela (materialne pravice), po drugi strani pa spoštovanje njegovih moralnih interesov pri izkoriščanju njegovega dela (moralne pravice), torej ima avtor izključne pravice delo izkoriščati tako v materialni kakor v nematerialni obliki. Pravica do materialnega izkoriščanja vključuje pravico reprodukcije, distribucije, predelave in priredbe, izkoriščanje nematerialne pravice pa pravico do priznanja in navedbe avtorstva, spoštovanja dela, prve objave in skesanja[301]. Imetnik pravic se lahko odloči tudi, da iz različnih razlogov (reklamnih, izobraževalnih ali drugih) svojih pravic ne bo v celoti uveljavljal, kar pa mora biti jasno objavljeno. V tem primeru gre torej za prenos materialnih avtorskih pravic na tretjo osebo. Obseg in omejitve pri prenosu se določajo v pogodbi. Prenos pravic je lahko ekskluziven ali neekskluziven, kar pomeni, da gre lahko za pravi prenos posamezne materialne pravice na tretjo osebo ali pa le za podelitev licence.

Glede obsega odpovedi pravic imajo poseben status računalniški programi v prosti uporabi (ang. *freeware*), programi iz javne domene (ang. *public domain*) in programi v preizkusni uporabi (ang. *shareware*). Gre za varovane računalniške programe, pri katerih se imetnik avtorskih pravic odloči, da teh pravic na njih ne bo v celoti uveljavljal. Komercialna programska oprema (ang. *payware*) se licencira in plačuje. Pri brezplačnih programih imetnik avtorskih pravic prepusti program v brezplačno uporabo, ki obsega reproduciranje, predelavo in nadaljnje distribuiranje prvotnega ali predelanega programa. Avtor vedno obdrži moralne pravice, kolikšen del materialnih pravic obdrži, pa je odvisno od primera do primera. Pri programih na preizkušnjo imetnik za določen čas omogoči prosto preizkusno uporabo (reprodukcija in distribucija), po preteku tega časa pa mora uporabnik program kupiti (minimalna licenčnina), če ga želi naprej uporabljati. Gre torej le za posebno obliko distribucije.

V javno domeno sodijo dela, ki po pravu intelektualne lastnine niso varovana (so povsem nevarovana). Programsko opremo iz javne domene je mogoče prosto uporabljati, spreminjati, razmnoževati, si jo deliti z drugimi ali komercialno izkoriščati kopije ali različice dela, pridobljene na legitimen način. Delo je torej na razpolago za splošno uporabo brez omejitev intelektualne lastnine (na programih ni nobenih pravic, ne moralnih ne materialnih). Neko delo lahko preide v javno domeno na tri načine: bodisi ne doseže pogojev za varstvo po pravu intelektualne lastnine, bodisi se upravičenec odpove pravicam do intelektualne lastnine ali jih ne uveljavi ali mu jih v določenem roku ni uspelo uveljaviti po ustaljenih postopkih, tretji razlog pa je lahko pretek varstva intelektualnih pravic. Če delo iz javne domene (ponovno) uporabimo, nad njim ne moremo zahtevati varstva pravic intelektualne lastnine – samo morebitna izboljšava ali nadgradnja tega dela lahko zapade pod varstvo intelektualne lastnine.

[301] Po Oman [83] je edino merilo za ugotavljanje, kaj je izkoriščanje dela v avtorskopravnem pogledu, oziroma kdaj je mogoče uveljavljati materialne avtorske pravice, reproduciranje avtorskega dela kot nematerialne dobrine. Trdi namreč, da se pri razlaganju izkoriščanja avtorskega dela kot nematerialne dobrine delajo napake, če se za merilo takega izkoriščanja upoštevajo samo ekonomske koristi.

Posebna kategorija programske opreme ali programov so t.i. odprtokodni programi (ang. *open-source*). Avtorske pravice na tej programski opremi obstajajo in se tako kot pri plačljivi prenašajo na uporabnika z licenco.

V osnovi lahko avtorske pravice uveljavlja samo avtor računalniškega programa ali drugače povedano, avtorske pravice nastanejo vedno pri avtorju računalniškega programa. Ko računalniški program nastane v času zaposlenosti (v delovnem razmerju), pa pravice pripadajo delodajalcu[302] – delodajalec namreč pridobi izključno in neomejeno vse materialne avtorske pravice do računalniškega programa, moralne pravice pa ostanejo avtorju. Enako je tudi pri naročilu računalniškega programa, kjer naročnik pridobi vse avtorske materialne pravice. Na splošno torej velja, da nastanejo in ostanejo avtorske pravice vedno pri avtorju (razvijalcu računalniškega programa), razen ko je dodelitev pravic drugače pisno določena s pogodbo ali pogodbenim dogovorom, s katerim se pravice po 112. členu ZASP izključijo delno ali v celoti.

Lastništvo računalniškega programa in lastništvo pravic na računalniškem programu je možno prodati, licencirati ali ga podariti. Imetnik lastninske pravice lahko tudi uveljavlja licenco ali eno oziroma več izključnih pravic avtorskopravnega varstva.

Biti imetnik (lastnik) avtorskih pravic (ang. *copyright holder* ali *copyright owner*) na računalniških programih pomeni imeti možnost uveljavljanja izključnih pravic, ki izhajajo iz avtorskopravnega varstva[303]. Dejanja, ki jih avtor računalniškega programa vsem drugim lahko prepove oziroma so zanje potrebna posebna dovoljenja:

- reproduciranje[304] (izdelava kopij) varovanega dela – v celoti ali sestavnih delov,
- izdelava izpeljanih (derivativnih) del iz varovanega dela oziroma predelava varovanega dela (prevod, prilagoditev, priredba ali drugačna predelava računalniškega programa ter reproduciranje rezultatov teh predelav[305]) ,
- distribuiranje varovanega dela (prodaja ali drugačen prenos lastništva, na primer najem, lizing, oddaja) ter
- druge pravice[306].

Avtor lahko navedene izključne pravice delno ali v celoti prenese na tretjo osebo z licenčno pogodbo. Licenčne in druge pogodbe so podrobneje obravnavane v razdelku A.6 te priloge.

[302] 112. člen ZASP, ki ureja Delovno razmerje in avtorsko pogodbo [133].
[303] 113. člen ZASP [133].
[304] Nalaganje, prikazovanje, izvajanje in prenašanje računalniškega programa v zakonu niso izrecno opredeljeni kot oblike reproduciranja, vendar je za tovrstna dejanje potrebno dovoljenje avtorja. Na splošno pa velja, da shranjevanje računalniškega programa v pomnilnik (četudi začasni) že predstavlja reproduciranje.
[305] Avtor prvotnega programa naj bi obdržal pravice tudi v novem, predelanem ali izpeljanem delu. Po avtorjih Komentarja k ZASP sodijo med predelave računalniškega programa prevodi iz enega programskega jezika v drugega, spremembe izvorne v strojno kodo in obratno, dodatki k programu, opustitve določenih delov programa ali prerazporeditve prvotnega programa [104], str. 272.
[306] Med izključne pravice na avtorskih delih sodita tudi javno predstavljanje del ter javno prikazovanje del (literarno, glasbeno, dramsko, koreografsko, pantomima, grafično, kiparsko, filmsko ali drugo avdiovizualno delo), vendar v primeru računalniških programov te pravice niso zanimive.

A.1.3 Omejitve izključnih pravic

Trajanje avtorskih pravic: Avtorska pravica traja še 70 let po avtorjevi smrti, pri skupinskih delih pa še 70 let po smrti zadnjega soavtorja, kar je pri računalniških programih izjemno dolgo obdobje.

Upravičena uporaba oziroma dovoljeni način uporabe[307] *(*ang. *fair use)*: Iz avtorskega prava ZDA poznana doktrina "*fair use*" , pri nas pa dovoljena ali upravičena uporaba lastniku licence, torej upravičenemu uporabniku, omogoča uporabo avtorskopravno varovanega materiala na primeren način brez soglasja imetnika pravic na tem materialu. Lastniku kopije varovanega dela oziroma zakonitemu pridobitelju (upravičenemu uporabniku) računalniškega programa daje ZASP pravico kopiranja programa (izdelava največ dveh varnostnih primerkov), če je to potrebno za njegovo izvajanje oziroma njegovo uporabo, pri čemer dovoljenje avtorja ni potrebno. Upravičeni uporabnik lahko zaradi uporabe računalniškega programa, ravno tako brez avtorjevega dovoljenja, odpravi napake, proučuje in preskuša program, če to stori pri nalaganju, prikazovanju, izvajanju, prenašanju ali shranjevanju programa. Omogočena so mu tudi reproduciranje, prilagajanje in predelava programa z namenom, da se s tem poveča njegova uporabnost oziroma koristnost, kar pa je zelo kontroverzno, saj so ta dejanja s 113. členom brez avtorjevega dovoljenja prepovedana. Splošno sodijo med dovoljene načine uporabe avtorskopravno varovanega dela nekomercialne oblike uporabe, kot so uporaba v izobraževalne in raziskovalne namene, komentiranje, novinarsko poročanje, pisanje kritik ali parodij. Pri ocenjevanju, ali gre za dovoljeno oziroma upravičeno uporabo ali ne, se poleg samega namena uporabe upoštevajo še narava varovanega dela, količina in pomembnost uporabljenega dela glede na celotno delo ter učinek te uporabe na tržišče. [104]

Pravice prve prodaje: lastniku kopije je dovoljena prodaja te kopije brez posebnega dovoljenja imetnika avtorskih pravic, ni mu pa dovoljena izdelava nadaljnjih kopij.

Dekompiliranje ali razgradnja[308]: Dekompiliranje je postopek, pri katerem poskušamo s posebnim prevajalnikom na podlagi prevedene kode v strojni jezik vnovič sestaviti prvotni program v višjem programskem jeziku. Če izvedemo dekompilacijo računalniškega programa ali samo dela tega programa, da bi dosegli interoperabilnost (medsebojno povezanost in interaktivnost) z drugim, neodvisno ustvarjenim programom ali računalniško strojno opremo, pri čemer to izvede pridobitelj licence, drugi upravičen uporabnik ali v njihovem imenu pooblaščena oseba, kateri potrebne informacije niso bile predhodno dostopne, lahko to storimo brez avtorjevega dovoljenja. Vendar z dekompilacijo pridobljene informacije ne smejo biti uporabljene za drug namen, razen za doseganje interoperabilnosti, ne smejo se odstopiti tretjim osebam ali uporabiti za razvijanje, proizvodnjo ali trženje drugega računalniškega programa, ki je zelo podoben dekompiliranemu programu. Kot tako je dekompiliranje vrsta obratnega (povratnega) inženiringa. [104] Glede na sodno prakso [78, 79] je možno zatrditi, da obratno inženirstvo ali dekompilacija ne predstavljata kršitve avtorskih pravic, če je to edini način, ki pelje do odkritja idej in funkcionalnih elementov, utelešenih v računalniški program (informacije o vmesnikih, ki

[307] 114. člen ZASP [133].
[308] 115. člen ZASP [133].

omogočajo interoperabilnost med različnimi programi in strojno opremo) in kjer obstaja legitimen razlog za tako početje. V teh primerih se dekompilacija obravnava kot dovoljen način uporabe (ang. *fair use*).

A.1.4 Obratno (povratno) inženirstvo

Pri obratnem ali povratnem inženirstvu[309] poskuša uporabnik ustvariti ekvivalent originala izvorne kode in objektne kode programa oziroma poskuša priti do razvojnih podrobnosti. Gre za proces, ko se program oziroma programski izdelek obdela v nasprotni smeri njegovega razvoja (nasproten proces programskemu inženirstvu), kar pomeni, da se najprej z dekompilacijo pridela človeku razumljivo izvorno kodo[310], iz katere je mogoče preučiti strukturo programa in njegove tehnične parametre. Bistvo povratnega inženirstva je v tem, da se z analizo pridobljene objektne kode preučita načrt in zgradba nekega programa. Razlogi za izvajanje obratnega inženirstva pa so lahko različni:

- učenje neke nove tehnike programiranja (iz akademskih razlogov ali za ustvarjanje tržnega izdelka),
- povečanje učinkovitosti uporabe programa (odkrivanje napak, prilagajanje programa posebnim zahtevam uporabnika ali spreminjanje programa, da bo združljiv z drugim programom ali delom strojne opreme),
- razvoj specifičnega, tržnega, vendar nekonkurenčnega programa, ki naj bi bil združljiv z analiziranim originalnim programom ali
- razvoj funkcionalno enakovrednega izdelka, ki pa naj bi bil konkurenčen originalnemu programu.

Tako po EU Direktivi [109] kot po ZASP (118. člen) [133] predstavlja obratno inženirstvo javno distribuiranih računalniških programov, vključno z njihovo dekompilacijo (razgradnjo) in ponovnim zbiranjem (disasembliranjem)[311], dovoljeni način uporabe, ko je to potrebno za dosego interoperabilnosti z drugim programom. Izrecno dopustna oblika povratnega inženirstva po ZASP je t.i. povratno inženirstvo "black-box"[312], pri katerem se lastnosti vmesnikov spoznavajo s preizkušanjem programa, torej brez prevajanja strojne kode v izvorno.

Če so računalniški programi za javno distribucijo varovani s pravom poslovne skrivnosti, je njihovo obratno inženirstvo legalno.

A.1.5 Kršenje avtorskih pravic

Veliko vprašanj se poraja v povezavi z dilemami, kdaj sploh pride do kršitve avtorskih pravic na računalniških programih in kateri vidiki programske opreme so sploh (lahko) avtorskopravno varovani.

---

[309] Ang. *reverse engineering*.
[310] Večina programov se prodaja samo v obliki objektne ali strojne kode, iz katere je težko oziroma skoraj nemogoče brez posebnih metod odkriti ideje in načela, na katerih je bil program zgrajen.
[311] Obratno zbiranje – postopek, ki ga izvede program, ki prevedeni program iz strojnega jezika prevede nazaj v zbirni jezik. Programerjem je v pomoč pri razumevanju delovanja programov, katerih izvirno besedilo ni na voljo. Pogosto se uporablja tudi za iskanje napak. [147]
[312] Komentar ZASP [104], str. 276.

Do kršitve avtorskih pravic pride vedno, ko nek upravičen uporabnik izvede dejanja, za katera bi potreboval dovoljenje avtorja oziroma imetnika pravic, ne da bi to dovoljenje predhodno pridobil, in pa v primeru, ko nek uporabnik na kakršenkoli način uporablja avtorskopravno varovani računalniški program, do katerega ni prišel po legalni poti (neupravičeni uporabnik). Kot kršitev se šteje tudi distribucija nedovoljenih primerkov računalniškega programa ali posest nedovoljenega primerka računalniškega programa za gospodarske namene[313].

A.2 Patentnopravno varovanje

Večina evropskih držav je članica Evropske patentne konvencije (ang. *European Patent Convention*), ki se ukvarja z obravnavo varovanja izumov s patenti. Patent je pravica industrijske lastnine, s katerim se varuje izum. Izumitelju ga podeli država in mu s tem omogoči, da za omejeno časovno obdobje, navadno 20 let, drugim prepove gospodarsko uporabo izuma. Izum[314] kot tak pa pomeni rešitev določenega problema na področju tehnologije. S patenti se lahko varujejo tehnične inovacije oziroma izumi, ki morajo biti novi (glede na "znano stanje tehnike" [86]), inventivni, industrijsko uporabljivi in "doseženi z ustvarjalnim delom na ravni izumiteljstva" [86]. Izumi so lahko izdelki, stroji, procesi ali metode, nanašajo se torej lahko na izdelek ali postopek. S patentom ni mogoče neposredno varovati odkritja, znanstvene teorije, matematične metode, računalniškega programa in drugih pravil, načrtov, metod in postopkov za duhovno aktivnost[315]. Intelektualne zamisli (ideje) se lahko patentirajo samo, če so opredmetene v tehnični rabi. Patenti so torej bolj primerni za opredmetene izdelke, manj pa za informacijske dobrine, ki so neopredmetene, digitalne zapise vseh vrst, tudi računalniške programe.

Če bi želeli računalniške programe uvrstiti v eno od kategorij, ki lahko postanejo izum, potem bi jih bilo umestno uvrstiti med procese, saj predstavljajo skupek navodil, ki jih računalnik razume in na njihovi podlagi izvede določene operacije. Če naj bi bili deležni varstva s patentom, potem morajo biti podlaga za izdelavo neke konkretne tehnične "stvaritve", dokazati je torej treba, da je računalniški program inovativen proces[316]. Najbolj upravičeni do varstva s patentom so zato računalniški programi, ki izvajajo transformacije. Tudi nematematični računalniški algoritmi so lahko predmet varstva s patentom.

Imetnik patenta ima izključne pravice, da izdela, uporabi in proda patentiran izum, nima pa izključne pravice do izdelave izpeljanega (derivativnega) izuma ali spreminjanja patentiranega predmeta ali procesa. Ob prijavi izuma za patent je namreč treba predložiti natančen in popoln opis izuma, saj lahko vsakdo brez izjeme uporablja

[313] 116. člen ZASP [133] ne razlikuje med tem, ali z gotovostjo vemo, da gre za nedovoljeni primerek, ali samo domnevamo, pod gospodarskim namenom pa je mišljena gospodarska dejavnost, ki se opravlja na trgu zaradi dobička.
[314] Definicije izuma ne najdemo v zadevnih mednarodnih pogodbah, pa tudi v nacionalnih zakonih ne. Običajno se ne določi, kaj lahko varujemo s patentom, torej kaj je izum, pač pa se opredeli, kaj ne more biti predmet patentnega varstva. [86]
[315] 52. člen Evropske patentne konvencije in 11. člen Zakona o industrijski lastnini (ZIL-1).
[316] V sodni praksi se je v zvezi s tem določanjem porajalo veliko dilem in je bilo sprejetih veliko kontroverznih odločitev. [78]

razkrite informacije pri razvoju novih ali drugačnih tehničnih stvaritev [86]. Patenti v bistvu samo prepovedujejo nedovoljeno gospodarsko uporabo izumov neposredno na izdelkih. Tretje osebe torej ne morejo uporabljati izuma na izdelkih ali v tehnoloških postopkih za proizvodnjo in prodajo brez dovoljenja lastnika patenta.

Slabost patentov je v njihovi visoki ceni in dolgemu času čakanja nanje. Po drugi strani pa so zelo močna oblika varstva, saj varujejo izumiteljeve interese na načine, ki jih avtorsko pravo ne pozna (patent natančno opredeljuje, kaj je last imetnika pravic intelektualne lastnine). Varovanje avtorskih pravic in varovanje s patentom sta možna sočasno – patent za funkcionalnost ali algoritem, torej idejo, avtorske pravice pa ščitijo izraz te ideje.

A.3 Pravo poslovne skrivnosti

Razvijalci poskušajo pogosto varovati programsko opremo in spremljajočo dokumentacijo kot poslovno skrivnost[317]. Kot poslovno skrivnost lahko varujemo "know-how", ki ga vsebuje programska oprema – varovan je lahko programski izdelek v celoti, ideje, algoritmi, tehnike, programska orodja, programske komponente, informacije, kompilacije, metodologije, matematični modeli kot osnova za izvedbo programske opreme, načrti, specifikacije, uporabljeno znanje idr. Skratka, kot poslovno skrivnost je možno varovati vse, kar je ali bo v bodoče vir kompetitivne prednosti v poslovanju. Programska oprema, ki je varovana kot poslovna skrivnost, se ne sme pojavljati v javnosti. Če želimo to programsko opremo prenesti na druge, morajo podpisati dogovor o varovanju poslovne skrivnosti (ang. *nondisclosure agreement*), preden pridejo v stik s programsko opremo.

Poslovna skrivnost je vedno varovala podjetja pred vdorom oziroma krajo njihove skrivnosti, ni pa opredeljeno, katere izključne pravice pridobi lastnik poslovne skrivnosti. Pravzaprav mora lastnik poslovne skrivnosti za svoje pravice skrbeti sam. Problem nastane na primer v primeru, ko nekdo pride do poslovne skrivnosti prek obratnega inženirstva objektne kode, ki je zakonsko dovoljeno tudi nad avtorskopravno varovanimi računalniškimi programi[318]. Poslovna skrivnost lahko teoretično sovpada z varstvom avtorskih pravic, ko je ta skrivnost "utelešena" v pisnem delu, ne more pa sovpadati s patentom.

A.4 Blagovna ali storitvena znamka

Ime, simbol, logotip ali slogan, ki se uporabljajo za identifikacijo računalniškega programa oziroma programskega izdelka – njegovega izvora – je možno varovati kot blagovno ali storitveno znamko[319], odvisno od tega, ali obravnavamo programsko opremo kot neko dobrino, blago ali kot storitev (programska oprema pri tem

[317] Zakon o gospodarskih družbah, 5. poglavje (39. in 40. člen). [135]
[318] Za namene učenja, odpravo nepravilnosti v delovanju programske opreme, za izdelavo različice programske opreme, ki bo delovala na drugem sistemu, ko želimo doseči interoperativnost z drugo programsko opremo [133].
[319] Blagovna ali storitvena znamka je namenjena razlikovanju blaga oziroma storitev enega gospodarskega subjekta od podobnega ali istovrstnega blaga oziroma podobnih ali istovrstnih storitev drugega gospodarskega subjekta v gospodarskem prometu [136].

ni varovana). Blagovna ali storitvena znamka je kot pravica intelektualne lastnine opredeljena z Zakonom o industrijski lastnini. Drugim prepoveduje uporabo imena (znamke) računalniškega programa, ne pa izdelave enakega izdelka ali storitve. Lahko prepreči distribucijo računalniškega programa, če pomnilniški nosilec, ovitek ali priročnik nosi zaščitno znamko, ne more pa preprečiti kopiranja programa ali prodaje neavtoriziranih kopij. Kot taka torej ne zadošča za popolno varstvo računalniških programov.

A.5 Varovanje pred nelojalno konkurenco

Pri varovanju pred nelojalno konkurenco oziroma zatiranju nelojalne konkurence gre za omejevanje uporabe nečesa, kar je bilo razvito kot inovacija, novost. Konkurenčno pravo[320] se lahko uporabi samo za varstvo elementov, ki so varovani z avtorskimi pravicami ali s patentom in kjer to varstvo ni bilo doseženo. Kot taka pa ta oblika varstva ni tako učinkovita kakor avtorskopravna ali patentnopravna. Namen varstva pred nelojalno konkurenco ni varovanje nekega izdelka kot takega, ampak prepoved nedovoljenega vedenja na tržišču v povezavi z nekim varovanim izdelkom. Gre za preprečevanje distribucije izdelka nekega avtorja, ne pa za preprečevanje neavtorizirane reprodukcije tega izdelka. Kot taka se ta oblika varstva lahko uporablja le kot dodatno sredstvo varovanja računalniškega programa v kombinaciji z drugimi oblikami.

A.6 Pogodbeno pravo

Pogodbe lahko predstavljajo dodatek k eni od doslej opisanih oblik varovanja računalniških programov, lahko pa jih obravnavamo tudi kot povsem samostojno obliko varovanja pravic na računalniških programih. Po mnenju nekaterih strokovnjakov (npr. Remer) so pogodbe celo najboljši način varovanja računalniških programov, saj so "izterljivi pravni dogovori, ki natančno opredeljujejo pravice in odgovornosti vseh strani." [89] Dogovor o varovanju skrivnosti (ang. *nondisclosure agreement*) je npr. dobro sredstvo za varovanje poslovnih skrivnosti, sklene pa se lahko z vsemi, ki imajo dostop do poslovne skrivnosti (potencialne stranke, zunanji ocenjevalci, zaposleni). Taki dogovori morajo biti čimbolj konkretni. Najpomembnejši tip pogodbe ali dogovora med razvijalci programske opreme in njenimi uporabniki – strankami in pogosto hkrati tudi naročniki, pa je prav gotovo licenčni dogovor ali licenčna pogodba [321] (ang. *Licence Agreement/Contract* ali *Royalty agreement/contract*), ki zelo natančno opisuje medsebojni odnos med razvijalcem in prodajalcem ali stranko, ravno tako pa tudi program, ki je predmet pogodbe[322]. S pogodbo o prenosu avtorskih pravic je možno posamične materialne avtorske ali druge pravice avtorja vsebinsko, prostorsko ali časovno omejiti, jih izključno ali neizključno prenesti. Licenciranje je osnovna paradigma distribucije računalniške programske opreme že od samega nastanka te industrije.

[320] Zakon o varstvu konkurence. [139]
[321] Po obligacijskem zakoniku (OZ) [116], določbe o licenčni pogodbi (čl. 704 – 728).
[322] V tak dogovor lahko vključimo natančen opis dostave, plačila razvijalcu, geografske omejitve licence, spreminjanje programa, ciljno tržišče, tip računalnika, vzdrževanje, spreminjanje programa, učenje, zaščite avtorskih pravic, garancije, odškodnine, veljavnost pogodbe, arbitraže idr.

Pri distribuciji in trženju računalniških programov pridejo najpogosteje v poštev pogodba o naročilu programa, licenčna pogodba, pogodba o hrambi računalniške kode (ang. *escrow agreement*), različne oblike pogodb za dodatna zavarovanja računalniških programov (npr. pogodbena kazen) in druge, pogoj je le, da pogodbe niso v nasprotju s kogentnimi določbami ZASP (npr. člena 114/5, 115/4) [104].

A.6.1 Licenčna pogodba

Licenčna pogodba ali licenca predstavlja najprimernejši način prenosa pravic do uporabe oziroma izkoriščanja programske opreme in/ali računalniških programov.
Z licenco pridobitelj licence proti plačilu od dajalca licence pridobi le pravico izkoriščanja patentiranega izuma, tehničnega znanja in izkušenj, znamke, vzorca ali modela[323], pa tudi pravice avtorja na računalniškem programu[324]. Pridobitelj licence za računalniški program pridobi eno ali več naslednjih pravic:

- pravico reproduciranja sestavnih delov ali celotnega računalniškega programa,
- pravico prevoda, prilagoditve, priredbe ali drugačne predelave,
- pravico distribuiranja izvirnika programa in njegovih primerkov v katerikoli obliki (s podlicenco, prodajo ali najemom).

V licenčni pogodbi mora biti natančno določeno, katere pravice se prenašajo na uporabnika in v kakšnem obsegu[325]. Pridobitelj licence mora izkoriščati predmet licence na dogovorjen način, v dogovorjenem obsegu in v dogovorjenih mejah (vsebinske[326], prostorske[327], časovne[328] in količinske[329] omejitve, ki jih licenčna pogodba nalaga pridobitelju licence). Vsakršno izvrševanje pravic, ki niso v skladu z zakonom ali licenčno pogodbo, pomeni kršenje pogodbe in velikokrat tudi zakonodaje.

[323] 704. člen OZ [116].
[324] 2. odstavek 113. člena ZASP [133].
[325] Ker gre za izključne pravice avtorja se šteje, da določena pravica ni prenesena na uporabnika, če to ni izrecno določeno v licenčni pogodbi. [116]
[326] Licenca mora natančno določati pravice, ki jih dajalec licence odstopi pridobitelju licence: ali je prenos izključen ali neizključen, kako je z nadaljnjo prenosljivostjo pravic in kakšen je obseg prenosa pravic. Nekatere pravice se na uporabnika prenesejo na podlagi zakona in jih s pogodbo ni mogoče spremeniti.
[327] Pravica izkoriščanja predmeta licenčne pogodbe je navadno prostorsko neomejena, avtorske pravice pa se vedno prenašajo teritorialno. V praksi se z licenco odstopi določena pravica samo za določeno geografsko območje, določeno podjetje ali natančno določeno lokacijo [104].
[328] Trajanje licence za programsko opremo je časovno omejeno na čas zakonitega varovanja avtorske in drugih pravic, s katerimi je programska oprema varovana (70 let po avtorjevi smrti pri avtorskopravnem varovanju, 20 let v primeru patentnega varstva oziroma 10 let, če gre za patent s skrajšanim varstvom). V praksi zakonite časovne omejitve niso pomembne, saj programska oprema zastari že po nekaj letih. Lahko pa je v licenčni pogodbi dogovorjen krajši rok od časa zakonitega varstva pravic intelektualne lastnine.
[329] Uporabnik dobi licenco za določeno število dovoljenih inštalacij (namestitev). Navadno vsebujejo licence dovoljenje za reprodukcijo dveh varnostnih kopij (2. odstavek 114. člen ZASP [133]), vsako nadaljnje reproduciranje mora biti v pogodbi natančno določeno.

Pojem izključne licence pomeni, da sme predmet licence izkoriščati samo tisti, na katerega je licenca prenesena. Če v pogodbi ni navedeno, da gre za izključno licenco, se šteje, da je dana neizključna licenca[330]. Pri izključni licenci lahko upravičenec vsakogar izključi iz izkoriščanja predmeta licence, določena pravica pa se lahko prenese samo na enega uporabnika. Pri programski opremi po naročilu se navadno materialne avtorske pravice in druge pravice avtorja na programu izključno in neomejeno prenesejo na naročnika, če ni s pogodbo določeno drugače. V primeru standardne programske opreme pa se podeljuje neizključna licenca, kar pomeni, da se pravica do uporabe programske opreme ne prenese le na enega, ampak poljubno mnogo pridobiteljev. Pridobitelj licence ima v primeru izključne licence vedno možnost, da jo odstopi, medtem ko neizključna licenca ni prenosljiva brez izrecnega soglasja dajalca licence. Če je licenca neizključna, velja, da uporabnik ni upravičen do podeljevanja podlicenc, torej pravica do uporabe ni prenosljiva. V licenčni pogodbi za programsko opremo je navadno določba, ki brez dovoljenja dajalca licence prepoveduje prenos licence na tretjo osebo.

Licenčna pogodba pa ni edina, s katero lahko avtor prenaša pravice do uporabe na druge osebe, ampak to lahko stori tudi s *prodajno pogodbo*. Glavna prednost licence pred prodajno pogodbo je v tem, da se z licenčno pogodbo lahko podajo dodatni pogoji[331] za uporabo programa, ki jih pri prodajni pogodbi[332] ni. Pri prodaji posameznega primerka računalniškega programa kupec pridobi lastninsko pravico na tem programu in pomnilniškem mediju, na katerem je program shranjen, prodajalec zadrži avtorsko pravico, patent, blagovno znamko ... Na prodanem programu je izčrpana avtorjeva pravica distribuiranja. Pri prodaji pravice intelektualne lastnine kupec pridobi avtorsko pravico, patent, blagovno znamko ... in postane imetnik pravice intelektualne lastnine [66]. Problem izčrpanja avtorjeve pravice je vprašljiv tudi pri distribuiranju s prvo prodajo ali drugačno obliko pridobitve lastninske pravice na izvirnem programu, še posebno, ko distribucija poteka prek interneta, ko program ni fiksiran na materialnem nosilcu. Vendar se računalniški programi in druga avtorska dela v elektronski obliki zelo redko prodajajo, pač pa se na uporabnike prenašajo z licencami ali se na uporabnike prenaša pravica do uporabe oziroma izkoriščanja računalniškega programa.

ZASP[333] licenčno pogodbo omenja v 2. odstavku 113. člena kot pravni naslov za pravice do legalne uporabe programa, saj se v zvezi z računalniškimi programi s končnimi uporabniki sklepajo predvsem licenčne pogodbe. Tovrstne pogodbe imajo navadno obliko t.i. adhezijskih pogodb, kar pomeni, da je njihova vsebina vnaprej pripravljena in natisnjena. Za licenčne pogodbe velja pisna oblika[334]. Med seboj se licenčne pogodbe razlikujejo glede na vrsto programa (npr. sistemski, aplikacijski, standardni, posebni ipd.), glede na vrsto uporabe (npr. posamično ali v mreži), število licenciranih programov (skupinske licence, odprte licence)[335].

[330] 3. odstavek 707. člena OZ [116] in 1. odstavek 75. čl. ZASP [133].
[331] Ponudnik dodatne pogoje postavi predvsem zaradi nadzora nad distribucijo, skupnega trženja različnih produktov in cenovne diferenciacije. [79]
[332] S prodajno pogodbo se prodajalec zavezuje, da bo izdelek, ki ga prodaja, izročil kupcu in bo slednji s tem pridobil lastninsko pravico, kupec pa se zavezuje, da bo prodajalcu plačal kupnino [66].
[333] Matično področje za licenčno pogodbo je Obligacijski zakonik (XIV. poglavje, členi 704 – 724), in sicer v zvezi s pravicami industrijske lastnine, ki se *lex generalis* uporabijo tudi za računalniške programe.
[334] Če je pravilo o pisnosti kršeno, se sporne ali nejasne določbe razlagajo v korist avtorja (ZASP [133], člen 80/2).
[335] Komentar ZASP [104], str. 273.

A.6.2 Vrste licenčnih pogodb za programsko opremo

a) Licenčne pogodbe za končne uporabnike

a.1.) Podpisane licenčne pogodbe: V licenčni pogodbi se navedejo pogoji, pod katerimi lahko uporabnik uporablja programsko opremo, ki je predmet te pogodbe.[336] Večinoma so te pogodbe rezultat pogajanj med ponudnikom in uporabnikom, zato o njihovi veljavnosti načeloma ni pomislekov.

a.2.) »*Shrink-wrap*« ali ovojnične licenčne pogodbe: To so licenčne pogodbe, ki so priložene računalniškemu programu (v paketu[337], kjer je pomnilniški nosilec s programom) v tiskani obliki. Tovrstne pogodbe spremljajo predvsem standardno, paketno programsko opremo, ki se distribuira prek različnih distributerjev v obliki strojne kode in so le avtorsko, redko tudi patentno varovani. V tej pogodbi so enaki pogoji kakor na podpisanih licenčnih pogodbah, edina razlika je v tem, da pogodb uporabnik ne podpiše. Ker lahko uporabnik tako pogodbo dokaj preprosto spregleda (uporabnik naj bi pogoje iz pogodbe sprejel v trenutku, ko raztrga ovojnico pogodbe), se določbe licenčne pogodbe ponovijo na zaslonskih slikah programa in uporabnik mora z aktivnim posegom potrditi svoje strinjanje s temi določbami, kar se enači z uporabnikovim podpisom licenčne pogodbe. Zaradi omenjenih in drugih problemov prilagajo ponudniki v paket še registracijsko kartico, ki naj bi jo uporabnik podpisano vrnil ponudniku (nič pa ga k temu ne zavezuje). Za licenčne pogodbe "shrink-wrap" velja, da je treba vsa nejasna določila razlagati v korist uporabnikov[338]. Kljub pravnim pomislekom[339] je poslovna praksa tovrstne licenčne pogodbe sprejela kot običajen način sklepanja pogodb.

a.3.) "*Click-wrap*" licenčne pogodbe: Teoretično so to digitalne izvedenke "shrink-wrap" licenčnih pogodb. Z njimi se srečamo v primeru e-poslovanja in spletnih trgovin. Do sklenitve te pogodbe pride pred ali po opravljeni transakciji. Uporabnik mora tako pogodbo na spletni strani prebrati in jo z nekim aktivnim dejanjem potrditi (navadno s klikom na ustrezen gumb). Po Zakonu o elektronskem poslovanju in elektronskem podpisu [134] so te pogodbe popolnoma veljavne.

b) Druge licenčne pogodbe

Gre za mešane pogodbe, ki vsebujejo določila o licenci. Skoraj vedno so podpisane, torej sledijo nekim pogajanjem.

b.1.) Pogodba o reprodukciji in distribuciji: To je pogosta vrsta pogodbe, ki jo srečamo pri vseh kategorijah programske opreme. Sklene se med avtorjem programske opreme in založnikom (podjetjem, ki skrbi za reprodukcijo in distribucijo) ter založnikom in posameznimi distributerji. Pogosto prihaja med

[336] Ti so: pogoji uporabe, prepoved prenosa pravic, druge omejitve ter omejitve odgovornosti za stvarne napake ter odškodninske odgovornosti [66].
[337] Ime je ta licenčna pogodba dobila po tanki prozorni ovojnici na zunanji strani paketa s programom, pod katero naj bi bila, dejansko pa je pogodba v paketu.
[338] 83. člen OZ [116].
[339] Glej [78], str. 299 – 385.

avtorji in distributerji do sporov, ki so posledica kršenja predvidenih rokov za dokončanje projekta s strani avtorja, nedelovanja programske opreme v skladu z dokumentacijo, nepravilnega delovanja programske opreme ali neizvedene določene dogovorjene funkcionalnosti in podobno. Zato so pogodbe med avtorji in distributerji večinoma najkompleksnejše in najnatančnejše od vseh licenčnih pogodb za programsko opremo [66].

b.2.) Pogodba o hrambi izvorne kode (ang. *escrow agreement*): Pri hrambi izvorne kode gre za depozit izvorne kode ter pripadajoče tehnične in uporabniške dokumentacije pri tretji osebi[340], da ta preda kodo uporabniku pod pogoji iz pogodbe. V primeru licenčnih pogodb rešuje hramba izvorne kode končne uporabnike pred problemom neustreznega vzdrževanja programske opreme s strani ponudnika in problem stečaja ponudnika. Če torej želi uporabnik sam premostiti omenjena problema, mora imeti na voljo izvorno kodo. Za uporabnika so posebej pomembne določbe o uporabi, reprodukciji in spremembah izvorne kode ter tehničnih informacijah, pridobljenih iz "hrambe računalniške kode", saj lahko nad izvorno kodo počne samo tisto, kar mu je s temi določbami dovoljeno. Pogosto razvijalci programske opreme, katere izvorno kodo posredujejo v hrambo, če uporabniki deponirano programsko opremo uporabijo pri izdelavi nove programske opreme (izpeljano delo), od uporabnikov zahtevajo prenos pravic intelektualne lastnine nazaj na njih.

b.3.) Pogodbe o razvoju individualne programske opreme

V bistvu gre za pogodbe o naročilu avtorskega dela, ki vsebujejo tudi določila licenčne pogodbe. Po ZASP (112. člen) se pri naročilu računalniškega programa materialne pravice izključno in neomejeno prenesejo na naročnika ali delodajalca. Če želijo avtorji obdržati materialne avtorske pravice, morajo skleniti tudi licenčno pogodbo o prenosu materialnih avtorskih pravic nazaj na avtorja ali ustrezne določbe vključiti v pogodbo o naročilu.

Licenciranje prilagojene aplikacijske (ang. *off-the-shelf*) programske opreme

Prilagojena ali aplikacijska programska oprema je navadno rezultat sodelovanja med razvijalci/programerji in stranko/naročnikom, za katerega specifične potrebe se ta programska oprema zgradi. Pri določanju elementov licenčne pogodbe za tako programsko opremo je treba določiti:

- kdo bo lastnik programske opreme, ko bo ta končana, in kdo bo imetnik (lastnik) avtorskih pravic (lahko je dajalec licence – razvijalec, lahko je pridobitelj licence – naročnik, lahko pa oba); zelo pogosto, predvsem ko je razvijalec veliko podjetje, ki svojo programsko opremo prilagaja tudi za druge naročnike, dobi naročnik le licenco za uporabo tega izdelka;
- trajanje licence (za določen čas ali za vedno);
- koliko licenc potrebuje naročnik za normalno delo (uporabo programske opreme);

[340] V tujini so za hrambo izvorne kode razvita specializirana podjetja, lahko pa to opravljajo tudi notarji, banke, zavarovalnice ipd.

- kdo vse ima licenco (pravico) za uporabo te programske opreme (samo naročnik ali tudi druga konkurenčna podjetja); če je licenca ekskluzivna, potem lahko programsko opremo uporablja samo naročnik, če pa isto programsko opremo uporabljajo tudi druge stranke, je treba v pogodbi navesti omejitve;
- ali bodo v pogodbi zajeta tudi bodoče vzdrževanje in dostop do nadgradenj programske opreme, ki je predmet pogodbe;
- ali dajalec licence jamči za dogovorjeno, pravilno delovanje naročene programske opreme;
- varovanje poslovnih skrivnosti s strani razvijalca[341].

A.6.3 Običajna določila v licenčnih pogodbah za programsko opremo

Z analizo licenčnih pogodb (v 6. poglavju so te pogodbe imenovane kot standardne licenčne pogodbe), ki je v nadaljevanju služila kot temelj za izdelavo predloga vsebine licenčne pogodbe za ponovno uporabno komponento, lahko hitro izluščimo nek standarden nabor določil, ki so vključena v vsako licenčno pogodbo za programsko opremo. Določila so sicer lahko različno poimenovana, vendar je njihova vsebina v različnih pogodbah enaka ali vsaj zelo podobna.

Elementi, na katere se nanaša pogodba: V licenčnih pogodbah je pomembno zelo natančno določiti, na kaj se te pogodbe nanašajo. Predmet licenčne pogodbe je lahko samo računalniški program, lahko so pa tudi vzdrževanje, popravki in razširitve tega programa.

Definicije: Definiranje ključnih tehničnih in poslovnih terminov pripomore k zmanjšanju nesporazumov med osebjem ponudnika in uporabnika ter k sodnikovemu razumevanju problema v primeru pravnega spora.

Dobavni roki: Predvsem so ti pomembni pri programski opremi po naročilu ali programski opremi za večje sisteme, kjer zamude lahko povzročijo velikanske škode.

Zagotovila o sposobnosti in ustreznosti ponudnika: V pogodbi naj bi bila tudi določba o tem, da je dajalec licence (ponudnik, avtor) izkušen v izdelavi programskih rešitev za aktualno področje.

Priprava računalniškega sistema za namestitev: Ponudnik navadno postavi zahteve, ki jih mora izpolnjevati računalniški sistem, uporabnik pa mora sistem uskladiti z zahtevami. V pogodbo je možno vključiti določbo, da ponudnik pred namestitvijo preveri primernost sistema. Seveda so te določbe zanimive samo v primeru naročila nestandardne programske opreme.

[341] Naročnik dejansko sodeluje pri gradnji njemu prilagojene programske opreme, prilagoditve pa se nanašajo na naročnikove poslovne metode, koncepte poslovanja, terminologijo, v programsko opremo so vgrajene naročnikove ideje ipd., zato mora s pogodbo doseči zagotovilo, da razvijalec teh poslovnih skrivnosti ne bo izdal ali uporabil pri prilagajanju programske opreme za druge naročnike ali konkurenco.

Namestitev: Določiti je treba, ali je namestitev programa že vključena v ceno ali ne. Ob namestitvi mora ponudnik opraviti tudi preizkušanje programa (testiranje).

Objektna-izvorna koda: Navadno je predmet licence le objektna koda, izvorno kodo pa je mogoče dobiti le na podlagi posebne licence.

Določila o razvoju programske opreme: Lahko gre za razvoj nove, individualne programske opreme ali za prilagoditve obstoječe standardne ali individualne programske opreme določenemu uporabniku.

Svetovalne storitve, prilagoditve sistemov in razvoj: Prilagoditve in nastavitve računalniškega sistema ob namestitvi programske opreme se opravljajo kot svetovalne storitve. Če so zelo obsežne, je treba skleniti novo pogodbo ali v osnovni pogodbi natančno določiti obseg storitev, njihove cene ter roke in sankcije za neizpolnjevanje določb.

Spremembe in vzdrževanje računalniškega programa ter pravice na spremenjenem programu: Zakoniti uporabnik programa lahko sam prevaja, prilagodi, priredi ali drugače predela program brez dovoljenja avtorja, če je to potrebno zaradi uporabe programa v skladu z njegovim namenom.[342] Kakršnekoli drugačne določbe, v nasprotju z navedenim, so nične. Avtorske pravice na spremembah ima tisti, ki je predelavo opravil, torej v tem primeru pridobitelj licence.[343] Z licenčno pogodbo pa lahko ponudnik programske opreme na uporabnika prenese pravico do sprememb programa zunaj okvira, ki ga določa zakon, torej mu lahko dovoli tudi nadaljnji razvoj programa (pogosto pri specialni programski opremi, ki jo uporabnik dobi v obliki izvorne kode). Izvorno kodo lahko uporabnik uporabi samo za razvoj programske opreme za lastne potrebe. Lahko pa pridobi tudi pravico do dajanja podlicenc za spremenjeni del, vendar le v obliki strojne kode (v nasprotnem primeru bi lahko konkuriral ponudniku). Če je zakoniti pridobitelj licence (uporabnik) opravil nad predmetom licence spremembe ali izboljšave, jih vključil v svoj izdelek ter ga ponudil tudi drugim uporabnikom, je mogoče v licenčnih pogodbah najti določbe, ki prenašajo vse pravice na spremembah na dajalca licence. Če spremembe v programu opravi ponudnik na podlagi pogodbe o naročilu, je treba posebej določiti, da ima materialne in druge avtorske pravice na spremembah ponudnik, saj v nasprotnem primeru velja, da jih ima naročnik (uporabnik).[344]

Dovoljenje za uporabo programa na rezervni lokaciji oziroma rezervni namestitvi: Če mora imeti uporabnik zaradi zagotavljanja učinkovitega in varnega poslovanja nameščen program istočasno tudi na rezervnih lokacijah, mora biti to v pogodbi izrecno določeno.

Dostop do računalniškega sistema: Pri večjih projektih mora uporabnik dajalcu licence omogočiti dostop do sistema (za namestitev, vzdrževanje, izobraževanje in razvoj), vendar je treba natančno določiti časovne omejitve takšnega dostopa.

[342] 1. odstavek 114. člena ZASP [133].
[343] 2. točka 2. odstavka 113. člena ZASP [133].
[344] 113. člen ZASP [133].

Glavno vodilo komercialnih ponudnikov programske opreme je prav gotovo čim večji dobiček. Ni pa dobiček edini razlog za uporabo licenčnih pogodb. Na to kaže pojav brezplačnih programov (ang. *freeware*) in programov z odprto kodo (ang. *open-source*). Na teh dveh vrstah programske opreme avtorske pravice obstajajo, pravica do uporabe pa se prenaša z licenco. Pri odprtokodnih programih ne gre za posebno vrsto licence, ampak za določila o tem, kakšne morajo biti licence, da bodo usklajene z doktrino "*open-source*"[345]. Skupne značilnosti licenc za odprtokodno programsko opremo so: prosta distribucija skupaj z izvorno kodo; prosto spreminjanje in distribuiranje spremenjene programske opreme, tudi v komercialne namene, s tem, da morajo biti spremembe izvorne kode jasno ločene od originalne izvorne kode; odprtokodno programsko opremo lahko uporablja kdorkoli v poljubne namene; licenca se prenese na tistega, ki mu je bila programska oprema distribuirana; licenca mora biti nevtralna do tehnologije.[346] Sodna praksa še ni potrdila pravne veljavnosti licenc za odprtokodno programsko opremo, pa tudi odškodnina za kršitev licence ni dorečena.

Pravna teorija razlaga pravno naravo licenčnih pogodb na različne načine. Nekateri teoretiki menijo, da gre za pogodbo *sui generis*, drugi pa, da gre le za obliko rabokupne pogodbe. Pri licenčnih pogodbah za programsko opremo skoraj nikoli ne gre za čiste licenčne pogodbe, ampak za mešane pogodbe, kar pomeni, da vsebujejo določila o licenci, podjemu, hrambi, prodaji ter drugih pravnih institutih. Ker se z licenčno pogodbo za programsko opremo prenašajo posamične materialne avtorske pravice od avtorjev na uporabnike programske opreme, gre za avtorsko pogodbo, zato se za te licenčne pogodbe primarno uporabljajo pravila avtorskega pogodbenega prava[347], sekundarno pa pravila o licenčni pogodbi iz OZ.

A.6.4 Mednarodne licenčne pogodbe

Tržišče računalniške tehnologije je razpredeno po vsem svetu, zato potreba po enotnem mednarodnem pravnem varstvu narašča. Zaradi velikih razlik v sistemih pravnega varstva pravic intelektualne lastnine, v tradicijah, kulturi, terminologijah, postopkih in zakonodajnih ogrodjih v posameznih državah je sprejetje takega globalnega varstva zelo zahtevno. Pri sklepanju mednarodnih licenčnih pogodb je zato treba upoštevati vse te razlike[348].
Mednarodni dogovor o pravnih normah je nujen za uspešno rast globalne digitalne ekonomije. Po Samuelsonovi [94] je globalna digitalna ekonomija sektor z velikimi potenciali, kateremu je nujno treba posvetiti dovolj pozornosti tudi na pravnem področju.

[345] Richard Stallman, ustanovitelj Free Software Foundation ter eden od avtorjev licence GNU GPL (*General Public Licence*) je te licence poimenoval "*copyleft*", saj uporabniku omogočajo nadaljnje kopiranje oziroma distribucijo v nasprotju z licenco "*copyright*".
[346] The Open Source Definition, version 2, junij 1991 [http://www.gnu.org/copyleft/gpl.html], 13.8.2004.
[347] Predvsem členi 73-84 ZASP [133].
[348] Pomemben zgodovinski korak na področju mednarodnega varstva računalniške tehnologije je bil dosežen z dogovorom TRIPS, ki je zlil koncepte Pariške in Bernske deklaracije in tako zajel večino vej intelektualne lastnine na enem mestu, uporabnih za večino držav. Dobro podlago za sprejetje take mednarodne pravne ureditve pa je tudi WIPO Copyright Treaty (glej 3.1.1 stran 48).

PRILOGA B:
ORODJA V PODPORO IZVAJANJU PONOVNE UPORABE

Celoten proces oziroma razvoj mora biti ustrezno podprt z orodji, ki omogočajo, spodbujajo in lajšajo ponovno uporabo. Orodja v podporo ponovni uporabi bistveno prispevajo h kakovosti komponent. Navadno so ta orodja nadgradnja oziroma dopolnitev splošnih programskih načrtovalskih in razvojnih orodij. Ena njihovih najpomembnejših lastnosti je obvladovanje ponovno uporabnih lastnosti komponent.

Nabor klasičnih tipov orodij, ki je potreben pri izvajanju razvoja na temelju ponovne uporabe, je prikazan v tabeli 3. Tabela povzema ugotovitve in predloge številnih avtorjev člankov in knjig iz seznamov A in C. Orodja so razvrščena v skupine ali kategorije, glede na to, v kateri fazi razvojnega življenjskega cikla jih je mogoče uporabiti, oziroma glede na procese, aktivnosti in opravila, ki jih podpirajo. Znotraj posamezne kategorije so razvrščena po tipih, glede na svojo funkcijo.

Kategorija orodja	Tip orodja	Funkcije orodja
Analiza in načrtovanje	Orodja za analizo in načrtovanje domene	Pomoč inženirjem domene pri razpoznavanju podobnosti med elementi domene in preverjanju sovpadanja elementov z obstoječimi modeli in arhitekturami. Pomoč razvijalcem pri pridobivanju in izboljševanju nabora modelov in arhitektur domene za ponovno uporabo.
	Orodja za analizo obstoječih izdelkov	Analiza obstoječih izdelkov in določanje strukturnih in funkcijskih vzorcev podobnosti.
	Orodja za analizo aplikacijskih zahtev	Navzkrižno preverjanje zahtev za obstoječe izdelke, kar omogoča zmanjšanje razpona med obstoječim in zahtevanim oziroma potrebnim.
	Orodja za načrtovanje aplikacije	Pregledovanje obstoječih arhitektur domene in izdelava seznama možnosti uporabe arhitekturnih komponent. Rezultat pregledovanja je formalna specifikacija načrta za programski izdelek, ki ga lahko obdelata orodji za dokumentiranje in razvoj novih izdelkov.
Gradnja komponent	Pametni urejevalniki	Poišče ustrezne izdelke in jih razčleni. Razvijalec nato lahko izdelke poveže z ustreznim kontekstom.
	Generatorji	Na temelju specifikacije načrta in vgrajenih informacij o domeni zgradi izdelek.
	Zbiralci	Zgradi izdelek na temelju specifikacije načrta in izdelkov iz drugih virov (izven orodja samega).
	Orodja za spreminjanje pogojev obstoječih komponent	Pakiranje želenih vzorcev, pridobljenih z analizo obstoječih izdelkov, v nove izdelke.
Preskušanje komponent	Preskuševalci prilagodljivosti	Inženirjem domene pomaga določiti in povečati preprostost ponovne uporabe danega izdelka.
	Preskuševalci splošnosti	Inženirjem domene pomaga določiti in spremeniti domeno uporabe določenega izdelka.
Upravljanje ponovne uporabe	Orodja za merjenje pridobitev pon. uporabe (stroški/pridobitve): - življenjskega cikla komponente - življenjskega cikla aplikacije	Določa stroške shranjevanja in iskanja izdelkov ter njihove amortizacije. Določa relativne stroške in pridobitve posedovanja ponovno uporabnih izdelkov. Določa večji oziroma prihranjen čas in napor razvoja in vzdrževanja pri ponovni uporabi.
	Upravljanje konfiguracije in različic komponente	Vzdržuje sled o načinu dostopa do izdelka, lastništvu izdelka, servisnih obveznostih ter različici izdelka, ki je v nekem programskem sistemu uporabljena.
	Orodje za analizo posledic modifikacij komponente	Vzdržuje sled o ponovnih uporabah nekega izdelka ter odvisnostih med izdelki.
	Orodje za analizo komponent na zalogi	Določa morebitna podvajanja ali sovpadanja izdelkov na zalogi ter starost in status posameznih elementov zaloge komponent.
	Orodje za katalogiziranje komponent	Formalen vpis izdelka v različne mehanizme shranjevanja in iskanja, vključno z osveževanjem iskalnih orodij in brskalnikov z deskriptorji ter iskalnimi pogoji.
	Orodje za iskanje in izbiranje komponent	Brskanje in dostop do izdelkov prek nastavitev parametrov tako za razvojni čas kakor za čas izvajanja.
	Orodje za certificiranje komponent	Podpira varno certificiranje izdelka glede na namembnost ponovne uporabe – znotraj projekta, oddelka, organizacije, veje industrije ipd.

Tabela 3: Nabor klasičnih tipov orodij, ki je potreben pri izvajanju razvoja na temelju ponovne uporabe.

PRILOGA C: RAZŠIRITVE TEMELJNEGA PODATKOVNEGA MODELA

C.1 Ocenjevanje in certificiranje

Razširitev, ki ureja področje ocenjevanja in certificiranja programskih komponent, je v okviru IEEE tudi sprejeta kot standard [63] in predstavlja ogrodje za označevanje in opisovanje postopkov ocenjevanja in certificiranja ter njihovih rezultatov. Ocenjevanje in certificiranje komponente se nanašata na proces določanja, ali je neka komponenta v skladu s predhodno postavljenimi merili zanjo, ki vključujejo tudi merila za sprejem komponente v knjižnico, (t.j. ocenjevanje ustreznosti komponente), ocenjevanje nepodvajanja komponente (preverjanje, ali že obstajajo podobne komponente v knjižnici) ter preverjanje sovpadanja komponente s standardi ponovne uporabe.

Standard privzema, da večina knjižnic organizira svoje postopke certificiranja in ocenjevanja nivojsko, saj omogoča tak način boljšo organizacijo posameznih aktivnosti certificiranja, za uporabnika pa nivoji predstavljajo informacijo ali referenco o tem, katera merila kakovosti dosega neka komponenta. Splošno predstavlja višji nivo tudi višjo stopnjo zaupanja v komponento, kar pa predstavlja tudi višje stroške certificiranja. Navadno vsaka knjižnica zase in po svojih merilih postavi merila za posamezne nivoje, zato se pri sodelovanju in povezovanju knjižnic z različnimi nivoji pojavi problem, saj različna merila uporabnika več med seboj sodelujočih knjižnic zmedejo. Dejstvo pa je, da mora biti posamezni knjižnici omogočeno, da po svoje definira postopke certificiranja, glede na njeno poslanstvo in skladno s standardi, specifičnimi za domeno te knjižnice. Ker je bilo stališče oblikovalcev standarda tako, da bi bilo nesmiselno predpisati standardno množico nivojev, so kot razširitev podatkovnega modela raje postavili standard, ki opisuje, kako naj se postopki preverjanj, ocenjevanj in certificiranj organizirajo.

Standard predstavlja skupni temelj, na katerem se da primerjati med seboj različne postopke in temelj za razumevanje ocenjevalnih in certifikacijskih aktivnosti ter rezultatov. Potencialni ponovni uporabnik komponente si lahko z informacijami o postopkih certificiranja in ocenjevanja ter o rezultatih za posamezno komponento pomaga pri primerjanju postopkov ocenjevanja in certificiranja posameznih knjižnic, razumevanju postopkov certificiranja neke knjižnice in razumevanju ocenjevalnih aktivnosti, ki so bile izvedene nad neko komponento.

Tehnika in pripadajoči podatkovni model, ki se uporabljata pri organiziranju, izbiranju, sporočanju in vodenju procesa certificiranja komponent, predstavljata ogrodje za certificiranje komponente.

Certificiranje je proces ocenjevanja ali preverjanja, ali neka komponenta ustreza vnaprej postavljenim merilom, ki naj bi zanjo veljali. To so t.i. certifikacijska merila, ki se postavijo za vsako komponento, opisujejo pa množico standardov, pravil in lastnosti, ki jim mora komponenta zadoščati, če naj pridobi certifikat ustreznosti[349] za določen nivo.

Certifikacijski nivoji se nanašajo na organizacijo postopkov certificiranja po korakih. Strogost zahtev in meril, katerim mora komponenta zadoščati, narašča s koraki. Pripadnost nekemu nivoju pove, kateri postopki in aktivnosti so se nad neko komponento izvedli v procesu certificiranja. Nivoji ter aktivnosti znotraj njih so odvisni od sredstev in znanja, ki ga ima osebje na voljo, od kakovosti komponent, ki se vključujejo v knjižnico ter potreb uporabnikov knjižničnih komponent, zato mora vsak naslednji predstavlja večje zaupanje v komponento in višjo stopnjo možnosti za njeno ponovno uporabo, hkrati pa tudi večji napor in višje stroške za izvedbo certificiranja. V splošnem naj bi bili nivoji za neko aplikacijsko domeno enaki in naj se ne bi spremenili, četudi se znotraj domene uporabijo različna certifikacijska merila.

Za določitev vrstnega reda nivojev je potrebno neko merilo. Glede na to, da mora potencialni ponovni uporabnik pred dokončno izbiro komponente iz množice kandidatnih komponent oceniti stroške ponovne uporabe te komponente, je smiselno komponente v nivoje razdeliti glede na razmerje med pridobitvami s ponovno uporabo in stroški izvedbe certificiranja znotraj danega nivoja. V standardu je za določitev certifikacijskih nivojev predlagan naslednji nabor oznak:

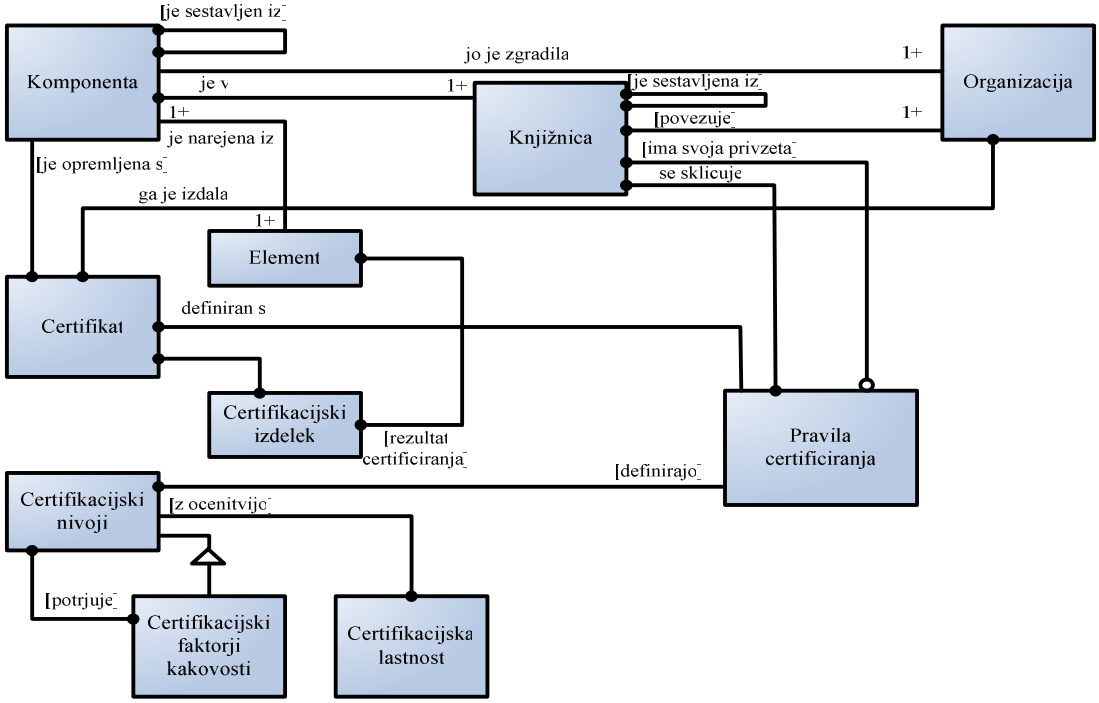
- neocenjena: označba komponente samo z njenim imenom;
- opisana: označba meta podatkov komponente (opisna informacija o komponenti in njeni uporabi);
- analizirana: certificiranje lastnosti komponente s pregledi ali metodami statične analize;
- preskušena: certificiranje lastnosti ali vedenja komponente med njenim izvajanjem.

Certifikacijski nivoji so zelo odvisni od tipa komponente, njenega namena in domene, organizacije knjižnice ponovno uporabnih komponent in števila potencialnih ponovnih uporabnikov komponente. Vsaka knjižnica se sama odloči, koliko in katere certifikacijske postopke bo izvajala nad komponentami, ki naj bi se vključile v knjižnico. Lahko se knjižnica tudi odloči, da sprejme vse komponente brez ocenjevanja ali pa določi minimalno množico meril, katerim morajo ustrezati vse komponente.

Merila so določena s certifikacijskimi pravili in postopki[350], možno pa jih je specificirati kakor množico certifikacijskih lastnosti, ki jih mora neka komponenta imeti. Certifikacijska lastnost je lastnost ali značilnost komponente, ki naj bi se ocenjevala v postopku certificiranja. Lastnost se lahko nanaša na to, kaj posamezna

[349] Potrdilo, da komponenta ustreza certifikacijskim merilom.
[350] Certifikacijski postopek je opis procesa certificiranja, nivojev certificiranj, meril za posamezne nivoje ter seznam uporabljenih metod, katerih rezultati so certifikacijski izdelki (seznami kontrolnih pregledov, metrike, poročila, certifikati ...).

komponenta je, kaj dela ali kakšne so njene povezave in odnosi z okoljem, v katerem deluje. Z oceno certifikacijskih lastnosti se določi ocena faktorja kakovosti komponente[351]. Vrednotenje certifikacijskih lastnosti lahko vključuje samo določene elemente (sestavne dele) komponente, so pa seveda lastnosti odvisne od tipa komponente oziroma njenih elementov.



Slika 29: Podatkovni model postopka certificiranja.

Znotraj postopka certificiranja se izvajajo različne certifikacijske metode, npr: pregled[352], statična analiza[353], preskušanje[354], formalna verifikacija[355], modeliranje[356], primerjalni preskus[357] ipd. Uporabljene metode

[351] Faktorji kakovosti komponente so atributi komponente, kot so popolnost, pravilnost, ponovna uporabnost, prenosljivost, uporabnost, funkcionalnost, zanesljivost, učinkovitost, vzdržljivost, varnost. Lahko so upravljavsko ali uporabniško usmerjeni. Katere faktorje kakovosti bo ocenjevala neka knjižnica za svoje komponente, je sestavni del postopkov certificiranja knjižnice.
[352] Proces, ki ga navadno izvedejo ljudje, z njim pa preverijo prisotnost določenih lastnosti v izdelku, ki ga pregledujejo.
[353] Avtomatiziran proces, s katerim naj bi pokazali, da ima izdelek neko posebno lastnost. Ocenjevanje komponente poteka na temelju njene oblike, strukture, vsebine ali dokumentacije (ne da bi prišlo do izvajanja izvorne kode).
[354] Eksperimentalni pristop, s katerim se dokazuje, da izdelek poseduje neko splošno lastnost. Lahko se izvede nad vsako izvedljivo komponento. Gre torej za izvajanje komponente na primerih, torej pod določenimi pogoji, pri tem pa se rezultati izvajanj opazujejo in zapisujejo ter se hkrati ocenjujejo določeni vidiki komponente.
[355] Proces dokazovanja (oblikovanja ali izpeljave dokaza), da izdelek poseduje neko lastnost.
[356] Proces izdelave formalne abstrakcije nekega izdelka za analiziranje neke lastnosti. Ta lastnost se navadno nanaša na delovanje izdelka in je njeno prisotnost s preskušanjem ali analizo težko ugotoviti ali izmeriti.
[357] Proces preskušanja zmogljivosti komponent v času izvajanja, npr. hitrost dostopa do diska, uporabo spomina, numerično natančnost, odzivni čas na določen dogodek.

certificiranja in lastnosti, ki se ocenjujejo, so odvisne od tipov elementov, ki neko komponento sestavljajo. Elementi so posamezni sestavni deli neke komponente (npr. dokumenti, specifikacije zahtev, preskusni primeri, izvorna koda, navodila za namestitev ter "beri-me" datoteke, algoritmi, arhitekture, podatkovne baze, podatkovni slovarji, podatkovne datoteke, načrti, modeli, razvojni plani in razvojni standardi, preskusni podatki, primeri in programi, rezultati certificiranj oziroma certifikacijski izdelki, podatki o različicah, dokumentacija, licenčni dogovori itn.).

Komponento sestavlja eden ali več elementov, ki so shranjeni v ločenih datotekah. Vsaka sprememba enega ali več elementov nujno vodi v novo različico komponente, kar zahteva nov certifikacijski postopek. Certificiranje komponente vključuje izvedbo certifikacijskega procesa nad njenimi elementi. Certifikat in vsi pripadajoči certifikacijski izdelki, ki so rezultat certificiranja, se nanašajo na specifično različico komponente in njenih elementov.

Organizacija lahko izbere isti certifikacijski postopek za vse komponente ali pa več različnih postopkov za posamezne družine komponent. Lahko se tudi odloči za nivojsko certificiranje, vendar brez določanja certifikacijskih faktorjev kakovosti, kar je primerneje za preprostejše certifikacijske postopke.
Certifikat ali potrdilo dokazuje, da je bila komponenta ocenjena in ovrednotena po določenih certifikacijskih meril. Za vsak uspešno izveden nivo certificiranja se izda ločen certifikat. Neka komponenta lahko pridobi večje število certifikatov. Če certificiranje komponente izvaja več različnih organizacij, vsaka izda svoj certifikat. Različnim postopkom certificiranja iste komponente morajo slediti različni certifikati. Tudi če se certificiranje izvaja v časovno različnih terminih, se za vsako uspešno preskušanje izda ločen certifikat.

C.2 Pravice intelektualne lastnine

Drugo področje, za katerega je bilo postavljeno standardno ogrodje [64], je področje pravic intelektualne lastnine, nanaša pa se na opisovanje in označevanje pravic intelektualne lastnine in drugih pravnih omejitev na komponentah knjižnice.

Standard [64] opisuje postopke upravljanja s pravicami in izmenjavo informacij o pravnih omejitvah na posameznih komponentah med knjižnicami, podaja smernice za izmenjavo informacij, za razširjanje informacij o postopkih upravljanja s pravicami, načinih razpečevanja (distribuiranja) programskih komponent in rezultatih uporabe postopkov nad komponentami, ki si jih medsebojno sodelujoče knjižnice izmenjujejo oziroma delijo. Velika slabost in omejitev tega standarda je njegova omejenost na pravne ureditve ZDA. Problem mednarodne izmenjave ali celo mednarodnega knjižničnega povezovanja torej ni rešen oziroma ga ta standard ne obvladuje, kar še dodatno dokazuje, da je problem neenotne obravnave pravic intelektualne lastnine na programskih komponentah močno prisoten.

Ponovno uporabna knjižnica je skupek komponent, članov knjižnice, pravil in postopkov, knjižničnih mehanizmov ter vseh spremljajočih podpornih dejavnosti in storitev. Pravila upravljanja s pravicami intelektualne lastnine ponovno uporabne knjižnice se nanašajo na postopke določanja in uveljavljanja pravnih omejitev ponovne uporabe knjižničnih komponent. Namen standarda za zapisovanje pravnih omejitev je izboljšati razumevanje pravic posameznih strank in dolžnosti ponovnega uporabnika komponente.

Medsebojno sodelovanje knjižnic in medsebojna delitev komponent, ki jih te knjižnice tržijo, v veliki meri prispevata k razpoznavnosti in vrednosti posamezne knjižnice, poleg tega pa je na ta način zagotovljena veliko večja vidnost posamezne komponente ter s tem tudi možnost njene ponovne uporabe. Delitev komponent, nad katerimi veljajo pravne omejitve, je lahko zelo nerodna, saj navadno zahteva pogajanja ali sklepanja ločenih dogovorov za vsako posamezno komponento in za vsak par med seboj sodelujočih knjižnic. Eden glavnih ciljev tega ogrodja je omogočiti vnaprejšnja pogajanja o licencah med ponovno uporabnimi knjižnicami, kar olajša razširjeno ponovno uporabo "omejene" programske opreme in izloči potrebo po ločenih pogajanjih.

Ogrodje je bilo zgrajeno, da bi bilo mogoče nedvoumno opisati in zapisati vse izvozne[358] in pravne omejitve nad programsko opremo ter s tem za ponovno uporabno knjižnico zmanjšati tveganja in bojazni pred odgovornostmi in pravdanji, ki v veliki meri preprečujejo ali omejujejo izmenjavo programske opreme med knjižnicami.

V standardu je posebej poudarjeno, da njegov namen ni pravno svetovanje, saj se omejuje le na podajanje definicij pravnih terminov in aktov, ki so potrebne za lažje razumevanje vsebine standarda. Pravni opisi in elementi se nanašajo izključno samo na pravo ZDA in niso v celoti prenosljivi na druge države zunaj ZDA.

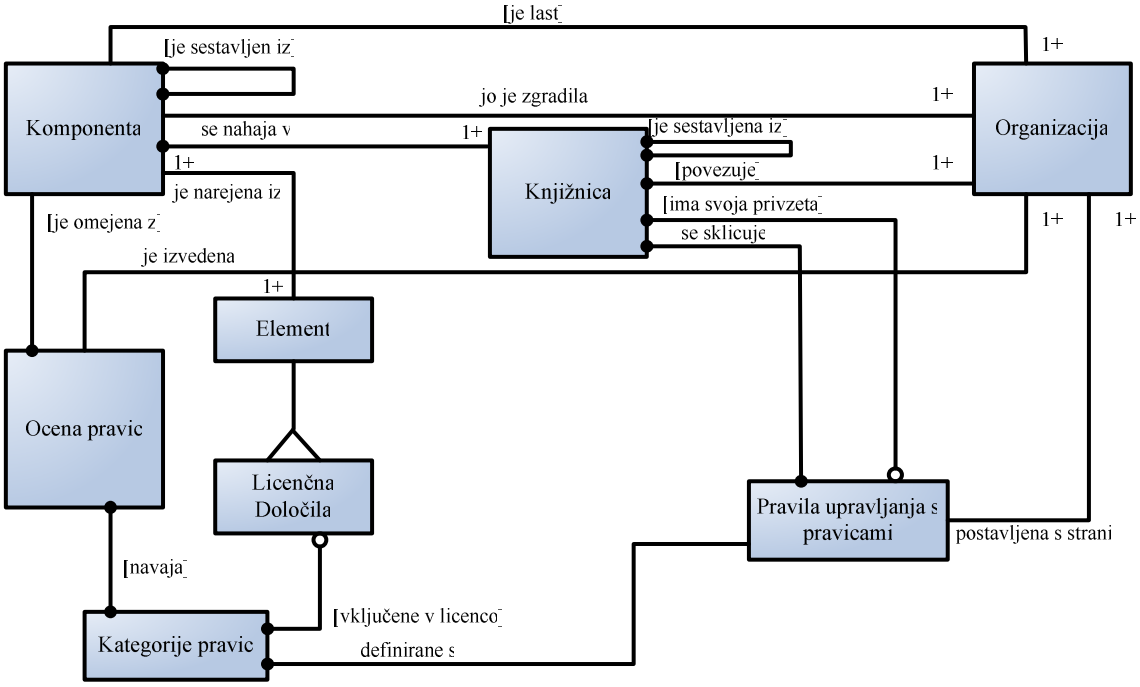
Ogrodje od vsake knjižnice zahteva, da definira svoja pravila upravljanja s pravicami intelektualne lastnine, predpisuje pa enotno obliko organizacije in sporočanja teh pravil.

Njegov namen je, da bi:

- zagotavljalo temelj za razumevanje pravil upravljanja s pravicami intelektualne lastnine,
- zagotavljalo temelj za razumevanje, katere pravice so povezane s posamezno komponento in katere omejitve ter odgovornosti se nanašajo na ponovnega uporabnika,
- definiralo standardni model za medsebojno elektronsko izmenjavo pravil upravljanja s pravicami intelektualne lastnine,

[358] Postavlja se vprašanje, ali je v času globalizacije in globalnega trga sploh smiselno govoriti o izvozu in izvoznih omejitvah. Pri mednarodnih virtualnih knjižnicah bi lahko problem ugotavljanja kršitve pravic intelektualne lastnine ali kaznovanja v primeru kršitev ugotavljali in reševali tako, da bi za ugotavljanje kršitev pravic na komponentah veljala zakonodaja države imetnika lastninske pravice na komponenti, za ugotavljanje kršitev pravic pri uporabi knjižnice pa zakonodaja države knjižnice. Teoretično bi se lahko pojavili problemi mešanega lastništva, posebno pri izpeljanih ponovno uporabnih komponentah, kar bi zapletlo podano pravilo. Edina prava rešitev bi seveda bilo poenotenje pravne ureditve intelektualne lastnine, česar pa v zelo kratkem času ni pričakovati.

- definiralo standardni model za medsebojno elektronsko izmenjavo informacij o pravicah intelektualne lastnine, pravnih omejitvah in povezanih sorodnih pravicah na posamezni komponenti.



Slika 30: Podatkovni model za pravice intelektualne lastnine.

Tudi standard označevanja pravnih omejitev na komponenti je priporočljiv kot temelj vsakega knjižničnega modela, tudi interne knjižnice, zato je koncept tega označevanja v nadaljevanju podrobneje opisan kakor sledi iz [64].

Z licenčnimi določili specificiramo zahteve, povezane z licenciranjem. Neko licenčno določilo se lahko nanaša bodisi na komponento ali elemente, ki jo tvorijo, bodisi na posamezno kategorijo ali vrsto pravice. Ko izvedemo postopek ocenjevanja pravic na neki komponenti, zanjo pridobimo oceno pravic. Če ocenjevanje izvaja več organizacij, potem se posamezne ocene shranjujejo ločeno. Tudi ob različnih časih izvedene ocene se shranjujejo ločeno. Rezultati ocen pravic nad neko komponento določajo kategorijo pravic za to komponento. Kategorija pravic se neki komponenti priredi, ko gre ta skozi postopek ocenjevanja pravic, v bistvu pa je s kategorijo pravic določen tudi dostop do komponente. Kategorije pravic so vključene v licenčna določila za komponento, opredeljene pa so v pravilih upravljanja s pravicami, ki jih postavi ali sprejme posamezna organizacija. Slednja so sestavni del splošnih pravil knjižnice.

V šestem poglavju točka 6.1.4, sem definirala mehanizem prirejanja vzorcev licenčnih pogodb posameznim komponentam, ki se vključujejo v knjižnico, ter ustrezno nadgradili zgornji podatkovni model za pravice intelektualne lastnine (glej sliko 30).

PRILOGA D: POJMOVNIK

A

ABSD *(Asset Based Software Development)*
Tehnologija razvoja na temelju ponovno uporabnih izdelkov.

analiza domene *(domain analysis)*
Proces, v katerem se identificirajo, zbirajo, organizirajo, analizirajo in predstavljajo podobnosti ter razlike med obstoječimi sistemi neke aplikacijske domene. Na temelju te analize se definirajo programske arhitekture ali ogrodja, ki podpirajo gradnjo novih sistemov ali aplikacij te domene.

aplikacija *(aplication, custom aplication)*
Glej programska oprema.

arhitektura domene *(domain architecture)*
Splošen načrt programskega sistema domene, ki ponazarja organizacijsko strukturo tega sistema.

avtorska pravica *(copyright)*
Ključno in najpomembnejše pravno sredstvo za obvladovanje pretoka idej, znanja in produktov znanja.

B

bela komponenta (*white-box component, white-box asset)*
Komponenta, katere vidljivost je popolna in jo je mogoče poljubno spreminjati ter prilagajati.

bela ponovna uporaba *(white-box reuse)*
Ponovna uporaba belih komponent, ki razvijalcem omogoča prilagoditev komponent trenutnim zahtevam s spreminjanjem samih komponent.

blagovna ali storitvena znamka *(trade mark)*
Pravno sredstvo, s katerim se varujejo imena, simboli, logotipi, slogani, ki se uporabljajo za identifikacijo nekega izdelka ali njegovega izvora.

C

CBSD *(Component Based Software Development)*
Tehnologija objektnega razvoja na temelju črnih, izvršljivih ponovno uporabnih komponent in ogrodij.

certificiranje *(certification process)*
Proces ocenjevanja ali preverjanja, ali neka komponenta ustreza vnaprej postavljenim merilom, ki naj bi zanjo veljala. To so t.i. certifikacijska merila.

certifikacijska lastnost *(certification property)*
Lastnost ali značilnost komponente, ki naj bi se ocenjevala v postopku certificiranja. Lastnost se lahko nanaša na to, kaj posamezna komponenta je, kaj dela ali kakšne so njene povezave in odnosi z okoljem, v katerem deluje. Z oceno certifikacijskih lastnosti se določi ocena faktorja kakovosti komponente.

certifikacijska merila *(certification criteria)*
Opisujejo množico standardov, pravil in lastnosti, ki jim mora komponenta zadoščati, če naj pridobi certifikat ustreznosti za določen nivo.

certifikacijske metode *(certification method)*
Metode, ki se izvajajo v postopku certificiranja, npr. pregled, statična analiza, preskušanje, formalna verifikacija, modeliranje, primerjalni preskus.

certifikacijski nivoji *(certification levels)*
Nanašajo se na organizacijo postopkov certificiranja po korakih.

certifikacijski postopek *(certification policy)*
Opis procesa certificiranja, nivojev certificiranj, meril za posamezne nivoje ter seznam uporabljenih metod, katerih rezultati so certifikacijski izdelki (seznami kontrolnih pregledov, metrike, poročila, certifikati ...). Lahko jih je specificiramo kot množico certifikacijskih lastnosti, ki jih mora neka komponenta imeti.

certifikat *(certificate)*
Potrdilo, ki dokazuje, da je bila komponenta ocenjena in ovrednotena po določenih certifikacijskih merilih.

certifikat ustreznosti *(certificate)*
Potrdilo, da komponenta ustreza certifikacijskim merilom.

COTS *(Commercial Off-The-Shelf Components)*
Glej črna komponenta.

Č

črna komponenta *(black-box component, black-box asset)*
Komponenta, ki je ni mogoče na noben način bistveno spremeniti, ker njena notranjost ni vidna (enkapsulacija) in jo je mogoče prilagoditi le kakor prek vmesnika.

črna ponovna uporaba *(black-box reuse)*
Ponovna uporaba črnih komponent, brez spreminjanja in prilagajanja komponent. Način ponovne uporabe, pri kateri komponent ne spreminjamo ali kakorkoli prilagajamo, ampak jih uporabimotakšne, kakršne so.

D

dekompiliranje ali razgradnja *(decompilation)*
Postopek, pri katerem poskušamo s posebnim prevajalnikom na podlagi prevedene kode v strojni jezik vnovič sestaviti prvotni program v višjem programskem jeziku.

dobavitelj ponovno uporabnih komponent *(reusable component supplier)*
Tisti, ki skrbi za trženje ponovno uporabnih komponent. Lahko je zaposlen v isti organizaciji kot razvijalci teh komponent, lahko pa igra podobno vlogo kot založnik v svetu tiskanega gradiva. Predstavljal naj bi vmesni člen med razvijalcem komponent in knjižnico oziroma knjižnicami.

dogovor o varovanju poslovne skrivnosti *(nondicslosure agreement)*
Dogovor med vsemi, ki imajo dostop do neke poslovne skrivnosti, s katerim se obvežejo, da te poslovne skrivnosti ne bodo razkrili oziroma izdali.

domena *(domain)*
Množica sistemov ali aplikacij, ki imajo določene skupne lastnosti in jih je zato mogoče organizirati kot zbirko ponovno uporabnih komponent pri gradnji novih sistemov znotraj domene.

domenska razdalja *(domain distance)*
Razlike med ciljno aplikacijsko domeno (domeno, v kateri bo komponenta ponovno uporabljena) in trenutno domeno (domena, za katero je bila komponenta razvita) ali količino napora, ki ga zahteva spreminjanje funkcionalnosti obstoječe komponente, da bi jo prilagodili natančno določenim trenutnim zahtevam in potrebam.

F

Faktor kakovosti komponente *(certification quality factor)*
Atribut komponente, npr. popolnost, pravilnost, ponovna uporabnost, prenosljivost, uporabnost, funkcionalnost, zanesljivost, učinkovitost, vzdržljivost, varnost. Lahko je upravljavsko ali uporabniško usmerjen. Izbira faktorjev kakovosti, ki jih bo ocenjevala knjižnica za svoje komponente, je sestavni del postopkov certificiranja knjižnice.

FBSD *(Frame Based Software Development)*
Tehnologija neobjektnega razvoja na temelju ponovno uporabnih ogrodij.

formalna verifikacija *(formal verification)*
Proces dokazovanja (oblikovanja ali izpeljave dokaza), da izdelek poseduje neko lastnost.

I

individualna programska oprema *(custom developed software)*
Programska oprema po naročilu, narejena posebej po željah in potrebah določenega uporabnika.

intelektualna lastnina *(intellectual property)*
Vrsta lastnine, ki izvira iz človekovega intelekta oziroma razuma. Ko je ta lastnina opredmetena v blagu in/ali storitvah, jo lahko imetnik pravice ali oseba, ki jo je ta pooblastil, komercialno izkoriščata.

inženirstvo domene *(domain engineering)*
Gradnja ponovno uporabnih komponent in izdelkov, metod, orodij in spremljajoče dokumentacije, namenjenih reševanju problemov med razvojem sistema oziroma podsistema z uporabo znanja o modelih domene in programskih arhitekturah.

izpeljano ali derivativno delo *(derivative work)*
Rezultat predelave nekega dela, ki postane novo samostojno delo.

J

javna dobrina *(public domain work)*
Delo, ki ga ni mogoče zaščititi v sistemu intelektualne lastnine ali mu je zaščita že potekla. Lahko se ga prosto uporablja, spreminja, razmnožuje, deli z drugimi ali komercialno izkorišča kopije ali različice dela, pridobljene na legitimen način.

javnodomensko delo *(public domain work)*
Glej javna dobrina.

K

katalog komponent *(component catalog)*
Seznam komponent, ki jih knjižnica ponuja ponovnim uporabnikom, opremljen s kratkim opisom vsebin in/ali delovanja komponente.

kategorija pravic intelektualne lastnine *(intellectual property rights category)*
Rezultat ocene pravic nad neko komponento.

klasificiranje *(classification)*
Glej razvrščanje.

knjižnica ponovno uporabnih komponent *(library of reusable components, reusable library)*
1. Podatkovna baza komponent, ki so bile načrtno razvite za ponovno uporabo v različnih kontekstih.
2. Vezni člen med razvojem ponovno uporabnih komponent (razvoj za ponovno uporabo) in razvojem programskih sistemov iz ponovno uporabnih komponent (razvoj s ponovno uporabo).
3. Skupek komponent, članov knjižnice, pravil in postopkov, knjižničnih mehanizmov ter vseh spremljajočih podpornih dejavnosti in storitev.

komercialni programi *(payware)*
Programi, ki se tržijo kot masovna programska oprema. Z nakupom pridobi kupec licenco za uporabo tega programa.

komponenta *(component, asset, artifact)*
Glej programska komponenta.

komponentna knjižnica *(component library)*
Glej knjižnica ponovno uporabnih komponent.

komponentna tehnologija *(component technology)*
Tehnologija, ki predvideva razvoj sistemov ali aplikacij na temelju ponovne uporabe, ne glede na izbrano metodologijo razvoja in/ali izbrani življenjski cikel.

komponentni model *(component model)*
Predstavlja vsebine komponent (elemente komponente) ter odvisnosti in povezave med njimi.

komponentno tržišče *(component marketplace, virtual component library)*

Medsebojna povezava dveh ali več komponentnih knjižnic, ki so dostopne prek skupne (izhodiščne) spletne strani, na kateri se predstavljajo s svojimi katalogi komponent.

kontekstna razdalja *(contextual distance)*
Razlike med kontekstom uporabe originalne komponente in kontekstom uporabe prilagojene komponente ali količino napora, ki ga zahteva pretvorba obstoječe komponente v želeno izvedbeno okolje (druga platforma in/ali programski jezik).

L

licenca *(licence,license)*
Glej licenčna pogodba.

licenčna določila *(licensing terms)*
Specificirane zahteve, povezane z licenciranjem.

licenčna pogodba *(licence agreement, royalty agreement)*
Način in obseg prenosa pravic do uporabe oziroma izkoriščanja programske opreme.

M

model domene *(domain model)*
Posplošitev sistemov domene in osnovo za postavitev komponentne knjižnice domene.

modeliranje *(modeling)*
Proces izdelave formalne abstrakcije nekega izdelka za analiziranje neke lastnosti. Ta lastnost se navadno nanaša na delovanje izdelka, njeno prisotnost pa je s preskušanjem ali analizo težko ugotoviti ali izmeriti.

O

obratno ali povratno inženirstvo *(reverse engineering)*
Ustvarjanje ekvivalenta originala izvorne kode in objektne kode programa oziroma poskušanje odkritja razvojnih podrobnosti programa.

P

paketna programska oprema *(off-the-shelf software)*
Standardna programska oprema, namenjena široki rabi, dosegljiva v računalniških prodajalnah ali na spletnih straneh proizvajalcev, narejena za nedoločeno število uporabnikov.

parametrizacija *(parametrization)*
Prilagajanje konkretnim zahtevam prek parametrov vmesnika.

patent *(patent)*
Je pravica industrijske lastnine, s katero se varuje izum (rešitev določenega problema na področju tehnologije).

piratstvo *(piracy)*
Način reprodukcije in distribucije ali razpečevanja računalniških programov, pri čemer so kršene avtorske in druge pravice, saj tretjim osebam omogoča uporabo nelegalnih kopij programa.

pogodba o hrambi računalniške kode *(escrow agreement)*
Pogodba o deponiranju izvorne kode ter pripadajoče tehnične in uporabniške dokumentacije pri tretji osebi (specializirano podjetje, notar, banka, zavarovalnica ipd.)

ponovna uporaba *(reuse)*
Pristop h gradnji oziroma sestavljanju programskih sistemov ali aplikacij iz ponovno uporabnih komponent, to je programskih izdelkov, ki so bili načrtno zgrajeni za večkratno ponovno uporabo v različnih kontekstih, pri čemer lahko te izdelke uporabimo nespremenjene ali pa jih prilagodimo konkretnemu kontekstu.

ponovno uporabna komponenta *(reusable component)*
1. Katerikoli programski izdelek ali produkt katerekoli razvojne faze, ki je bil izdelan ali zgrajen z namenom, da bo ponovno uporabljen tudi v kontekstih, drugačnih od tistega, v katerem je nastal. Zgrajen ali opremljen pa mora biti tako, da ga lahko poljubni ponovni uporabnik razume in uporabi brez avtorjeve pomoči.

2. Paket nekih operacij ali funkcij, ki ga je mogoče uporabiti pri gradnji večje komponente ali programskega sistema.

ponovno uporabni izdelek *(reusable component/artifact)*
Glej ponovno uporabna komponenta.

poslovna skrivnost *(trade secret)*
S poslovno skrivnostjo se varuje "know-how" v obliki izdelka, ideje, algoritma, tehnike, programskega orodja, komponente, metodologije, načrta, specifikacije ipd.

potrdilo *(certificate)*
Glej certifikat.

pregled *(inspection)*
Proces, ki ga navadno izvedejo ljudje, z njim pa preverijo prisotnost določenih lastnosti v izdelku, ki ga pregledujejo.

preskušanje *(testing)*
Eksperimentalni pristop, s katerim se dokazuje, da ima izdelek neko splošno lastnost. Lahko se izvede za vsako izvedljivo komponento. Gre torej za izvajanje komponente na primerih, torej pod določenimi pogoji, pri tem pa se rezultati izvajanj opazujejo in zapisujejo, hkrati pa se ocenjujejo določeni vidiki komponente.

prilagoditvena razdalja *(customization distance)*
Idejna razdalja med dejanskimi lastnostmi in funkcionalnostmi komponente ter želenimi oziroma zahtevanimi.

primerjalni preskus *(benchmarking)*
Proces preskušanja zmogljivosti komponent v času izvajanja, npr. hitrost dostopa do diska, uporabo spomina, numerično natančnost, odzivni čas na določen dogodek.

pripravljalno gradivo *(preparatory design material)*
Osnutek računalniškega programa, gradivo iz posameznih faz nastajanja računalniškega programa (diagrami, specifikacije, ipd.).

program v preizkusni uporabi *(shareware)*
Za določen čas brezplačen program, ki ga lahko uporabnik reproducira, predeluje in distribuira, po poteku tega časa pa se mora odločiti bodisi za njegov nakup (minimalna licenčnina) ali odstop od uporabe.

program v prosti uporabi *(freeware)*
Brezplačen program, ki ga lahko uporabnik reproducira, predeluje in nadalje distribuira.

programska arhitektura *(program architecture)*
Glej programsko ogrodje.

programska komponenta *(program component, software component)*
Katerikoli izdelek, vmesni ali končni, ki nastane med razvojem programskega sistema ali znotraj programskega procesa.

programska oprema *(software)*
Programski izdelek, ki nastane kot rezultat programskega procesa ali razvoja in je lahko sestavljen iz vnaprej pripravljenih programskih komponent, ki jih med seboj ustrezno povežemo v celoto.

programski proces *(program process, software process)*
Postopek, zaporedje korakov, katerega rezultat je programski izdelek ali programska komponenta.

programski sistem *(program system, software system)*
Glej programska oprema.

programsko ogrodje *(program framework)*
Opis vseh sestavnih delov programskega sistema, ki se da (pre)oblikovati, ter medsebojnih povezav med komponentami, ki bodo sistem sestavljale.

prozorna komponenta *(clear-box component, clear-box asset)*
Izvedbene lastnosti prozorne komponente so vidne samo prek določenih delov kode ali dokumentacije, kar omogoča boljše razumevanje delovanja komponente ter njeno učinkovitejšo ponovno uporabo, a jih ni mogoče spreminjati.

R

razčlenjenost *(component articulation)*
Označuje stopnjo popolnosti zagotavljanja (podajanja) rešitve problema posameznih delov ponovno uporabne komponente.

različica komponente *(component version)*
Izpeljanka iz neke ponovno uporabne komponente, pri kateri pa je bilo narejenih toliko sprememb, da ne moremo več reči, da je bila samo prilagojena trenutnim zahtevam in potrebam.

razvijalci ponovno uporabnih komponent *(reusable component developers)*
Tisti, ki sodelujejo na strani produkcije ponovno uporabnih komponent (inženirji domene, ki dobro poznajo potrebe komponentnega tržišča v domeni ter gradijo ponovno uporabne komponente).

razvijalec aplikacij *(aplication/system developer)*
Uporabnik ponovno uporabnih komponent - ponovni uporabnik oziroma razvijalec aplikacij ali programskih sistemov, imenovan tudi sestavljavec aplikacij, je strokovnjak, ki razvija aplikacije iz ponovno uporabnih komponent. Te lahko pridobi neposredno od razvijalcev oziroma dobaviteljev komponent, bolje pa je, če jih poišče v ustrezni domenski knjižnici (večja izbira, zagotovljena določena kakovost ipd.)

razvoj s ponovno uporabo *(development with reuse)*
Razvoj aplikacij ali programskih sistemov na temelju ponovne uporabe - aktivnosti, povezane z uporabo ponovno uporabnih komponent pri gradnji novih programskih sistemov ali aplikacij.

razvoj za ponovno uporabo *(development for reuse)*
Razvoj ponovno uporabnih komponent - aktivnosti ustvarjanja, pridobivanja in reinženiringa ponovno uporabnih komponent ali izdelkov, ki vključuje gradnjo modelov in arhitektur domene, novih ponovno uporabnih komponent ter reinženiring obstoječih komponent z namenom povečanja njihove ponovne uporabnosti.

razvrščanje *(classification)*
Urejanje komponent glede na njihovo vsebino, funkcionalnost, področje uporabe, dostopa in drugih kriterijev ali kombinacij kriterijev, ki jih postavi knjižnica v pravilih klasificiranja.

repozitorij ponovno uporabnih komponent *(repository of reusable components)*
Glej knjižnica ponovno uporabnih komponent.

S

sistematična ponovna uporaba *(systematic reuse)*
Ponovna uporaba, ki se izvaja znotraj dobro organiziranega in planiranega razvojnega procesa na temelju ponovne uporabe in omogoča doseganje prednosti, ki jih ponovna uporaba omogoča.

siva komponenta *(gray-box component, gray-box asset)*
Spreminjanje te komponente je omogočeno samo nad podmnožico elementov, ki to komponento sestavljajo oziroma tvorijo, in sicer prek parametrov komponente.

spremenljivost komponente *(component variability)*
Možnost izvedbe sprememb in prilagoditev nad komponento.

statična analiza *(static analysis)*
Avtomatiziran proces, s katerim naj bi pokazali, da ima izdelek neko posebno lastnost. Ocenjevanje komponente poteka na temelju njene oblike, strukture, vsebine ali dokumentacije (ne da bi prišlo do izvajanja izvorne kode).

strošek ponovne uporabe *(reuse cost)*
Seštevek stroška prilagajanja komponente ter stroška pridobivanja oziroma iskanja komponente.

pravna ureditev sui generis *(sui generis legal system)*
Svojevrstna pravna ureditev nekega področja.

T

testiranje *(testing)*
Glej preskušanje.

tržne komponente *(Open Market Components)*
Glej črna komponenta.

U

uporabniški vmesnik *(user interface)*
Določa način upravljanja s programom in predstavitve ter posredovanja podatkov na zaslonu.

upravitelj knjižnice *(library manager)*
Tisti, ki sodeluje pri organizaciji in delovanju knjižnice ponovno uporabnih komponent.

V

vidljivost komponente *(component visibility)*
Lastnost, ki pove, kako se izvedbene podrobnosti oziroma notranjost komponente odražajo navzven.

virtualna knjižnica *(virtual library)*
Glej komponentno tržišče.

vmesnik *(interface)*
Del programa, ki omogoča logično in po potrebi fizično medsebojno povezavo in interakcijo med posameznimi deli programske in/ali strojne opreme, tako da ti delujejo. Omogoča interoperabilnost med različnimi deli programske in strojne opreme.

Z

zahtevek za komponento *(request for proposal - RFP)*
Dokument, ki ga izdela pridobitelj komponent in v katerem so opisane vse zahteve, ki se nanašajo na ponovno uporabo komponent in jih mora dobavitelj komponent obvezno izpolniti.

zrnatost komponente *(component granularity)*
Lastnost, ki pove, koliko posebnih problemov lahko z neko komponento rešimo oziroma koliko alternativnih rešitev neka komponenta omogoča.

LITERATURA

Seznam A: Področje ponovne uporabe programskih komponent

1 *A Reuse Implementation Strategy (RIS) for the organization*. Army Reuse Center, 1993.

2 Allen P. "Using CBD to improve your business". *Component strategies*. SIGS Publications. July 1999, 2(1), str. 30-39.

3 Baldo J., Moore J., Rine D. "Software Reuse Standards". *StandardView* Vol. 5, No. 2. June 1997.

4 Bassett P.G. *Framing Software Reuse: Lessons From the Real World.* Yourdon Press Computing Series. Prentice Hall, 1997.

5 Buschmann F. "Finding & Using Patterns". *Object Expert*. Sept./Oct. 1996, str. 46-47,61.

6 Buschmann F. "What is a pattern?". *Object Expert*. March/April 1996, str. 17-18.

7 Conn R. *Software reuse - SE 508 Course*, *Version 2*. New Jersey. January 1996.

8 Crooks M. "Capitalizing on component reuse". *Component strategies*. SIGS Publications. July 1999, 2(1).

9 *DoD Software Reuse Vision and Strategy Document*. February 1996.

10 *Domain Engineering Methods and Tools Handbook, Volume I – Methods*. Comprehensive Approach to Reusable Defense Software (CARDS), STARS-VC-K017R1/001/00. December 1994.

11 Domajnko T. *Povečanje stopnje ponovne uporabe z uporabo vzorcev*. Doktorska disertacija. Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, 2001.

12 *DSSA Process Life Cycle*. GTE Govrnment Systems Corp., PD-081 DSSA-PG-001, Rev. 0.2. September 1992.

13 *Engineer's Handbook*. Central Archive for Reusable Defense Software (CARDS), VC-B002/002/00. Februar 1994.

14 Frakes W.B., Isoda S. "Success factors of systematic reuse". *IEEE Software 11*. 1994, str. 15-19.

15 Griss L.M. "Reuse 2001-2004: What next, now that we have solved all reuse problems?". The Foundation of Reuse, Proceedings of 9th WISR. Texas, 1999.

16 Griss L.M. "Software Reuse: Object and frameworks are not enough". *Object Magazine*, February 1995, str. 77-79.

17 Griss L.M. "The architecture and processes for a systematic OO reuse factory". Proceedings of 7th WISR (Workshop on institutionalizing software reuse). St. Charles, Illinois, 1995.

18 Guerrieri E. "Reuse success – when and how? ". The Foundation of Reuse. Proceedings of 9th WISR. Texas, 1999.

19 Jacobson, I., Griss M., Jonsson P. *Software reuse - Architecture, process and organization for business success*. ACM Press, New York. 1997.

20 Jakkola H. "The extensive role of reuse in software engineering ". V zborniku *Dnevi slovenske informatike 2002*. Portorož 2002, str. 1 – 16.

21 Johnson R.E. "An Introduction to Patterns". *ROAD*. May/June 1994, str. 41- 43.

22 Johnson R.E. "Patterns and Frameworks". *ROAD*. March/April 1995, str. 46 - 48.

23 Kachmarik M.B. "Commercial-off-the-shelf (COTS) products – an appropriate strategy for reuse in the development of object oriented systems ". Proceedings of 9th WISR, Austin, Texas, 1999.

24 Kain J.B. "Components: The Basics". *Object Magazine*, April 1996, str. 64-69.

25 Karlsson E.A. *Software Reuse: A holistic approach.* John Wiley & Sons Ltd.. Chichester, England, 1995.

26 Kralj A. *Komponentna gradnja informacijskih sistemov v okolju osebnih računalnikov*. Magistrsko delo. UL Ekonomska fakulteta. Junij 2000.

27 Lam W. "Risk management techniques for the transfer of reuse technology ". *Proceedings of 8th WISR (Workshop on institutionalizing software reuse)*. Columbus, Ohio, 1997.

28 Marshall S. et al. "Visualising reusable software over the web". V zborniku konference *Australian Symposium on Information Visualisation*. Australian Computer Society. Sydney, December 2001.

29 McGregor J.D., Sykes D.A. *Object-Oriented Software Development: Engineering Software for Reuse*. New York, 1992.

30 McInnis K. "Component-based design and reuse." Castek Software Factory. http://www.cbd-hq.com

31 *Model-Based Software Engineering* (Feature Oriented Domain Analysis – FODA). Software Engineering Institute. January 1997.

32 Moore J.W. "Fundamental principles of software reuse". Proceedings of 8th WISR. Columbus. Ohio, 1997.

33 Moore J.M. Bailin S.C. "Domain analysis: Framework for Reuse". *IEEE*. 1991.

34 Morisio M. Ezran M., Tully C. "Introducing reuse in companies: a survey of European experiences ". Proceedings of the 5th Symposium on software reusability (SSR). 1999.

35 *NATO Standard for the Development of Reusable Software Components*. NATO Communications and Information Systems Agency, Volume 1. 1996.

36 Neighbors, J.M. "An Assessment of Reuse Technology after Ten Years ". 3rd International Conference on Software Reusability. Rio de Janeiro, November, 1994.

37 Nielsen K. *Software Development with C++: maximizing reuse with object technology*. Academic Press Inc. Boston, 1995.

38 Oberndorf T. Brownsword L., Sledge C.A. "An activity framework for COTS-based systems". SEI – Software Ingeneering Institute. Pittsburg. October 2000

39 *Object Magazine*. June 1996.

40 Perry J.M. *Perspective on software reuse*. Technical report CMU/SEI-88-TR-22. Software Engineering Institute. Pittsburg, 1988.

41 Pinto P.E. "Promoting software reuse in a corporate setting". 1998. http://www.cs.cmu.edu

42 Poulin J.S. "The Foundation of Reuse ". Proceedings of 9th WISR. Texas, 1999.

43 Poulin J.S. Caruso J.M., Hancock D.R. "The business case for software reuse". *IBM Systems Journal 32, 4*. 1993, str. 567 - 594.

44 Prieto-Diaz R. "The disappearance of software reuse". ICSR (International Conference on Software Reuse)-3. Rio de Janeiro, Brazil, 1994, str. 225.

45 Reifer D.J. *Practical Software Reuse: Strategies for Introducing Reuse Concept in Your Organization*. John Wiley&Sons, Inc. 1997.

46 *Reuse Adoption Guidebook*, Software Productivity Consortium (SPC) Services Corporation. SPC-92051-CMC, Version 02.00.05. November 1993.

47 *Reuse-Driven Software Processes Guidebook*. Software Productivity Consortium, SPC-92019-CMC, Version 02.00.03. November 1993.

48 *Reusable Asset Specification (RAS)*. RFC Document#:ad/2003-10-12, Ver. 2.1. Oct. 2003.

49 Software Productivity Consortium. *Reuse Adoption Guidebook*. Technical Report SPC-92051-CMC, Version 02.00.05. November 1993.

50 *STARS Conceptual Framework for Reuse Processes (CFRP), Volume I: Definition*. Version 0.3, STARS-VC-A018/001/00. October 1993.

51 Steyaert P., Lucas C., Mens K. "Reuse contracts: Making systematic reuse a standard practise". Proceedings of 8th WISR. Columbus, Ohio, 1997.

52 Swaminathan V., Storey J. "Domain-Specific Frameworks". *Object Magazine*. April 97, str. 53-57.

53 *The Reuse-Oriented Software Evolution (ROSE) Process Model, Version 0.5 – DRAFT*. STARS-UC-05155/001/0, July 1993.

54 Tracz W. *Confessions of a Used Program Salesman: Institutionalizing Software Reuse*. Addison-Wesley, 1995.

55 Vatovec Krmac E. "Ponovna uporaba – kaj, zakaj, kdaj in kako". V zborniku posvetovanja *Dnevi slovenske informatike 1997*. Portorož, 1997, str. 24-34.

56 Vatovec Krmac E. *Ponovna uporaba komponent pri objektno usmerjenem razvoju programske opreme*. Magistrska naloga. UL Fakulteta za računalništvo in informatiko, Ljubljana, 1997.

57 Vatovec Krmac E. "Vključitev ponovne uporabe v objektno usmerjen razvojni proces programske opreme". Zbornik *Objektna tehnologija v Sloveniji 1997*. Maribor, 1997, str. 34-46.

58 Vatovec Krmac E. "Vloga ponovne uporabe pri razvoju programske opreme ". *Uporabna informatika, št. 2, letnik 5.* Ljubljana, apr/maj/jun 1997, str. 28-35.

59 Vatovec Krmac E. "What must be included in a reusable component?". V zborniku *Proceedings of the 4th international multi-conference*. Ljubljana, 2001, str. 369-374.

60 Wu Z.L. "Software Reuse and World Wide Web". www.cs.uwindsor.ca/users/w/wu5/report/510.htm

Seznam B: Področje digitalnih knjižnic

61 Browne S.V., Moore J.W. "Reuse Library Interoperability and the World Wide Web". *Communication of the ACM*, 1997.

62 IEEE Standard 1420.1: *Data Model for Reuse Library Inteoperability - Basic Interoperability Data Model (BIDM)*. 1995.

63 IEEE Standard 1420.1a: *Asset Certification Framework*. 1996 in 2002.

64 IEEE Standard 1420.1b: *Intellectual Property Rights Framework*. 1999 in 2002.

65 Samuelson P. "Encoding the law into digital libraries". *Communication of the ACM*. Vol. 41, No. 4. April 1998, str. 13 - 18.

Seznam C: Pravno varstvo programske opreme, intelektualna lastnina

66 Alič T. *Licenčna pogodba za programsko opremo*. Diplomsko delo. Univerza v Ljubljani, Pravna fakulteta. 2003.

67 Bogataj M. "Avtorsko pravo v dobi novih tehnologij". GV Pravna praksa, št. 36, 2000, str. 29.

http://www.ius-software.si

68 *Communications on copyright and related rights in the information society*. http://europa.eu.int/en/record/other/isc2en.htm

69 *Digital Signature Guidelines*. Legal Infrastructure for Certification Autorities and Secure Electronic Commerce. August, 1996.

70 Guibault L. "Discussion paper on the question of Exceptions to and limitations on copyright and neighouring rights in the digital era." *Council of Europe*. Strasbourg, October 1998.

71 Gervais D.J. "Copyright, money & internet". V Zborniku konference *Copyright office comes to California*. San Francisco, March 2004.

72 Gervais D.J. "The internazionalization of intellectual property: new challenges from the very old and the very new". *Fordham Intellectual Property, Media and Entertainment Law Journal*, 2002.

73 Graham P.S. "Intellectual preservation and electronic intellectual property". V zborniku *Proceedings of Technological strategies for protecting intellectual property in the networked multimedia environment*. Interactive Multimedia Asociation, 1994.

74 Heide T. "Access control and innovation under the emerging EU electronic commerce framework". 2000. http://www.law.berkeley.edu/journals/btlj/articles/vol15/heide/heide.html

75 *Intellectual property in the age of universal access*. Edited by Crawford D. ACM Sigsoft. 1999.

76 *Integrating Intellectual Property Rights and Development Policy.* Commision on intellectual property rights. London, September 2002.

77 *Joint recommandation concerning provisions on the protection of marks and other industrial property rights in signs on the internet (with explanatory notes).* WIPO Publ., 2001.

78 Lemley M., Jorde T.M., Menell P., Merges R.P. *Software and Internet Law.* Aspen Publishers Inc. New York, 2003.

79 Lemley M., Menell P., Merges R.P., Samuelson P. *Intellectual Property in the new technological age: selected statutes and cases.* Aspen Publishers Inc. New York, 1999.

80 Makarovič B. et.al. *Internet in pravo.* Ljubljana, UL Pravna fakultetea, 2003.

81 Marsland V. *European Computer Law: An Introductory Guide*. The Computer Law Assocciation. 1996.

82 McManis C.R. "Intellectual property protection and reverse engineering of computer programs in the United States and the European Community". http://www.law.berkeley.edu/journals/btlj/articles/vol8/McManis/html/text.html

83 Oman B. "Avtorska pravica". *GV, Podjetje in delo, št. 4*. Ljubljana, 1996, str. 448. http://www.ius-software.si

84 Pagenberg J., Geissler B. *License agreements, 3rd edition: patents, utility models, know-how, computer software*. Germany, 1991.

85 Pavliha M. et. al. *Zakon o elektronskem poslovanju in elektronskem podpisu (ZEPEP): s komentarjem.* Ljubljana, GV založba , 2002.

86 Pretnar B. *Intelektualna lastnina v sodobni konkurenci in poslovanju*. GV Založba. Ljubljana, 2002.

87 Puharič K. "Nova pravna vprašanja intelektualne lastnine v odprtih elektronskih okoljih". *GV Podjetje in delo*, št. 6. 1997, str. 1063. http://www.ius-software.si

88 Puharič K. "Pravno varstvo računalniških programov ". *GV Podjetje in delo*, št. 5. 1996. http://www.ius-software.si

89 Remer D. *Legal care for your software.* Aldershort, England, 1984.

90 *Report from the commission to the council, the european parliament and the economic and social committee on the implementation and effects of Directive 91/250/EEC on the legal protection of computer*

programs. Brussels, 2000.

91 Sajovic B. *Osnove avtorskega prava*. Ljubljana, UL Pravna fakulteta, 1974.

92 Samuelson P. "DRM {AND, OR, VS.} The Law". *Communications of the ACM*. Vol. 46, No. 4. April 2003, str. 41 - 45.

93 Samuelson P. "Embedding technical self-help in licensed software". *Communications of the ACM*. Vol. 40, No. 10. October 1997, str. 13 - 17.

94 Samuelson P. "Intellectual property and the digital economy: Why the anticircumvention regulations need to be revised". 1999. http://www.sims.berkeley.edu/~pam/papers/Samuelson_IP_dig_ecohtm.htm

95 Samuleson P. "Intellectual property for an information age: How to balance the public interest, traditional legal principles, and the emerging digital reality". *Communcations of the ACM*. Vol. 44, No. 2, February 2001, str. 67 - 68.

96 Samuelson P. "Legal protection for databases contents". *Communication of the ACM*. Vol. 39, No. 12. December 1996, str. 17 - 23.

97 Samuelson P. "Toward a new politics of intellectual property". *Communication of the ACM*. Vol. 44, No. 3, March 2001, str. 98 - 99.

98 Samuelson P., Davis R. "The digital dilemma: a perspective on intellectual property in the information age ". *Telecommunications policy research conference*. 2000.

99 Samuelson P., Deasy K., Martin A.C. *Proposal for a new »Right in software« clause for software acquisition by the Department of defense*. Technical report CM WSEI-86-2. Pittsburgh, September 1986.

100 Samuelson, P. and Deasy, K. *Intellectual Property Protection for Software*. SEI Curriculum Module SEI-CM-14-2.1. Carnegie Mellon University. July 1989.

101 Strowel B., Ide N. "What really happened in Geneva: An inside look at the WIPO Treaties ". *World computer law congress and Computer and telecommunications law update*. 1997, Washington.

102 Trampuž M. "Direktiva Sveta Evropskih skupnosti z dne 14. maja 1991 o pravnem varstvu računalniških programov". *GV, Pravna praksa*. Ljubljana, 1992, str. 16. www.ius-software.si

103 Trampuž M. "Predmet avtorskopravnega varstva pri računalniških programih". *GV Podjetje in delo*, *št. 5*. 1996, str. 1018. www.ius-software.si

104 Trampuž M., Oman B., Zupančič A. *Zakon o avtorski in sorodnih pravicah s komentarjem*. Gospodarski vestnik. Ljubljana 1997.

105 Vatovec Krmac E. "Razvoj programske opreme na osnovi ponovne uporabe in problematika pravic intelektualne lastnine ". V zborniku *Telekomunikacije 0I telecommunications*, Nova Gorica, 2001.

106 Westby J.R. *International Guide to Combating Cybercrime*. American Bar Association. 2003

107 *WIPO Intellectual property handbook: policy, law and use*. WIPO publ. No 489(E). Geneva, 2001.

Seznam D: Pravni viri

108 *Bernska konvencija za varstvo književnih in umetniških del* (Berne convention for the protection of literary and artistic works). Uradni list SFRJ-MP, št. 14/75. Uradni list RS-MP, št. 9/92.

109 *Direktiva sveta evropskih skupnosti o pravnem varstvu računalniških programov* (Council directive 91/250/EEC of 14 May 1991 on the legal protection of computer programs). Official Journal of the European Communities L 122, 17.5.1991, str. 42.

110 *Directive 2001/29/EC of the European parliament and of the Council of 22 May 2001 on the harmonisation of certain aspects of copyright and related rights in the information society*. Official Journal of the European Communities, L 167, 22.6.2001, str. 10.

111 *Council directive 92/100/EEC of 19 November 1992 on rental right and lending right and certain rights related to copyright in the field of intellectual property*. Official Journal of the European Communities L 346, 27.11.1992, str. 61.

112 *Council directive 2004/48/EC of the European Parliament and of the Council of 29 April 2004 on on the enforcement of intellectual property rights.* Official Journal of the European Union, L 157, 30.4.2004, str. 45.

113 *Council directive 93/98/EEC of 29 October 1993 on the harmonising the term of protection of copyright and certain related rights*. Official Journal of the European Communities L 290, 24.11.1993, str. 9.

114 *Directive 96/9/EC of the European Parliament and of the Council of 11 March 1996 on the legal protection of databases*. Official Journal of the European Communities L 77, 27.3.1996, str. 20.

115 *Konvencija o ustanovitvi Svetovne organizacije za intelektualno lastnino.* Uradni list SFRJ-MP, št. 31/72 in 4/86, Uradni list RS-MP, št. 9/92, 3/01.

116 *Obligacijski zakonik.* Uradni list RS, št. 83-4287/2001.

117 *Pariška konvencija za varstvo industrijske lastnine.* Uradni list SFRJ-MP, št. 5/74, 7/86, Uradni list RS-MP, št. 9/92.

118 *Pogodba o patentnem pravu.* Uradni list RS-MP, št. 4/02.

119 *Pogodba Svetovne organizacije za intelektualno lastnino o avtorski pravici.* Uradni list RS-MP, št. 25/99.

120 *Pravilnik o izvajanju 6. in 10. člena zakona o carinskih ukrepih pri kršitvah pravic intelektualne lastnine.* Uradni list RS, št. 76/01.

121 *Pravilnik o vsebini patentne prijave in postopku z deljenimi patenti.* Uradni list RS, št. 102/01.

122 *Predlog zakona o spremembah in dopolnitvah zakona o avtorski in sorodnih pravicah (ZASP-B).* Poročevalec DZ RS, št. 20. Ljubljana, 28.2.2004, str. 29.

123 *Predlog zakona o spremembah in dopolnitvah zakona o elektronskem poslovanju in elektronskem podpisu (ZEPEP-A).* Poročevalec DZRS, št.19. Ljubljana, 24.2.2004, str. 3.

124 *Report from the Commision to the Council, the European Parliament and the Economic and Social Committee on the implementation and effects of Directive 91/250/EEC on the legal protection of computer programs.* COM (2000) 199 final. Brussels, 10.4.2000.

125 *Sporazum o trgovinskih vidikih pravic intelektualne lastnine*. Uradni list RS-MP, št. 10/95.

126 *Strasbourški sporazum o mednarodni klasifikaciji patentov.* Uradni list RS-MP, št. 7/01.

127 *Svetovna konvencija o avtorski pravici.* Uradni list SFRJ-MP, št. 54/73, Uradni list RS-MP, št. 15/92.

128 *Uredba o cenah informacijskih in drugih storitev Urada RS za intelektualno lastnino.* Uradni list RS, št. 57/96.

129 *Uredba o pristojbinah za pridobitev in vzdrževanje pravic industrijske lastnine.* Uradni list RS, št. 110/01.

130 *Uredba o ratifikaciji Sporazuma o sodelovanju na področju patentov med Vlado Republike Slovenije in Evropsko patentno organizacijo.* Uradni list RS-MP, št. 15/93.

131 *Uredba o razširitvi evropskih patentov na Republiko Slovenijo.* Uradni list RS, št. 15/02.

132 *WIPO Copyright Treaty (WCT). Geneva, 1996.* Official Journal of the European Communities L 89/8. April 2000.

133 *Zakon o avtorski in sorodnih pravicah.* Uradni list RS, št. 21/95, 9/01, 30/01, 43/04.

134 *Zakon o elektronskem poslovanju in elektronskem podpisu (ZEPEP).* Uradni list RS, št. 57/2000, 73/2004.

135 *Zakon o gospodarskih družbah.* Uradni list RS, št. 30/93.

136 *Zakon o industrijski lastnini.* Uradni list RS, št. 45/01 in št. 13/92.

137 *Zakon o pravicah industrijske lastnine iz delovnega razmerja.* Uradni list RS, št. 45/95.

138 *Zakon o ratifikaciji pogodbe Svetovne organizacije za intelektualno lastnino o avtorski pravici (MSOILAP).* Uradni list RS, št. 25/99.

139 *Zakon o varstvu konkurence.* Uradni list RS, št. 18/93.

140 *Zakon o varstvu topografije.* Uradni list RS, št. 21/95.

Seznam F: Drugi viri

141 *Component Software Glossary.* Object Services and Consulting, Inc., 1996. *http://www.objs.com/survey/ComponentwareGlossary.htm*

142 *Digital signature guidelines tutorial. http://www.abanet.org/scitech/ec/isc/dsg-tutorial.html.*

143 *GNU General Public License.* http://www.gnu.org/copyleft/gpl.html

144 IEEE/EIA Standard 1074: *IEEE Standard for developing software life cycle processes*. 1997.

145 ISO/IEC Standard 12207: *Information technology - Software life cycle process*. ISO/IEC, Geneve, 1995.

146 ISO/IEC Standard 12207: *Information technology - Software life cycle process*, *Amendment 1*. ISO/IEC, Geneve, 2000.

147 *Leksikon računalništva in informatike*. Pasadena. Ljubljana, 2002.

148 *Slovar slovenskega knjižnega jezika*. Državna založba Slovenije. Ljubljana, 2002.

Seznam F: Pomembnejši spletni naslovi

Activities of the European Union Information Society: http://europa.eu.int/pol/infso

http://www.gnu.org

http://www.ius-info.si

http://www.licensingmodels.com

http://www.componentsource.com

http://www.omg.org

The Apache Software Foundation: http://www.apache.org

Castek Software Factory: http://www.cbd-hq.com

Flashline: http://www.flashline.com

Netlib Repository: http://www.netlib.org

IZJAVA

Podpisana Evelin Vatovec Krmac izjavljam, da sem doktorsko nalogo izdelala samostojno, pod vodstvom mentorja doc. dr. Tomaža Mohoriča in somentorja prof. dr. Marka Pavlihe.

Portorož, 3. februar 2005 *Evelin Vatovec Krmac*

ZAHVALA

... vsem mojim najdražjim. Čisto vsi so mi tako ali drugače stali ob strani. Hote ali nehote. Pomoč, ki so mi jo izkazali na tisoče različnih načinov, sem sprejela kot izraz njihove globoke ljubezni in spoštovanja. Upam, da jim ju bom znala in zmogla vrniti.

Posebej prisrčno se zahvaljujem tudi mentorju in somentorju za njuno strokovno in moralno pomoč.