УНИВЕРЗИТЕТ У БЕОГРАДУ

ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ

Марија Пунт

# ИНТЕРАКЦИЈА ЧОВЕК-РАЧУНАР У ИНТЕГРИСАНОМ ОКРУЖЕЊУ ДИГИТАЛНИХ ТВ ПРИЈЕМНИКА, МОБИЛНИХ УРЕЂАЈА И ИНТЕРНЕТА

докторска дисертација

Београд, 2015

UNIVERSITY OF BELGRADE

SCHOOL OF ELECTRICAL ENGINEERING

Marija Punt

# HUMAN-COMPUTER INTERACTION IN AN INTEGRATED ENVIRONMENT COMBINING DIGITAL TV, MOBILE DEVICES AND INTERNET

Doctoral Dissertation

Belgrade, 2015

**Ментор:**

др Јован Ђорђевић, редовни професор

Универзитет у Београду, Електротехнички факултет

**Чланови комисије:**

др Бошко Николић, ванредни професор

Универзитет у Београду, Електротехнички факултет

др Мирослав Дукић, редовни професор

Универзитет Сингидунум, Електротехника и рачунарство

др Мило Томашевић, ванредни професор

Универзитет у Београду, Електротехнички факултет

др Никола Теслић, редовни професор

Универзитет у Новом Саду, Факултет техничких наука

**Датум одбране:**

**Изјава захвалности**

Захваљујем се драгим људима који су ми несебично пружили значајну подршку током истраживања и рада на овој докторској дисертацији. Желела бих да се захвалим свом ментору професору др Јовану Ђорђевићу на дугогодишњој сарадњи, саветима и смерницама, професору др Николи Теслићу, колеги др Милану Бјелици и професору Владану Здравковићу на сарадњи и неизмерној помоћи приликом израде ове дисертације. Такође, захваљујем се и студентима Јовановић Стефану, Јеличић Ани, Родаљевић Бранку, Владичић Дијани, Вељковић Николи, Тршић Лазару, Јелисавчић Милану и Томић Михаилу на учешћу у истраживању. Захвалност дугујем и мојим колегама са катедре који ми увек пружају инспиративну и забавну радну атмосферу.

Посебну захвалност дугујем мом супругу који је увек био уз мене, имао времена да прочита моје радове на енглеском језику и пружи ми драгоцене савете. Захвална сам ћерци Ади која ме непрестано мотивише и држи моју радозналост константно пробуђеном, сину Леону који је у току израде ове дисертације био дивна беба која воли пуно да спава, остављајући мами време за рад. Захваљујем се мом оцу Милораду који ми је од малена усадио љубав према логичком начину размишљања и раду са студентима, мајци Зорки на неизмерној љубави и подршци приликом свих одлука које сам у животу доносила, мом старијем брату Мирку који ми је био узор и подршка и сестри од тетке Снежани која ми је неизмерно помогла у бризи око деце пружајући ми простор за рад и истраживање.

**Наслов докторске дисертације**: Интеракција човек-рачунар у интегрисаном окружењу дигиталних ТВ пријемника, мобилних уређаја и интернета

**Резиме:** Број мултимедијалних уређаја који су доступни у дневној соби и имају приступ интернету, као што су дигитални ТВ пријемници, таблет уређаји и мобилни телефони, је у константном порасту. Ове уређаје је на једноставан начин могуће повезати путем локалне кућне мреже или интернета. Такође, у последње време, дигитални ТВ пријемници се све чешће опремају и оперативним системима који се могу наћи на мобилним уређајима што омогућава и лак развој дистрибуираних апликација. Апликације развијене у овако интегрисаном окружењу дигиталних ТВ пријемника, мобилних уређаја и интернета могу корисницима да пруже нове сценарије интеракције човек-рачунар у односу на оне које пружају апликације развијене независно за сваки уређај. Проучавањем система у области интеракције човек-рачунар у окружењу дневне собе са дигиталним ТВ пријемницима и мобилним уређајима уочено је да се постојећи системи развијају независно, да су ограничене флексибилности и да им је фокус углавном на појединачним сценаријима интеракције човек-рачунар.

У овом раду представљена је SHARP софтверска платформа која пружа програмску подршку за ефикасан и униформан развој дистрибуираних апликација у интегрисаном окружењу дигиталних ТВ пријемника, мобилних уређаја и интернета. Циљ развијене платформе је да помогне истраживачима у области интеракције човек-рачунар да на једноставан и брз начин креирају нове апликације које нуде различите комбинације сценарија интеракције човек-рачунар у интегрисаном окружењу. Како би се постигао наведени циљ на почетку рада је направљен преглед постојећих система у области интеракције човек-рачунар у дневној соби са дигиталним ТВ пријемником и мобилним уређајима. Након тога, пронађене су заједничке карактеристике система и идентификовани су следећи сценарији интеракције које апликације развијене у интегрисаном окружењу могу да пруже корисницима: подела приказа на више екрана (на мобилним уређајима се приказује додатни садржај као пратећи ономе што је на великом ТВ екрану), употреба мобилног уређаја за управљање акцијама на централном екрану, детекција покрета коришћењем сензора, додатне чулне повратне информације

(хаптички одговор, светлосно обавештење), пасивно гледање телевизије обогаћено паралелним интерактивним активностима, корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака и интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама. Након идентификације сценарија интеракције човек-рачунар у интегрисаном окружењу предложена је, пројектована и имплементирана SHARP софтверска платформа тако да омогући развој апликација које корисницима нуде идентификоване сценарије интеракције. Како би се верификовале погодности развијене платформе одабрано је да се због велике популарности игара и њихове тржишне исплативости платформа искористи за развој друштвених игара које би могле да се играју у интегрисаном окружењу. Коришћењем платформе развијено је пет различитих игара које показују флексибилност платформе и представљају пример нових типова апликација које корисницима нуде различите комбинације идентификованих сценарија интеракције човек-рачунар. Испитивањем искуства 59 корисника донети су закључци о томе како корисници прихватају идентификоване сценарије интеракције. Верификација SHARP софтверске платформе је извршена поређењем величине, сложености и времена одзива апликација које су развијене коришћењем и без коришћења платформе. Резултати поређења су показали да је софтверска платформа утицала да се време потребно за развој и тестирање апликација смањи. Времена одзива прикупљена из апликација развијених без коришћења и коришћењем софтверске платформе су показала да је апликација развијена коришћењем платформе унела прихватљиво додатно кашњење. Такође, времена одзива у свим верзијама апликација су била у дозвољеним границама перформанси утврђених прагова.

**Кључне речи:** интеракција човек-рачунар, интерактивна телевизија, софтверска платформа, дигитални ТВ пријемници, мобилни уређаји, други екран, ТВ центричне игре

**Научна област:** Електротехника и рачунарство

**Ужа научна област:** Рачунарска техника и информатика

**УДК број:** 621.3

**Title**: Human-computer interaction in an integrated environment combining digital TV, mobile devices and internet

**Abstract**: The amount of digital multimedia devices in a modern day household capable of connecting to the internet has dramatically increased over the last years, including mobile devices such as smart phones and tablets as well as digital TV sets and set-top boxes. Since these devices are readily available and allow customization through software, development of distributed applications is greatly simplified. Applications developed in an integrated environment can offer users novel human-computer interaction scenarios compared to the scenarios that a single device could offer. By investigating existing systems in the field of Human-Computer Interaction (HCI) that combine digital TV and mobile devices in a living room environment it was observed that existing systems have ad-hoc implementations with limited flexibility and lack the ability to explore different combinations of HCI scenarios.

This dissertation presents the SHARP development framework allowing efficient implementation of distributed systems in an integrated environment consisting of digital TV, mobile devices and internet. The purpose of the framework is to assist researchers in the field of HCI to create innovative applications that would offer different combinations of HCI scenarios in the living room with less effort. At the start of the dissertation the results of a survey providing an overview of existing HCI systems targeting a living room environment are presented. Common characteristics of these systems are extracted and a list of the following HCI scenarios that applications, developed for an integrated environment, can offer are identified: sharing content on multiple screens (private content showing on the mobile device screen and public content showing on the central screen), using the mobile device to control actions on the central screen, motion detection using sensors, supplemental sensory feedback (haptic feedback, background lighting), combining passive viewing of television with parallel interaction, enriching the user experience with access to various information services (such as DVB, social media and metadata databases), combining interaction between co-located and non-co-located people. After identifying the scenarios a proposal for the SHARP development framework is outlined along with details on its design and implementation. The design of the SHARP framework allows developers to create

applications that combine the identified scenarios. As the target domain to verify the benefits of the framework the field of game development was chosen. Game applications have proven to be in demand and profitable. Five different games were developed using the SHARP framework to show the flexibility of the framework and to demonstrate novel applications that offer new combinations of scenarios to the users. Conclusions on how the users accepted the identified scenarios were made by examining the experience of 59 test users interacting with the developed game applications. The SHARP framework was verified by comparing size, complexity and responsiveness of the applications developed using the framework and without using the framework. The results show that the framework reduced both development and testing effort by moving complex logic required in a game implementation to re-usable components within the framework. The responsiveness measurements collected from the applications developed with and without the framework show that although the usage of the framework introduced an overhead causing a slightly larger response time, the response times for all versions were well within acceptable performance limits.

**Keywords**: human-computer interaction, interactive TV, development framework, mobile devices, second screen, TV centric gaming

**Scientific area**: Electrical and Computer Engineering

**Scientific subarea**: Computer Engineering and Information Theory

**UDC number**: 621.3

# САДРЖАЈ

# 1. УВОД

Број дигиталних мултимедијалних уређаја који се користе у савременом домаћинству се драматично повећао током последњих година, тако да су у типичној дневној соби корисницима на располагању дигитални ТВ пријемници, таблет уређаји и мобилни телефони. Дигитални ТВ пријемници су повезани на велике ТВ екране и имају могућност пријема и декодовања емитованог дигиталног сигнала. Такође имају и приступ интернету и за разлику од старијих телевизора се не употребљавају само за пасивно гледање емитованог ТВ програма, него се све чешће користе и за интерактивну забаву. За управљање садржајем на ТВ екранима користе се даљински управљачи који су намењени пасивној употреби и често нису практични приликом употребе интерактивних апликација, такође дозвољавају само једној особи да има контролу над телевизијским садржајем. Упоредо са гледањем телевизије корисници све чешће у дневној соби користе мобилне уређаје (таблет уређаје  и мобилне телефоне) како би проверили електронску пошту, разменили текстуалне поруке, претраживали интернет или приступали друштвеним мрежама. Мобилни уређаји немају велики екран, али имају јединствене особине као што су екран осетљив на додир и сензоре покрета који пружају нове методе интеракције.

Тржиште дигиталних ТВ пријемника и мобилних уређаја је јако профитабилно. Произвођачи опремају уређаје различитим атрактивним апликацијама како би обезбедили што већу продају. Међу најпопуларнијим категоријама софтвера који се развија за дигиталне ТВ пријемнике и мобилне уређаје су и игре. Игре се генерално користе у великом броју домаћинстава у технолошко развијеним земљама (70% у САД) и тржишно су исплативе. Дигиталне ТВ пријемнике и мобилне уређаје је на једноставан начин могуће повезати путем локалне кућне мреже или интернета. Такође, у последње време, дигитални ТВ пријемници се све чешће опремају и оперативним системима који се могу наћи на мобилним уређајима што омогућава и лак развој дистрибуираних апликација. Апликације развијене у овако интегрисаном окружењу дигиталних ТВ пријемника, мобилних уређаја и интернета могу корисницима да пруже нове

сценарије интеракције човек-рачунар у односу на оне које пружају апликације развијене независно за сваки уређај.

Проучавањем система у области интеракције човек-рачунар у окружењу дневне собе са дигиталним ТВ пријемницима и мобилним уређајима уочено је да се постојећи системи развијају независно, да су ограничене флексибилности и да им је фокус углавном на појединачним сценаријима интеракције човек-рачунар. Циљ истраживања које је приказано у овој докторској дисертацији је развој и имплементација софтверске платформе, која ће пружити програмску подршку за ефикасан и униформан развој дистрибуираних апликација у интегрисаном окружењу дигиталних ТВ пријемника, мобилних уређаја и интернета. Софтверска платформа би требало да помогне истраживачима у области интеракције човек-рачунар да на једноставан и брз начин креирају нове апликације које нуде различите комбинације сценарија интеракције човек-рачунар у интегрисаном окружењу. На овај начин би се омогућило да се на ефикасан начин испита како корисници прихватају нове типове апликација и комбинације сценарија интеракције које нуде. Како би се постигао циљ било је неопходно да се пронађу заједничке карактеристике постојећих система, да се идентификују могући сценарији интеракције човек-рачунар, да се изврши критичка анализа алата и на основу добијених резултата да се осмисли, пројектује и имплементира софтверска платформа.

Како би се верификовале погодности развијене платформе одабрано је да се због велике популарности игара и њихове тржишне исплативости платформа искористи за развој друштвених игара које би могле да се играју у интегрисаном окружењу. Игре развијене коришћењем имплементиране софтверске платформе би требало да покажу флексибилност платформе и могућност да се коришћењем платформе развију нови типови апликација који нуде различите комбинације идентификованих сценарија интеракције човек-рачунар у интегрисаном окружењу. Предложену софтверску платформу је потребно верификовати и поређењем величине, сложености и времена одзива апликација које су развијене коришћењем и без коришћења предложене платформе. Рад је представљен у главама које следе.

У глави два приказани су постојећи системи који пружају различите видове интеракције човек-рачунар у оквиру дневне собе са дигиталним ТВ пријемницима и мобилним уређајима. Након тога су представљени и начини интеракције човек-рачунар у играма, са посебним акцентом на игре са више играча које се играју употребом мобилних уређаја и уређаја који поседују велике екране. На крају су представљена ограничења постојећих система, идентификовани су сценарији интеракције човек-рачунар у интегрисаном окружењу дигиталних ТВ пријемника, мобилних уређаја и интернета и дат је предлог SHARP софтверске платформе која треба да пружи програмску подршку приликом развоја апликација у интегрисаном окружењу.

У глави три детаљно је описана реализација предложене SHARP софтверске платформе. Прво је приказана структура платформе и алати који су изабрани за реализацију. Након тога, реализација сваке компоненте софтверске платформе је детаљно описана. На крају је приказан начин размене информација између дистрибуираних компонената SHARP софтверске платформе.

У глави четири описано је пет различитих игара развијених употребом SHARP софтверске платформе. За сваку игру приказан је начин дистрибуције компонената платформе употребом различитих апликација у оквиру интегрисаног окружења. Такође, за сваку од игара представљена је и употреба различитих комбинација идентификованих сценарија интеракције. На крају су приказани резултати испитивања корисника приликом употребе развијених игара.

У глави пет су представљени резултати верификације SHARP софтверске платформе поређењем величине, сложености, густине сложености и времена одзива апликација развијених без и са коришћењем SHARP софтверске платформе. Најпре су приказани резултати поређења величине и комплексности целокупног програмског кода апликација развијених са и без употребе платформе. Након тога су представљени резултати поређења величине, сложеност и густине сложености програмског кода, категоризованог по свакој од коришћених компонената софтверске платформе. На крају су представљени резултати поређења времена одзива апликација развијених без и са коришћењем SHARP софтверске платформе.

У глави шест која представља закључак изложени су резултати рада, доприноси рада и правци могућег даљег истраживања. На крају је дат списак коришћене литературе.

# 2. ДЕФИНИЦИЈА ПРОБЛЕМА И ПРЕДЛОГ РЕШЕЊА

У оквиру ове главе прво ће бити представљени постојећи системи који пружају различите видове интеракције човек-рачунар у дневној соби са дигиталним ТВ пријемником и мобилним уређајима. Након тога, биће приказани различити начини интеракције човек-рачунара у играма, са посебним акцентом на игре са више играча које се играју употребом мобилних уређаја и уређаја који поседују велике екране. На крају ће бити представљена ограничења постојећих система, идентификовани сценарији интеракције човек-рачунар и дат предлог софтверске платформе која ће омогућити униформан и брз развој апликација у интегрисаном окружењу. Такође, детаљно ће се представити и разлози због којих је одабрано да се развијена софтверска платформа употреби за развој игара у интегрисаном окружењу.

## 2.1 Интеракција човек-рачунар у дневној соби са дигиталним ТВ пријемником и мобилним уређајима

Постојећи системи у оквиру дневне собе могу да се поделе на системе социјалне и интерактивне телевизије, системе који користе додатни екран и системе код којих се мобилни уређаји користе за препознавање покрета. Социјална и интерактивна телевизија представља системе у којима је испитана интеракција и комуникација човека са другим гледаоцима истог ТВ програма. Како углавном сви корисници ТВ уређаја поседују и мобилне уређаје у оквиру дневне собе развијају се и системи који укључују екране мобилних уређаја за приказ додатних информација. Такође, мобилни уређају се могу искористити и за управљање садржајем на великом екрану.

### 2.1.1 Социјална и интерактивна телевизија

Гледање телевизије је некада било заједничко искуство које је подразумевало окупљање породице и пријатеља у дневној соби како би гледали и коментарисали исти телевизијски програм [15]. Људи би конзумирали оно сто им је представљено на ТВ-у, а интерактивност са ТВ програмом би се огледала у менталном такмичењу са учесником ТВ квиза. У новије време људи све чешће гледају телевизију сами и асинхроно имајући на уму да домаћинства углавном поседују више од једног ТВ апарата, велики избор канала, могућност снимања телевизијског програма и гледања видеа на захтев. Развој технологије је омогућио везу дигиталних ТВ пријемника са интернетом и отворио могућности за нове видове интеракције гледаоца са ТВ програмом или пријатељима који нису у истој соби. У литератури су предложени бројни прототипови социјалних и интерактивних ТВ система који подржавају различите видове комуникације и интеракције људи у току гледања ТВ програма. Неки од најзначајнијих су: *AmigoTV* [15], *ConnectTV* [13], *CollaboraTV* [52] и *Motorola STV* [51].

Један од првих прототипова социјалне телевизије који омогућава комуникацију удаљених пријатеља у реалном времену у току гледања емитованог ТВ програма је *AmigoTV*. Комуникација је омогућена између одабране групе пријатеља који гледају исти телевизијски канал у датом тренутку. На тај начин се креира концепт виртуелне собе која припада одређеном каналу. Корисницима се као примарни вид комуникације нуди размена говорних порука. Гледаоци су представљени својим одабраним аватарима који се приказују на њиховом ТВ екрану и екрану њихових пријатеља који гледају исти канал.

Корисници имају могућност да изразе своја осећања адаптирајући и објављујући емоције својих аватара на екранима пријатеља. Ове емоције могу бити у виду слика, видео или аудио порука, могу бити персонализоване и прилагођене садржају који се у том моменту приказује на телевизији. Нпр. навијач неког фудбалског тима може да користи слику неког од играча као свој аватар, да је прилагођава у зависности од резултата (тужан, радостан, насилан и сл.) како би изразио своје емоције, слично као на стадиону, провоцирајући навијаче супротног тима из своје листе пријатеља. Графика се приказује на слоју преко телевизијског како би се гледалац остао удубљен у ТВ садржај. На слици

2.1 је приказана говорна комуникација и аватари гледалаца у току заједничког гледања фудбалске утакмице.

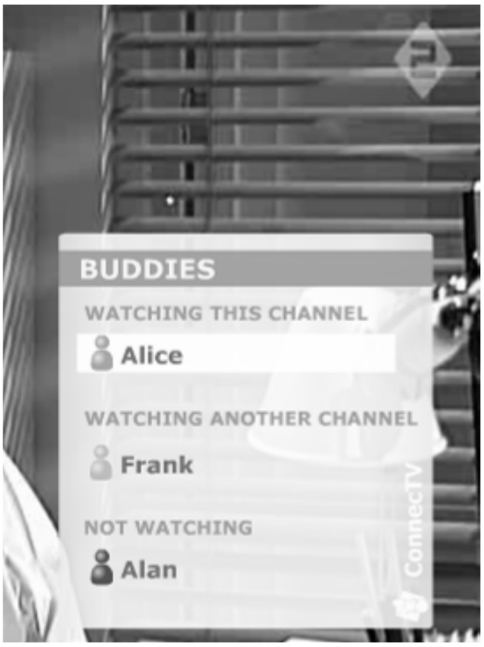
Слика 2.1 *AmigoTV*: говорна комуникација и представљање гледалаца аватарима, преузето из [19]

Како би препоручили или пронашли занимљив садржај, људи се често ослањају на пријатеље, па је у *AmigoTV* укључена и функција позива пријатељу да гледа исти канал. Гледалац има могућност да за сваки канал види списак пријатеља који га тренутно гледају и имају могућност да им се прикључе. Како би се заштитила приватност *AmigoTV* пружа кориснику могућност да постави своју видљивост на заузет или није укључен. Видљивост се може постављати појединачно за сваки канал, а предвиђена је и могућност забране примања гласовних порука. Како би се обезбедило једноставно управљање садржајем користи се само даљински управљач, а избегнуто је коришћење тастатуре и миша је избегнуто.

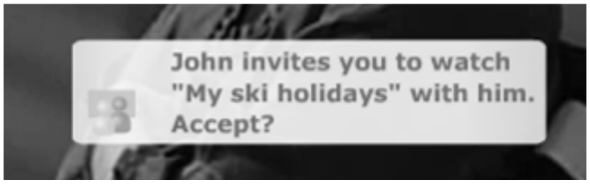
Пилот пројекат, назван *ConnectTV* представља сервис који има за циљ да интегрише комуникацију у искуство гледања телевизије и да истражи утицај који социјализација има на понашање и искуство гледалаца. Сервис нуди функционалности од којих су најважније: преглед програма и емисија које пријатељи тренутно прате, преглед препорука пријатеља и могућност заједничког гледања видео садржаја који није везан за програм који се емитује на ТВ-у.

У току гледања телевизије *ConnectTV* нуди могућност обавештавања корисника о томе ко од пријатеља гледа исти програм. У тој ситуацији корисници

могу да успоставе и међусобну говорну комуникацију. Како би се обезбедила интеграција са гледањем телевизије користе се микрофон и ТВ звучник. Корисници могу да приступе својој листи пријатеља и прегледају шта они управо гледају, као што је приказано на слици 2.2.




Слика 2.2 *ConnectTV*: преглед листе пријатеља, преузето из [13]




Слика 2.3 *ConnectTV*: позив пријатељу за заједничко гледање слика са одмора, преузето из [13]

Корисник има могућност да у току гледања ТВ програма за који сматра да би био занимљив и неком од његових пријатеља истог тренутка пошаље препоруку. У случају када пријатељ исто гледа телевизију на екрану му се приказује порука са препоруком и позивом за пребацивање на одговарајући канал. У случају када корисник не гледа телевизију, у моменту када је препорука послата започеће

снимање препорученог програма, како би корисник касније могао да га погледа. Локација на којој се снимљени програм памти може бити код корисника коме је послат позив или код корисника који је упутио позив, одакле га је могуће преузети коришћењем *peer-to-peer* архитектуре. Такође снимљени садржај је могуће запамтити и преко трећег лица који нуди простор за складиштење.

Осим гледања телевизије *ConnectTV* се може искористити и за гледање другог садржаја. Корисници могу да позову једни друге да гледају нпр. видео са одмора, као што је приказано на слици 2.3. У случају када корисник прихвати позив у истом моменту садржај се приказује на оба телевизора. Оба корисника могу да управљају садржајем који се гледа, да га паузирају , премотавају итд.

Предложени *ConnectTV* сервис је тестиран је у 50 домаћинстава чији чланови су имали ранијих социјалних контаката [14]. Сваки корисник је имао рачунар опремљен ТВ картицом, даљинским управљачем и *ConnectTV* софтвером. Основна мерења су показала да су корисници гледали телевизију у просеку 16,1 час недељно пре употребе *ConnectTV*-а. У почетној фази тестирања гледаност је порасла на просечно 17,9 часова недељно, да би се касније спустила на 12,3 часа. Технички проблеми су довели до тога да се одређен број људи опредељивао за гледање телевизије без употребе *ConnectTV*-а. У току гледања телевизије без *ConnectTV*-а корисници су се осећали доста неутрално: помало активно, помало пријатно, помало изоловано и помало досадно. Употребом *ConnectTV*-а порастао је осећај активности и повезаности, али је осећај задовољства опао. Резултати су такође показали да је више од 50% корисника окарактерисало *ConnectTV* као интересантан додатак искуству гледања телевизије и да би га требало унапредити додатним комуникационим функцијама.

Систем, назван *CollaboraTV*, је развијен како би се проучило и демонстрирало заједничко искуство гледања телевизије подржавајући синхрону и асинхрону комуникацију удаљених пријатеља. Идеја презентована у раду је да се емисије које се приказују на телевизији приказују са „виртуелном публиком“. „Виртуелна публика“ представља аватаре гледалаца са могућношћу изражавања емоција и коментарисања. Корисници имају могућност приступа листи пријатеља и електронском програмском водичу. Програмски водич приказује све емисије које

су доступне у систему, са нагласком на оне које су пријатељи гледали. Такође нуди и посебну листу најпопуларнијих емисија.

У случају када пријатељи синхроно гледају исту емисију имају могућност да у реалном времену размењују текстуалне поруке. Ове поруке се снимају и пријатељу који касније гледа исти програм ће се приказивати паралелно са емисијом. Такође, корисник у току гледања може додати и своје поруке. Осим порука могу се снимити тачке од интереса додавањем негативних или позитивних реакција на садржај као и различитих израза лица коришћењем аватара. На слици 2.4 је представљен део виртуелне публике из перспективе гледаоца. Аватари приказују претходне гледаоце истог програма. Коментар је представљен облачићем са леве стране, централни аватар приказује емоције изразом лица, а десни аватар изражава позитивну реакцију подизањем палца. Ове анотације су такође привремено приказане и на линији напретка, изнад аватара.

Систем је развијен коришћењем клијент-сервер архитектуре. На клијентској страни се приказује садржај и контролише програм који клијент тренутно гледа. Сервер кориснику доставља тражени видео, координише приступима истом видеу, омогућава оцену програма, приступ анотацијама и листи пријатеља. Тестирање је извршено у трајању од 4 недеље. Систем је тестирало 16 студената за време трајања летњег распуста. Студенти су имали претходних социјалних контаката у току студија, а за време распуста су се налазили у различитим градовима. Имали су могућност избора гледања телевизије са или без *CollaboraTV*-а. Понуђено им је 37 различитих емисија са укупно 600 различитих епизода и били су замољени да погледају барем 2 епизоде, од којих би једну гледали у исто време са пријатељем, а другу у различитим временским интервалима.

Бележене су активности корисника и корисници су попуњавали Ликерт упитнике. Већина корисника је изразила задовољство приликом коришћења и описала систем као забаван. Више од половине корисника се сложило да је у поређењу са традиционалним начином гледања телевизије виртуелна публика пријатнија. Корисници су били веома ангажовани док су гледали телевизију са виртуелном публиком, што им је обогатило искуство гледања и дало осећај друштвеног присуства. Неки од корисника су објаснили да им је гледање истог

програма који изазива јаке емоције код пријатеља повећао осећај повезаности. Жалбе су се односиле на број људи који је био укључен у истраживање, претпостављајући да би искуство гледања било много боље да је постојао већи број учесника.


Слика 2.4 *CollaboraTV*: виртуелна публика, преузето из [52]

За разлику од осталих система *Motorola STV* користи и амбијентално обавештавање о присуству корисника. Као и у претходним прототиповима социјалне и интерактивне телевизије корисник може да приступи листи својих пријатеља, да види ко од пријатеља тренутно гледа телевизију и који програм. Главни мени прототипа је приказан на слици 2.5. Као додатне опције корисник има могућност да прегледа своју историју гледаних програма као и могућност да направи план емисија које ће пратити и да прегледа план својих пријатеља. Има могућност да прегледа шта су његови пријатељи гледали у последњих 10 минута. Пријатељи могу позивати једни друге да гледају исту емисију. Ако два пријатеља у исто време гледају исти програм имају могућност да један другоме шаљу неку од 20 предефинисаних порука.

Прототип је имплементиран коришћењем рачунара са ТВ картицом на ком се извршавао GB-PVR софтвер и био је повезан на велики ТВ екран. Социјална телевизија је развијена као додатак GB-PVR софтверу допуњујући га новим прозорима, менијима и контролама које се приказују преко ТВ слике. За све видове интеракције коришћен је даљински управљач. Ако је корисник у другој

соби или није у могућности да види ТВ, коришћени су додатни уређаји како би био обавештен о присуству пријатеља. Као примарни уређај коришћена је *AmbientOrb* лампа која има могућност мењања боје. Одговарајућа боја представља индикатор броја пријатеља који гледају телевизију. Због потребе да буде повезана серијским каблом са рачунаром лампа је била смештена у дневној соби. Како већина корисника велики део времена проводи и ван дневне собе коришћен је и други уређај, LCD дисплеј, који је бежичном везом повезан са прототипом.


Слика 2.5 *Motorola STV*: главни мени, преузето из [51]

Испитивање је извршено посматрањем пет домаћинстава која су у току две недеље користили предложени прототип система. Систем је инсталиран у дневној соби, као месту на коме се најчешће гледа телевизија. Искуства корисника су прикупљана из интервјуа и дневним бележењем активности. Резултати су показали да је корисницима највише сметало што у систему не постоји могућност конверзације, понуђене поруке најчешће нису најбоље описивале оно што су хтели једни другима да кажу, па су често звали једни друге телефоном. Коришћење амбијенталних уређаја појачало је знатижељу и подстицало кориснике да укључе телевизор како би видели шта пријатељи гледају или шта су означили да ће гледати. Неки од испитаника су изјавили да су често гледали програм који је био жеља пријатеља, а не оно што би сами изабрали. Систем им је

помогао да науче више о навикама гледања телевизије својих пријатеља и дао им је осећају да су у друштву.

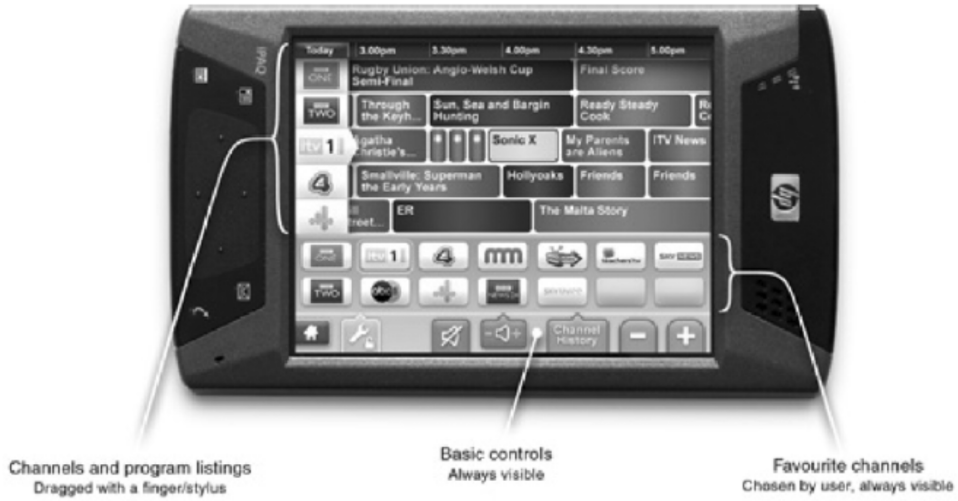
Сви наведени прототипови система се фокусирају на ситуације у којима су гледаоци физички одвојени и покушавају да обезбеде њихову социјализацију нудећи различите видове комуникације као што су: размена гласовних, текстуалних порука, могућност да се искажу емоције, коришћење листе пријатеља и провера шта они гледају, коришћење аватара како би се приказали гледаоци и коришћење додатних уређаја како би и када не гледају телевизију корисници имали увид у то колико пријатеља је гледа. Оно што је заједничко за све системе је да се углавном фокусирају на ситуације у којима само један корисник седи испред телевизора.

### 2.1.2 Интерактивни системи који користе други екран

У последњих неколико година осим телевизора који је одувек био доминантни уређај у дневној соби све је више и других уређаја и екрана који се појављују у овом заједничком простору. Као пример обично се лаптоп уређај налази на сточићу за кафу, мобилни уређаји у џеповима корисника и таблет уређај на софи. Студија која је спроведена у сарадњи *Nielsen* и *Yahoo!Inc* компаније [75] показује како 42% власника таблет уређаја и 40% власника мобилних телефона користи своје уређаје у паралели са гледањем телевизије. Активности које корисници обављају су приступ е-пошти, интернету, размена текстуалних порука и као најфреквентнији приступ друштвеним мрежама [54]. Како интерактивна телевизија постаје све популарнија и осим ТВ екрана у дневној соби су присутни и други екрани који се користе у паралели јавља се идеја да се ти додатни екрани могу искористити и у циљу побољшања искуства гледања телевизије.

У раду [21] представљен је систем у ком се користи лични дигитални помоћник (*PersonalDigitalAssistant* - PDA) за приказивање персонализованог електронског програмског водича и додатне контроле као што су управљање јачином звука и промена канала. Главна идеја је да се део графике која би се иначе приказивала на телевизору приликом прегледа програмског водича приказује на додатном екрану и да се исти уређај користи уместо даљинског управљача приликом употреба најчешћих команди јер као предност поседује

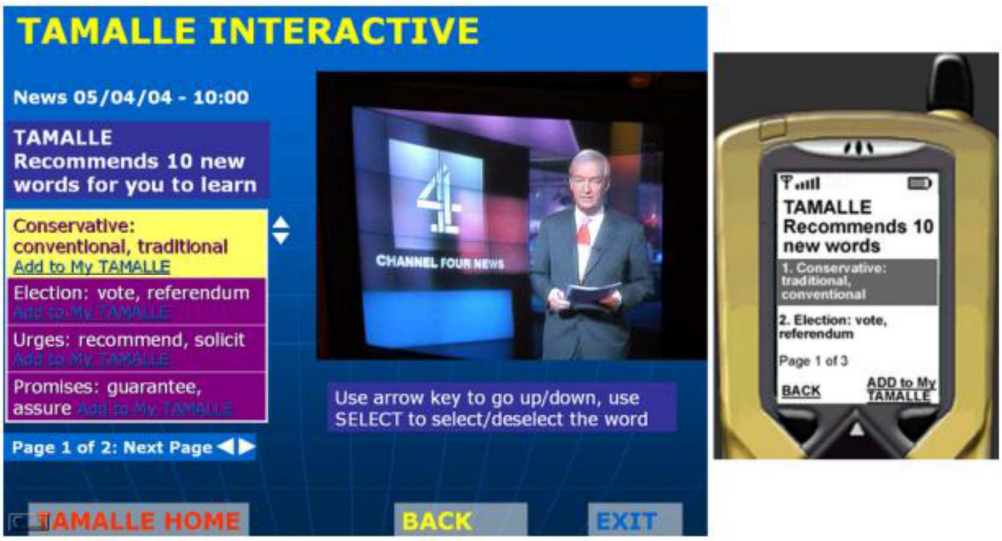
особину да не мора да се упери ка телевизору. Кориснички интерфејс је приказан на слици 2.6. Како би се оценило задовољство корисника систем је испробан у 20 домаћинстава. Чланови домаћинстава којих је укупно било 62 су интервјуисани пре и након употребе система.




Слика 2.6 Кориснички интерфејс на екрану персоналног дигиталног помоћника у систему у ком се додатни екран користи за управљање телевизором, преузето из [21]

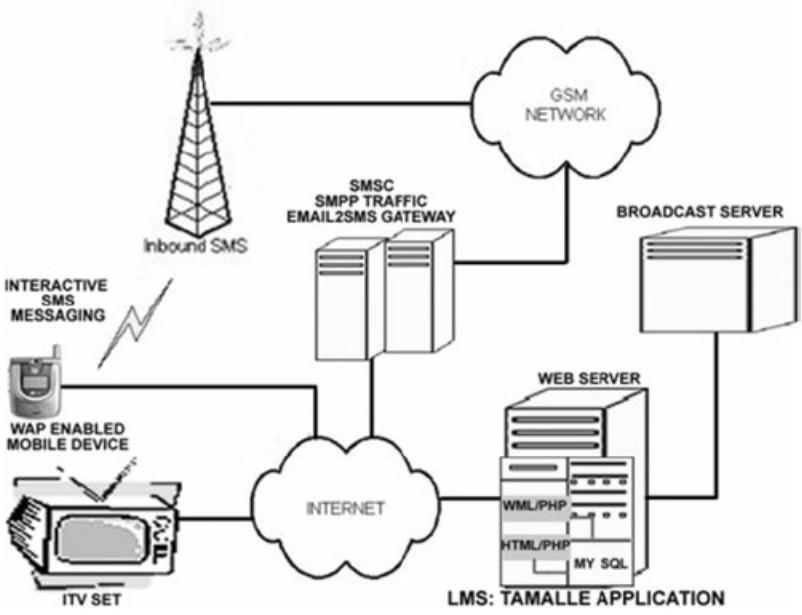
Систем је имплементиран коришћењем лаптоп рачунара који има могућност пријема ТВ сигнала. Лаптоп је имао могућност рада на два различита начина. У првом начину рада ТВ канали су могли да се мењају помоћу личног дигиталног помоћника који је са лаптоп рачунаром био повезан преко *Bluetooth*везе, а у другом су ТВ канали мењани коришћењем инфраред даљинског управљача. За приказ електронског програмског водича коришћени су *TV-Anytime* [65] метаподаци доступни преко одговарајуће интернет странице. Иницијални резултати употребе су показали да је реакција корисника на приказ програмског водича на додатном екрану била веома позитивна. Већина корисника је изјавила да коришћење још једног уређаја за приказ информација и управљање ТВ садржајем проналази корисним. Корисницима се више свидело да читају са екрана личног дигиталног помоћника него са ТВ екрана и радије су користили лични дигитални помоћник него даљински управљач.

Пројекат назван TAMALE (*Television and Mobile phone Assisted Language Learning Environment*) [29] је развијен са идејом да се коришћењем интерактивне телевизије и мобилних уређаја помогне неформално учење страног језика. Кориснички интерфејс оба уређаја је приказан на слици 2.7. Десна страна телевизијског екрана је искоришћена за приказ ТВ програма, док се са леве стране приказују препоручене речи и изрази са објашњењем. Речи од значаја је могуће на мобилном уређају додати у персонализовану листу за касније преглед и учење. Такође, пре гледања емисије могуће је учитати и погледати препоручене речи и кратак опис емисије.




Слика 2.7 Кориснички интерфејс TAMALE пројекта – симултани приказ на ТВ и мобилном уређају, преузето из [29]

Прототип система је развијен коришћењем клијент-сервер архитектуре. Сервер је задужен за достављање емитованог програма и садржаја који је везан за учење језика дигиталним ТВ пријемницима и мобилним уређајима који представљају клијенте. Такође, садржај који је потребно доставити клијентима се чува на серверу у одговарајућој бази података. Двосмерна комуникација између сервера и дигиталног ТВ пријемника се успоставља коришћењем модема, ADSL-а или широкопојасног кабла, а комуникација између сервера и мобилног уређаја коришћењем WAP протокола и интерактивних SMS порука. Архитектура TAMALE прототипа је представљена на слици 2.8.

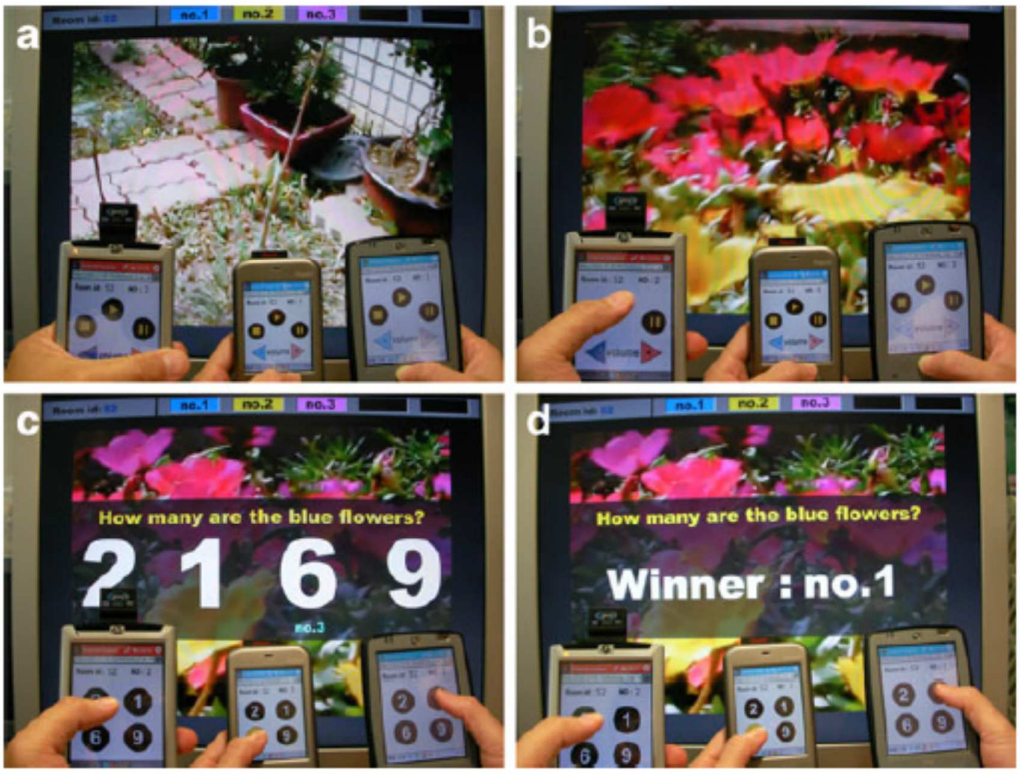



Слика 2.8 Архитектура TAMALE прототипа, преузето из [29]

Евалуација система је извршена на Универзитету у Брајтону у школској лабораторији. Систем је испробало 14 волонтера, студената и евалуација је извршена опсервацијом, слободним интервјуима и попуњавањем унапред припремљених упитника. Корисници су систем оценили као користан, лак за употребу, такође као позитивну ствар су навели и повећање мотивације за учење страних речи, а највише користи имали су приликом учења идиома и израза. Као негативну страну корисници су напоменули одвлачење пажње од праћења ТВ емисије и потешкоће у читању текста у паралели са праћењем ТВ емисије.

У раду [72] развијен је систем, назван *Multi-user Variable Remote Control iTV service* (MVC-iTV), који омогућава да се мултимедијални садржај приказан на телевизији, контролише помоћу више мобилних уређаја који функционишу као даљински управљачи. Корисници су смештени у такозване „виртуелне собе“ и у исто време може бити активно више „виртуелних соба“. Корисник може да се пријави само у једној „виртуелној соби“, а у свакој соби постоји један уређај са великим екраном на ком се приказује јавни садржај. Корисник се пријављује у „виртуелну собу“ уношењем одговарајућег кода који се појављује на јавном

екрану. Када се региструје у одређеној соби корисник може да управља садржајем који се појављује на великом екрану те собе, али не може да утиче на садржај екрана у другим собама.

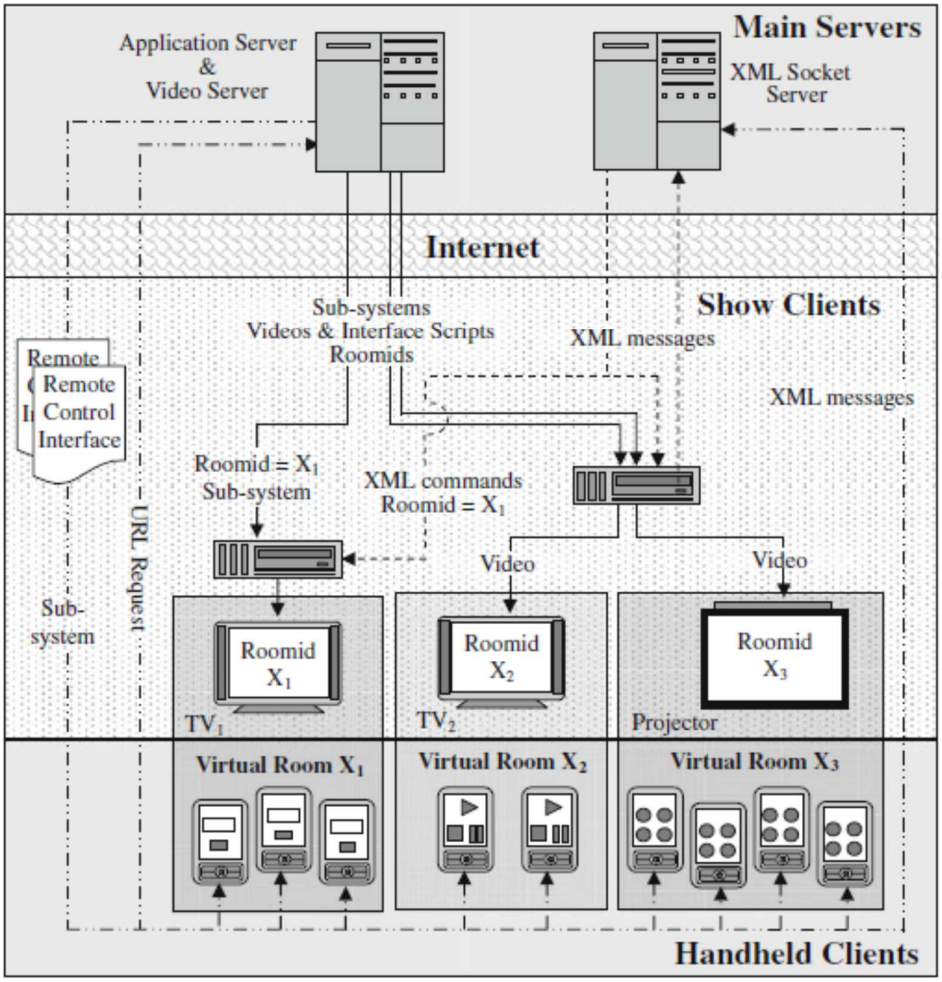



Слика 2.9 Кориснички интерфејс на ТВ и мобилним уређајима MVC-iTV система, преузето из [72]

На слици 2.9 у горњем левом углу, означеним са a, је представљен почетни кориснички интерфејс на мобилним уређајима. Сваки клијент има могућност да заустави, паузира и покрене видео који се тренутно приказује. У горњем десном углу, означеним са b, један од корисника покреће паузирани видео. Интерфејс на мобилним уређајима је освежен у зависности од тога шта се приказује на телевизијском екрану, као што је представљено у доњем левом углу слике, означеним са c. У доњем десном углу слике, означеним са d, представљен је утицај корисника на видео који се приказује.

Систем је развијен коришћењем IM (*Instant Messaging System*) система који је базиран на M2M (*Machine to Machine*) комуникацији. Идеја аутора је да се систем

који се користи за комуникацију различитих корисника може искористи и за слање порука између различитих уређаја које исти корисник употребљава. Архитектура система, представљена на слици 2.10, је подељена на три слоја: слој сервера, слој ТВ уређаја који се користе за приказ информација и слој мобилних уређаја који се користе за управљање садржајем. Сервери контролишу шта се на ком уређају приказује и управљају везом између уређаја. На ТВ уређајима се приказује видео достављен од стране видео сервера. Ови уређаји такође примају и извршавају поруке у случају када корисници предузму акције коришћењем мобилних уређаја. Мобилни уређаји примају од сервера и интерфејс који је потребно да прикажу. Интерфејс се састоји од виртуелних дугмића које је могуће притиснути.




Слика 2.10 Архитектура MVC-iTV система, преузето из [72]

Прототип система је развијен коришћењем *Microsoft ASP* технологије на серверима и *Adobe Flash* платформе за развој корисничких интерфејса на клијентима. Самсунгов 19'' монитор је повезан на рачунар са *Microsoft Windows* оперативним системом симулирајући ТВ пријемник. Претпоставка аутора је да би овакав систем мога да буде занимљив и за реализацију у реалним условима и да би могао бити реализован коришћењем дигиталних ТВ пријемника.

У раду [16] дат је предлог интерактивног система код ког се додатни екран мобилног уређаја користи у циљу приказивања информација о програму који се тренутно гледа на телевизији. Када се стартује апликација на мобилном уређају корисник има могућност да детектује о ком програму се ради. На телевизији се емитује 18 различитих канала са по петоминутним видеима који су припадају различитим жанровима (драма, спорт, филмови, путовања итд). Претраживање канала се врши снимањем амбијенталног звука коришћењем мобилног уређаја и претраживањем базе видеа. На основу добијених информација кориснику се приказују додатне информације о програму који гледа са леве стране екрана мобилног телефона, а са десне стране на располагању му је главни мени, као што је приказано на слици 2.11. Из главног менија корисник има могућност да изабере информације као што су сиже програма, информације о глумцима, статистике везано за спорт, додатне информације о неком производу и да комуницирају користећи размену текстуалних порука са другим корисницима система који гледају исти програм.

Прототип система је направљен користећи уређаје са Андроид оперативним системом. У циљу евалуације задовољства корисника приликом употребе система изабрано је 20 корисника који су испробавали систем. Корисници су требали да одраде следеће задатке: синхронизацију мобилног и ТВ уређаја, претрагу додатних информација о филму и глумцима, проверу статуса филма и куповину производа. Након обављених задатака изражавали су своје задовољство коришћењем Ликерт скале. Корисници су успешно обавили све задатке. Могућност аутоматског детектовања програма који се гледа се нарочито свидела корисницима, али су изразили и забринутост о поузданости система у бучној средини. Корисници су изразили жељу да у будућности користе апликацију, али

под условом да је бесплатна. Такође, дали су предлоге за другачији дизајн система и изразили жељу за додатним карактеристикама као што су препоруке програма и могућност да управљају телевизијом коришћењем мобилног уређаја.


Слика 2.11 Кориснички интерфејс система који користи екран мобилног уређаја за приказ додатних информација о програму који корисник тренутно прати, преузето из [16]

Имајући на уму да интерактивна телевизија постаје све популарнија и да се у паралели са гледањем телевизије користе и други уређаји отварају се различите могућности за употребу додатних екрана како би обогатили искуство гледања телевизије. Иако је пораст броја уређаја у дневној соби позната ствар, још увек је недовољно истраживања спроведено о начинима њиховог повезивања и употребе од стране корисника.

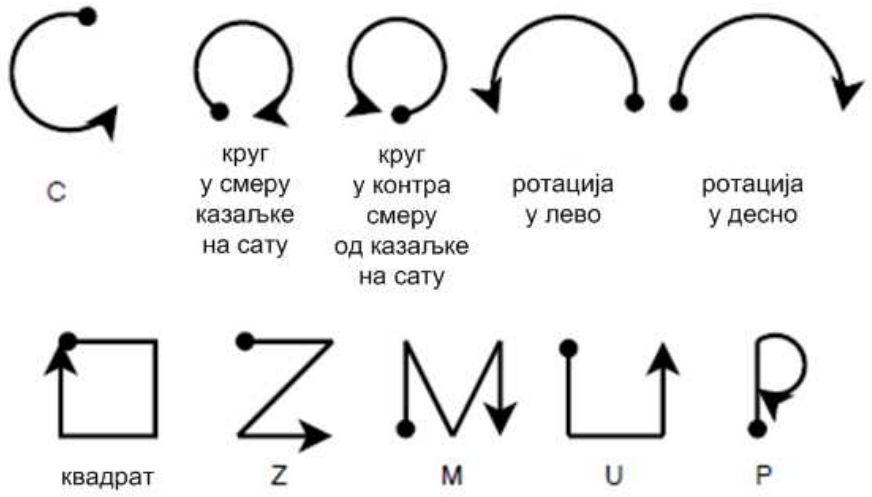
### 2.1.3 Препознавање покрета коришћењем мобилних уређаја

Мобилни уређаји су опремљени различитим моћним сензорима као што су: GPS (*Global Positioning System*) сензори, звучни сензори, камере, сензори осветљења, температурни сензори и акцелерометри. Због своје мале величине, широке употребе, могућности да приме, шаљу и обрађују податке отварају простор за нова истраживања на пољу интеракције човека са рачунаром. Употреба

акцелерометра је првенствено пружила могућност за аутоматско одређивање положаја у ком корисник држи уређај. На основу положаја уређаја садржај на екрану се приказује хоризонтално или вертикално [44]. Осим ове основне намене, употреба акцелерометра је нашла примену и у различитим врстама апликација које се користе за препознавање покрета корисника који поседује мобилни уређај.

У литератури су представљена решења која детектују различите покрете људи коришћењем већег броја акцелерометара распоређених на различитим деловима људског тела [5]. Најновија истраживања показују да је само поседовањем мобилног телефона у џепу, без потребе за поседовањем додатног уређаја, могуће ефикасно разликовати акције као што су: шетање, трчање, пењање уз степенице, спуштање низ степенице, седење или стајање [43].

Систем назван *PhonePoint Pen* користи акцелерометре уграђене у мобилне уређаје како би препознао карактере енглеског алфабета [2]. Држањем телефона као оловке корисник може да пише кратке поруке у ваздуху. Убрзање које се детектује приликом померања телефона у руци је могуће трансформисати у потезе који се могу представити као одређени карактери.




Слика 2.12 Покрети које је могуће препознати коришћењем *grmobile* система, преузето из [39]

На сличан начин у раду [39] развијен је систем назван *grmobile* са циљем да испита детекцију десет различитих покрета. Покрети су представљени на слици

2.12. Сваки од покрета је могуће извести на два начина или померањем телефона у ваздуху или коришћењем екрана осетљивог на додир. Тестирањем система је показало да је око 89% покрета телефоном у ваздуху успешно препознато, односно 98% покрета прстом по екрану телефона.

Због ограничења која пружају стандардни уређаји за управљање садржајем на великом екрану као што су даљински управљачи, миш и тастатура велики број истраживања се фокусирао и на испитивањима различитих могућности управљања садржајем употребом мобилних уређаја у циљу препознавања интерактивних гестова. Спроведено је истраживање у циљу утврђивања прихватања гестова као начина интеракције са мобилним уређајем [11]. Од учесника истраживања је тражено да испробају употребу гестова као што је додир екрана прстима, померање прстима по екрану и померање самог уређаја у различитим правцима. Учесници истраживања су јасно фаворизовали гестове у односу на традиционалне начине интеракције притиском на дугме.

Слика 2.13 *MobiToss*систем и гест „бацања“ видеа на велики екран, преузето из [58]

У раду [58] представљен је систем, назван *MobiToss*, који омогућава пренос и преглед мултимедијалног садржаја са мобилног телефона на велики екран коришћењем гестова. Корисник коришћењем мобилног телефона може да сними

фотографију и видео, а након тога коришћењем геста „бацања“ да снимљени фајл пренесе на велики екран у циљу лакшег прегледа. Такође искошењем мобилног уређаја у једну или другу страну омогућава се и управљање видеом. *MobiToss* је испитало 25 корисника и изведен је закључак да су сви корисници уживали у употреби апликације, са посебним акцентом на коришћење геста „бацања“. На слици 2.13 је приказан гест „бацања“ видеа на велики екран.

Као најчешће коришћени гестови за управљање садржајем на великим екранима су: искошења мобилног уређаја, симулација „бацања“ коришћењем мобилног уређаја и коришћење екрана осетљивог на додир за селекцију [23]. Искошењем мобилног телефона могуће је вршити претраживање фајлова и навигацију, додиром је омогућена селекција, а гестом бацања је могуће извршити пренос података са мобилног уређаја на велики екран. Подаци које се преносе могу бити видео, музички фајлови и локација.

## 2.2 Интеракција човек-рачунар у играма

У овом поглављу посебна пажња посвећена је интеракцији човек-рачунар у играма. Прво ће се представити начини и уређаји који се користе за препознавање покрета у играма са посебним акцентом на употреби мобилних уређаја као контролера у играма. Након тога, представиће се различити начини реализације умрежених игара са више играча. У циљу очувања комуникације лицем у лице, а коришћењем предности које пружа дигитализација приказаће се и постојећи системи у којима је покушана дигитализација друштвених игара.

### 2.2.1 Препознавање покрета у играма

Стабилан раст и напредак у индустрији видео игара у последњих неколико деценија довео је и до великог пословног напретка ове гране која од 2007. године чак превазилази и бруто зараду филмске индустрије. Развој графичких картица које омогућавају визуелно богате и реалистичне сцене у видео играма је великим делом одговоран за велики напредак у овој грани индустрије. Како квалитет графике више не представља велику новост програмери и дизајнери видео игара су почели да траже нове начине да привуку и забаве играче. Један од трендова је и обезбеђивање нових форми интеракције које употребу традиционалних уређаја за

интеракцију (тастатура, миш и џојстик) замењују сложенијим уређајима као што су: простирка за плес, гитара, бубњеви, показивачки уређаји и слично. На слици 2.14 су представљена три примера уређаја за интеракцију са играма: простирка за плес, гитара и *Wiimote* показивачки уређај.

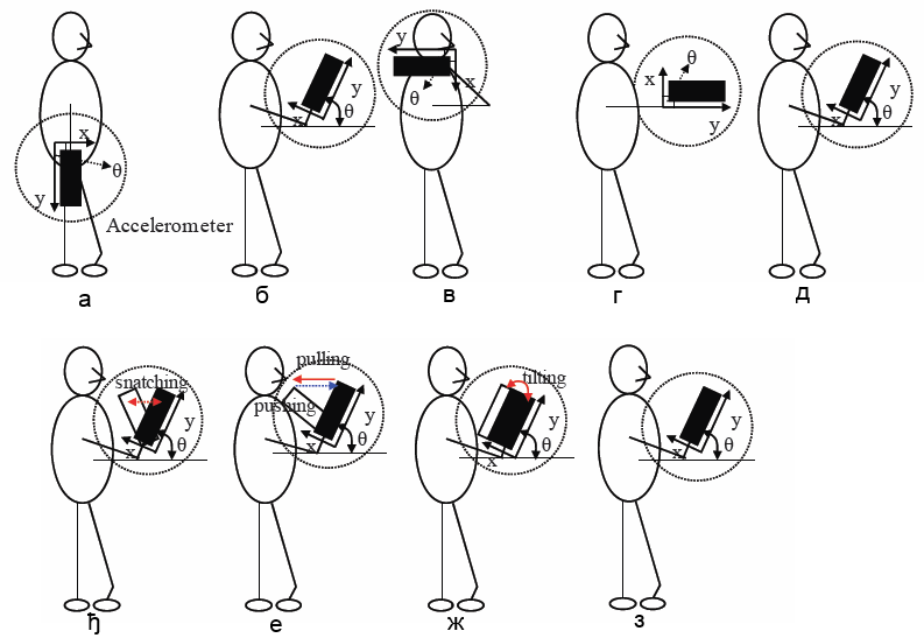

Слика 2.14 Уређаји за интеракцију са играма: простирка за плес, гитара и *Wiimote* показивачки уређај

Нитендо је једна од првих компанија која је инвестирала у показивачки уређај, назван *Wiimote*, који има могућност детекције покрета играча. Овај уређај омогућава играчима да интерреагују са игром коришћењем својих природних покрета тела [73]. Игре које користе *Wiimote* уређај препознају ротацију око x,y и z осе показивачког уређаја коришћењем акцелерометра и оптичких сензора и на тај начин детектују покрет руке. Слично *Wiimote* уређају и Сони је представио уређај, назван *PlayStationMove*, за детекцију покрета у играма које користе *PlayStation* конзолу. Компанија Мајкрософт је имала другачији приступ и лансирала је систем, назван *Kinect*, који користи камере и сензоре у циљу праћења и препознавања покрета играча. Уређај *Kinect* представља велики напредак на пољу интеракције човека и рачунара у играма елиминишући употребу додатног контролера. Уместо улазног уређаја користи се природни покрет тела, обезбеђујући интуитивну контролу игре [79]. Сасвим је извесно да су главни произвођачи видео игара заинтересовани за истраживање нових и интуитивних

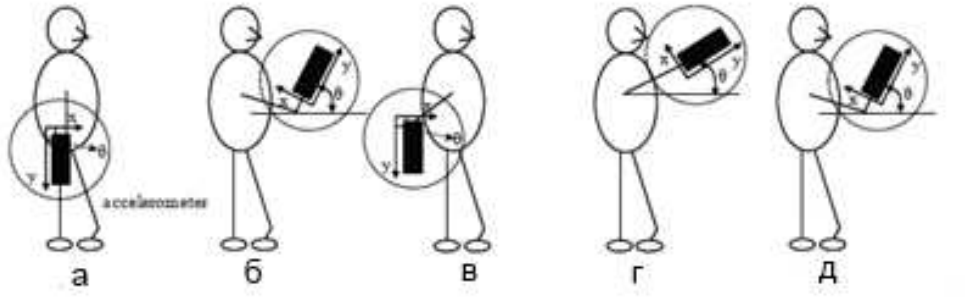
начина интеракције, а степен оригиналности у интеракцији коју пружају може бити један од кључних фактора за успех производа.

У новије време, у оквиру играчке заједнице се појавило и интересовање за различите начине употребе мобилних уређаја као контролера у играма. Већина популарних мобилних уређаја не поседује довољно процесорске снаге да би извршавала графички захтевне игре, али се могу употребити за управљање играма коришћењем сензора и екрана који су осетљиви на додир. Како би слали или примали податке ослањају се на *Bluetooth* и *Wi-Fi* технологију.

У раду [3] мобилни телефони опремљени акцелерометром су искоришћени за препознавање тачно дефинисане секвенце покрета која може да се искористи у играма пецања и куглања. Коришћен је аутомат стања како би се на основу излаза акцелерометра по x и y оси препознали прелази из једног положаја у други. На слици 2.15 је представљена секвенца покрета за игру пецања, а на слици 2.16 за игру куглања.




Слика 2.15 Покрети коришћени код игре пецања и положаји акцелерометра а) стартно стање б) почетак игре в) замах г) бацање д) чекање да риба загризе ђ) риба је загризла е) повлачење ж) ротирање з) завршетак, преузето из [3]




Слика 2.16 Покрети коришћени код игре куглања и положаји акцелерометра а) стартно стање б) почетак игре в) замах г) бацање д) завршетак, преузето из [3]

Као пример коришћења мобилних уређаја као контролера у игри која се приказују на великом екрану развијен је *Popet* систем [69]. Игра је имплементирана коришћењем клијент-сервер архитектуре, а комуникација између мобилних уређаја и сервера се одвија коришћењем *Bluetooth* технологије. Коришћењем акцелерометра, које поседују мобилни уређаји, бележи се искошење уређаја у леву или десну страну. Вредности са излаза акцелерометра се шаљу централном серверу игре који израчунава нову позицију аватара у зависности од тога колико је телефон искошен, као што је представљено на слици 2.17. У оквиру *Popet* система предвиђено је слање информација само у једном смеру, од мобилног уређаја ка централном серверу, али је изостављена могућност комуникације и у другом смеру, од сервера ка мобилним уређајима.

Слика 2.17 *Poppet* систем, преузето из [69]

У раду [47] развијен је систем који такође омогућава контролисање игара на великом екрану помоћу мобилних уређаја и за разлику од *Popet* система пружа могућност двосмерне комуникације са циљем да се екран мобилног уређаја искористи за приказ информација као што је број освојених поена. Систем је такође развијен коришћењем клијент-сервер архитектуре и за размену података користи *Bluetooth* технологију. Игром се управља само притиском на одговарајуће дугмиће мобилног уређаја, а коришћење уграђених сензора у циљу препознавања покрета је у овом раду изостављено.

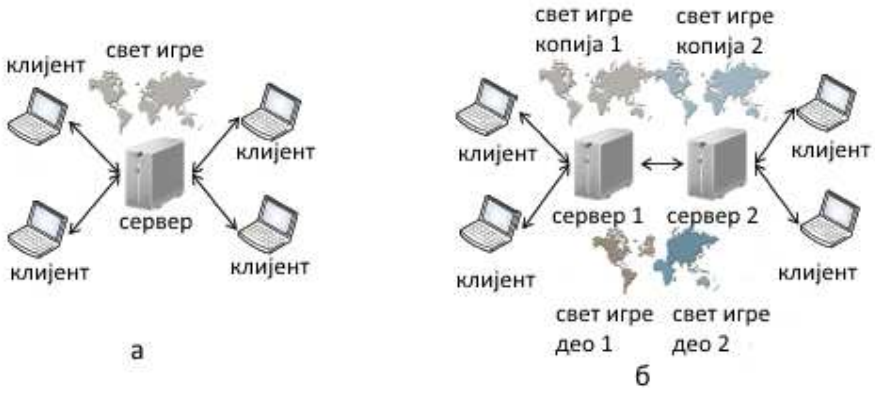
### 2.2.2 Умрежене игре са више играча

Умрежене игре са више играча пружају великом броју корисника могућност да у исто време користећи интернет играју исту игру. Корисници могу да играју као независни противници, могу да се групишу у тимове, да играју једни против других или да играју против саме игре која је реализована применом вештачке интелигенције. Неке од постојећих умрежених игара са више играча (*WorldofWarcraft* [10], *EVEOnline* [27] и *Final Fantasy XIV* [32] ) су у могућности да опслуже хиљаде играча и остварују приходе који се мере билионима долара.

Постоје две основне архитектуре умрежених игара са више играча клијент-сервер архитектура која тренутно преовладава и *peer-to-peer* (P2P) архитектура [76]. Код клијент-сервер архитектуре на серверу су смештене главне копије свих објеката који се појављују у игри. Главна копија света игре се такође чува и ажурира на серверу. Све акције играча се шаљу серверу. Клијенти се повезују са сервером и од њега примају потребне информације о свету игре, а свим заинтересованим клијентима сервер шаље информације потребне за ажурирање објеката који се појављују у игри.

Како би се повећала скалабилност система могуће је додавање и више сервера који опслужују различите групе клијената [42]. Архитектуре које користе више сервера се могу поделити на две категорије. У првој категорији постоји неколико потпуних инстанци света игре, који се називају парчићи (*shards*), а један сервер одржава једно парче. Сваки сервер је одговоран за одређени скуп клијената и поседује комплетну копију света игре. Један сервер и скуп клијената који му је придружен прате традиционалну клијент-сервер архитектуру. У зависности од

географске локације корисници се додељују одговарајућем серверу. Сваки сервер је одговоран за одвојене регионе као што су на пример Европа и Америка. Најчешће не постоји потреба за међусобном комуникацијом између сервера. У другој категорији постоји само један свет игре који је подељен на неколико региона, а сваки регион одржава одређени сервер. Сви играчи се налазе у истом свету игре, али у већини случајева они су у могућности да интерреагују само са играчима који припадају истом региону. Играчи могу да прелазе из једног региона у други, али то захтева постојање механизма предаје између различитих сервера. Када се играч приближи граници другог региона потребне информације о суседном региону се шаљу играчу и играч може једноставно да пређе у нови регион. Механизам предаје може бити и транспарентан играчу и у том случају се од играча тражи да прође кроз специјална врата или капију како би ушао у нови регион. Клијент-сервер архитектура са једним и већим бројем сервера је приказана на слици 2.18.




Слика 2.18 Клијент-сервер архитектура игре а) са једним сервером и б) са више сервера

Други начин повећања скалабилности система је коришћење *peer-to-peer* архитектуре. Код *peer-to-peer* архитектуре сваки чвор се уједно понаша и као сервер и као клијент. Сваки чвор може бити одговоран за одржавање главних копија делова света игре и за информисање осталих чворова о променама. Овај модел је најчешће јако скалабилан јер је оптерећење распоређено на све чворове и укључивање новог чвора додаје нове ресурсе. У *peer-to-peer* системима кориснику је једноставно да вара, па је највећа мана ових система сигурност [6]. Како не постоји централни сервер који одржава главно стање *peer-to-peer* системе је теже

одржавати, контролисати и постоји већа вероватноћа појављивања недоследности. Како се број чворова у систему повећава тако расте комплексност система и потреба за координацијом. *Peer-to-peer* архитектура игре је приказана на слици 2.19.

Централизована клијент-сервер архитектура је популарнија из разлога што пружа већу контролу над игром. Креатори игре могу на једноставан начин да ажурирају стање игре и преузму контролу над неопходним исправкама у софтверу. Такође, сервер је задужен за разрешавање конфликата, па је одржавање конзистентности једноставније него код система са *peer-to-peer* архитектуром.


Слика 2.19 Peer-to-peer архитектура игре

Са развојем мобилних уређаја развијале су се и игре које је могуће на њима играти. Како мобилни уређаји имају ограничене ресурсе све је више система који захтевна израчунавања пребацују на интернет сервере уместо да их извршавају на самом мобилном уређају [41]. Дужности је могуће поделити тако што ће се израчунавања везана за логику игре извршавати на интернет серверима [78], а израчунавања везана за исцртавање на мобилним уређајима. Такође, могуће је и логику игре и исцртавања обавити на серверима, а мобилним уређајима само проследити видео који је потребно да се прикаже [34].

### 2.2.3 Дигитализација друштвених игара

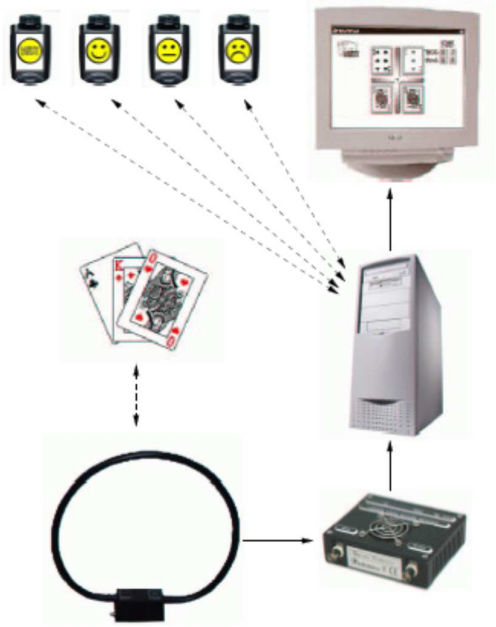
Играње друштвених игара је одувек представљало важан део социјализације људи. Друштвене игре се играју хиљадама годинама доприносећи забави,

релаксацији и едукацији. Иако у садашњици компјутерска индустрија креира мноштво алтернатива традиционалним друштвеним играма оне се и даље играју и продају у великом броју. Оно што доприноси успеху традиционалних друштвених игара је свакако социјални аспект. Готово све игре су намењене за више играча и њихово играње се организује у циљу окупљања пријатеља како би провели заједничко време. С обзиром на то да играчи седе заједно око стола и гледају се у очи, они имају могућност да интерпретирају мимике и гестове које им помажу у разумевању акција других играча. Играчи могу заједно да се смеју или да вичу једни на друге. Са друге стране, играње видео игара уводи новине као што су аудио, визуелна презентација и свет игре који зависи од маште програмера. Играчи могу добити тренутну повратну информацију о предузетој акцији што подстиче емотивну укљученост и бољу удубљеност у игру. Недостатак видео игара је интеракција људи лицем у лице. Иако се развијају и видео игре које су намењене за више играча који могу и физички да се налазе у истој просторији, интеракција човека је и даље првенствено усмерена на екран рачунара.

Како и друштвене и видео игре имају својих предности и мана јавила се идеја да се на неки начин обе врсте игара споје. Неколико покушаја је учињено да се премости јаз између друштвених игара на табли и видео игара [7], а утврђено је и да видео игре које су базиране на играма на табли спајају предности традиционалних друштвених и видео игара [4] комбинујући социјалну интеракцију са визуелним, аудио могућностима и повратним одговором видео игара [30]. Играчи могу да процене намере других играча посматрањем њихових акција [68], а компјутер може да преузме задатке који су људима досадни или тешки, као што је нпр. бројање поена. Још једна предност је и могућност да се сачува статус игре и да се игра настави касније. У циљу дигитализације друштвених игара на табли користи се додатни хардвер као што су: пројектори, визуелно препознавање, идентификација путем радио фреквенције RFID (*Radio-frequency identification*) и велике табле (површине) које реагују на додир. У току игре могу да се користе физички објекти као што су пиони чије се кретање детектује камерама, објекти опремљени RFID таговима или виртуелни објекти на физичким површинама.

У раду [57] приказан је систем, назван *Smart Playing Cards*, који представља покушај дигитализације игре картама. *Smart Playing Cards* користи стандардни шпил карата за игру, као и у традиционалној игри, а провера правила, одређивање ко је на потезу, рачунање резултата и провера ваљаности потеза су дигитализоване. За реализацију прототипа коришћен је систем идентификације путем радио фреквенције који је повезан са десктоп рачунаром, лични дигитални помоћници и стандардни шпил од 52 карте у ком је свака карта опремљена RFID тагом.

Развијени софтвер се састоји од главне апликације која се извршава на десктоп рачунару и клијентске апликације која се извршава на личним дигиталним помоћницима. Обе апликације су имплементиране коришћењем Јава програмског језика. Главна апликација одржава стање игре коришћењем система идентификације путем радио фреквенције. Клијентска апликација врши класификацију потеза играча на исправне и неисправне потезе и у зависности од исхода користи визуелни индикатор на екрану личног дигиталног помоћника. Архитектура *Smart Playing Cards* система је представљена на слици 2.20.



Слика 2.20 Архитектура *Smart Playing Cards* система, преузето из [57]

Коришћени систем идентификације путем радио фреквенције може поуздано да детектује највише 12 карата на столу, такође поседује антену димензија 70x50cm и опсег детекције му је лопта пречника коришћене антене, што оставља довољно простора да карте на столу буду детектоване, али играчи морају да воде рачуна о томе да карте које држе у руци буду ван домашаја антене. Проблем се јавља и када се два или више тага преклопе приликом бацања карата на сто. У тој ситуацији систем није у стању да детектује ни једну од бачених карата.

Експериментална платформа, названа STARS, која коришћењем различитих мултимедијалних уређаја подржава класичне друштвене игре на табли представљена је у раду [50]. Ова платформа омогућава програмерима да креирају сложене сценарије игара који могу садржати елементе за сарадњу и конкурентске елементе у истој игри. Од компонената се користи плазма екран осетљив на додир који служи за приказ табле игре. Изнад табле је смештена камера која снима тренутну поставку система. Камера омогућава систему да детектује и идентификује пионе на интерактивном екрану. Као додатак, табла поседује и идентификацију путем радио фреквенције која у комбинацији са тагованим објектима може да се користи за учитавање и памћење различитих сценарија и игара. Коришћење STARS платформе у току игре, приказано је на слици 2.21.

За приказ јавних информација игре као што су резултат, мапа целог света игре или за додатне ефекте и анимације у случају игара са приповедањем користи се велики вертикални екран. Такође, сваки играч поседује и лични дигитални помоћник за приказ приватних информација. Сваки физички пион на табли може на пример да има свог виртуелног парњака на дигиталном помоћнику који ће водити евиденцију о јачини и здрављу карактера у игри или о прикупљеном инвентару. У случају тимских игара, чланови истог тима могу једни другима да шаљу поруке коришћењем личног дигиталног помоћника. Јак фокус је стављен и на пружање аудио канала за комуникацију са корисницима система. Корисници система могу да чују јавне гласовне поруке путем звучника или приватне поруке путем слушалица.

Слика 2.21 STARS платформа у току игре, преузето из [50]

Коришћењем предложене платформе развијене су и испробане различите игре као што су типична игра на табли „Монопол“ и комплекснија „Витез чаробњак“ (*Mage Knight*) игра. Осам различитих група девојчица узраста од 11 до 14 година су испробале развијене игре. Генерална опсервација је била да је главни циљ платформе, обезбеђивање забавне и директне интеракције између играча, постигнут. Играчи су били импресионирани богатством презентације укључујући визуелни квалитет и аудио излаз. Оно што је потцењено је потребна стабилност система.

Дигитална игра картама која комбинује таблу осетљиву на додир и гестове коришћењем мобилних уређаја, названа *Poker Surface*, представљена је у [25]. Интеракције у игри могу да се реализују на два начина. Први начин је директним додиром табле (*Microsoft Surface*), а други начин је употребом мобилних телефона са акцелерометром, повезаних коришћењем *Bluetooth* конекције са таблом. Мобилни телефони се користе за препознавање гестова и приказ приватних информација играча (као што су карте у руци). Пример коришћења мобилних телефона за интеракцију са таблом је представљен на слици 2.22.

Слика 2.22 Пример апликације која интегрише интеракцију са мобилним уређајима и таблом осетљивом на додир, преузето из [25]

Површина интерактивне табле осетљиве на додир је искоришћена као основна површина за игру. Играчи могу да се сместе на све четири стране табле, где добијају део табле који представља њихов приватни простор. Додатно, сваки играч може да прикључи свој мобилни телефон и да га користи за детекцију гестова и као приватни екран. Гестови који су подржани коришћењем телефона су: подизање телефона вертикално што представља поглед у карте, ротирање телефона за 90° лево или десно уз дрмусање представља чекирање и спуштање телефона у хоризонталан положај представља затварање карата. Када користе само интерактивну таблу карте и чипови су приказани на табли и померају се додиром. Због видљивости карата играчи морају да их заклањају руком или папиром.

Систем је испробало 20 корисника који су били подељени у 7 група од по 3 играча. Сви учесници су од раније били упознати са правилима покера. Свака група је два пута играла игру, једном директно коришћењем табле, а други пут коришћењем мобилних телефона као играчке конзоле. Након завршетка учесници су попуњавали унапред припремљене упитнике оцењујући колико им је било тешко или лако да изведу поједине акције и колико им се свака интеракција свидела. Прелиминарни резултати су показали да иако је корисницима било теже

да користе мобилне уређаје за препознавање гестова, радије су их користили него директан додир табле.

Експеримент изведен како би се проучила употреба мобилних уређаја као контролера приликом једноставних интеракција са таблом осетљивом на додир представљен је у раду [48]. Извршено је поређење три различите форме интеракције: додир табле, коришћење телефона као опипљивог контролера на екрану осетљивом на додир и директном манипулацијом на екрану телефона. Изведен је закључак да употреба телефона као опипљивог контролера на екрану осетљивом на додир није увела побољшања у односу на директан додир табле, чак је и кашњење било веће. Као најприхватљивије решење показала се директна манипулација на екрану телефона.

## 2.3 Сценарији интеракције човек-рачунар у интегрисаном окружењу дневне собе

Интегрисано окружење дигиталних ТВ пријемника, мобилних уређаја и интернета је представљено на слици 2.23. Апликације развијене за дигиталне ТВ пријемнике имају могућност приступа интернету и подацима из дигиталног видео тока. Такође, апликације развијене за дигиталне ТВ пријемнике корисницима осим пасивног гледања телевизије пружају и интеракцију са ТВ програмом и са удаљеним пријатељима. Мобилни уређаји опремљени сензорима имају могућност препознавања различитих акција корисника, а осим тога поседују и екран који је могуће искористити за приказ додатних информација у односу на оне које се приказују на великом ТВ екрану. Када се поменуте особине и функционалности дигиталних ТВ пријемника и мобилних уређаја узму у обзир можемо идентификовати следеће сценарије интеракције човек-рачунар које апликације које се развијају у интегрисаном окружењу могу да пруже корисницима:

1. подела приказа на више екрана (на мобилним уређајима се приказује додатни садржај као пратећи ономе што је на великом ТВ екрану)
2. употреба мобилног уређаја за управљање акцијама на централном екрану
3. детекција покрета коришћењем сензора

4. додатне чулне повратне информације (хаптички одговор, светлосно обавештење)
5. пасивно гледање телевизије обогаћено паралелним интерактивним активностима
6. корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака
7. интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама


Слика 2.23 Интегрисано окружење дигиталних ТВ пријемника, мобилних уређаја и интернета

У табели 2.1 је дат преглед идентификованих сценарија интеракције човек-рачунар код постојећих система у области интеракције човек-рачунар у окружењу дневне собе са дигиталним ТВ пријемницима и мобилним уређајима. Прегледом постојећих система у области интеракције човек-рачунар у окружењу дневне собе са дигиталним ТВ пријемницима и мобилним уређајима можемо закључити да се већина система развија независно и да се фокусирају на појединачним апликацијама које корисницима по питању интеракције нуде занимљиве, али најчешће недовољно флексибилне системе, ограничене на тачно дефинисане сценарије употребе.

Табела 2.1 Преглед идентификованих сценарија интеракције човек-рачунар код постојећих система у области интеракције човек-рачунар у окружењу дневне собе

| **Област** | **Системи из литературе** | **Сценарији интеракције човек-рачунар** | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Социјална и интерактивна телевизија | *Amigo TV* [19] | | | | + | | | Један корисник интерреагује са једним удаљеним корисником |
| | *Connect TV* [13] | | | | + | | | Један корисник интерреагује са једним удаљеним корисником |
| | *CollaboraTV* [52] | | | | + | | | Један корисник интерреагује са једним удаљеним корисником |
| | *Motorola STV* [51] | | | | + | | + | Један корисник интерреагује са једним удаљеним корисником |
| Интерактивни системи који користе други екран | Додатни екран за управљање телевизијом [21] | + | + | | | + | | Један корисник |
| | TAMALLE [29] | + | | | + | | | Један корисник |
| | MVC-iTV [72] | + | + | | + | | | Група корисника смештена у истој просторију |
| | Додатни екран за приказ информација о ТВ програму [16] | + | | | | + | | Један корисник интерреагује са једним удаљеним корисником |

| Област | Системи из литературе | Сценарији интеракције човек-рачунар | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Препознавање покрета коришћењем мобилних уређаја | *PhonePointPen* [2] | | | + | | | | Један корисник |
| | *Grmobile* [39] | | | + | | | | Један корисник |
| | *MobiToss* [58] | | + | + | | | | Група корисника смештена у истој просторију |
| | Пецање и куглање [3] | | + | + | | | | Један корисник |
| | *Poppet* [69] | | + | + | | | | Група корисника смештена у истој просторију |
| | *BlueWave* [47] | + | + | | | | | Група корисника смештена у истој просторију |
| Дигитализација друштвених игара | *SmartPlaying Cards* [57] | + | | + | | | | Група корисника смештена у истој просторију |
| | STARS [50] | + | | + | | | | Група корисника смештена у истој просторију |
| | *Poker Surface* [25] | + | + | + | | | | Група корисника смештена у истој просторију |

Заједничка карактеристика система социјалне и интерактивне телевизије је персонализација ТВ уређаја са фокусом на ситуације у којима само један корисник седи испред телевизора. Изостављени су сценарији који би подстакли пријатеље и чланове породице да се физички окупе око телевизора и комуницирају лицем у лице. Ситуације у којима група људи седи испред једног телевизора и комуницира са групом људи на некој другој локацији најчешће нису узете у обзир. Слична ситуација је и код интерактивних система који користе други екран. Већина система, осим MVC-iTV , не узима у обзир да је могуће да више људи истовремено управља садржајем који се приказује на ТВ екрану. Такође, могућност да се мобилни уређаји истовремено користе као додатни екрани и као контролери са могућношћу препознавања покрета корисника није

узета у обзир. Фокус постојећих система је углавном на једном од ова два поменута сценарија.

## 2.4 Предлог решења

Како би се омогућио ефикасан и униформан развој дистрибуираних апликација у интегрисаном окружењу дигиталних ТВ пријемника, мобилних уређаја и интернета предложена је SHARP (*SHARed sPace*) софтверска платформа. Ова софтверска платформа треба да омогући развој апликација које корисницима нуде идентификоване сценарије интеракције човек-рачунар у интегрисаном окружењу. Употреба SHARP софтверске платформе би требала да омогући да се у оквиру једне апликације корисницима пружи више различитих сценарија интеракције. Након идентификације сценарија интеракције потребно је одабрати и тип апликација које ће бити развијене употребом SHARP софтверске платформе.

Због своје популарности, масовне употребе и тржишне исплативости SHARP софтверска платформа би могла да се примени за развој игара које би се играле у интегрисаном окружењу нудећи корисницима идентификоване сценарије употребе. Истраживање из 2014. године показује да 2,97 билиона људи у свету има приступ интернету, а да 60% њих игра игре. Укупан профит који је остварен у 2014 години од игара износи 82 билиона долара [71]. Играње игара представља велики део активности људи који имају приступ интернету, превазилазећи полне границе и границе старости.

Са развојем мобилних уређаја и порастом њихове популарности људи све више времена проводе користећи мобилне апликације које представљају игре. У 2012. години људи су у просеку проводили око 1 сат и 20 минута дневно играјући игре коришћењем мобилног телефона, да би у 2014. години време порасло на више од два сата дневно у просеку. Од јануара до марта 2014. године корисници су 32% укупног времена које су провели коришћењем мобилних апликација провели на играње игара. Након игара, највећу популарност имале су и друштвене мреже са 29% укупног времена проведеног коришћењем мобилних апликација (17% на *Facebook* друштвеној мрежи, 1,5% на *Twitter* друштвеној мрежи и 9,5% на осталим друштвеним мрежама) [62]. Истраживање [24] као најпопуларније место

на ком се играју мобилне игре наводи кауч у дневној соби (69% људи), а 41% људи који користе мобилне игре их игра упоредо са гледањем телевизије.

Дигитализација друштвених игара које се могу играти у оквиру дневне собе се показала као добар избор јер омогућава очување интеракција лицем у лице, а пружа богату визуелну презентацију дигиталног света игре. Сва предложена решења дигитализације друштвених игара поседују наменски хардвер који је јако скуп и тежак за премештање. Интегрисано окружење би могло да се искористити као окружење за развој традиционалних друштвених игара као што су игре картама и игре на табли које би се извршавале на дигиталним ТВ пријемницима и мобилним уређајима без потребе за додатним наменским хардвером нудећи корисницима нове видове интеракције.

Развијене игре би корисницима могле понудити идентификоване сценарије интеракције на следећи начин. Главни садржај игре, као што је табла, отворене карте или тркачка стаза може да се прикаже на ТВ екрану. Екрани мобилних уређаја могу да се искористе за приказ приватних информација сваког играча, као што су карте у рукама, прикупљени инвентар и сл. Мобилни уређаји опремљени екраном осетљивим на додир, акцелерометром и магнетометром могу да се искористе и као контролери игре. Игра може да се обогати подацима који су обрађени из дигиталног видео тока, као што је нпр. електронски програмски водич који би се користио у току игре. Такође, игра може и да се прикаже и на графичком слоју преко емитованог телевизијског тока. Приступ интернету би могао да обезбеди да се игра са више играча прошири изван једне дневне собе и да омогући и играчима на другим локацијама да буду укључени у игру. Ово би омогућило тимску игру где би на пример припадници истог тима седели у истој соби. Додатно би се могла обавити и интеграција са социјалним мрежама и приступ јавној бази филмова.

Верификација предложене софтверске платформе се може извршити поређењем апликација које су развијене коришћењем и без коришћења предложене платформе. Метрике које се могу искористити за поређење различитих имплементација игара су: величина кода односно број физичких линија кода, цикломатска комплексност, густина комплексности и време одзива. Број физичких линија кода, цикломатска комплексност и густина комплексности

представљају основне метрике за процену времена израде и одржавања програмског кода [45]. Број физичких линија кода представља индикацију времена и труда потребног за развој програма [54]. Ова метрика је један од главних улаза за већину модела који служе за процену трошкова израде [12]. Цикломатска комплексност [49] се дефинише као број линеарно независних путања кроз програмски модул и индикатор је броја тестова који су потребни за евалуацију исправности модула. Густина комплексности [34] се дефинише као количник цикломатске комплексности модула и њене дужине изражене у физичким линијама кода. Ова метрика представља нормализовану комплексност секције кода и показатељ је потешкоћа одржавања. У области интеракције човек-рачунар време одзива апликације је од критичног значаја и од велике је важности да развијена софтверска платформа не уноси велика додатна кашњења у свеукупном одзиву, што би као последицу могло имати негативно искуство корисника [59].

# 3. SHARP СОФТВЕРСКА ПЛАТФОРМА ЗА РАЗВОЈ АПЛИКАЦИЈА У ИНТЕГРИСАНОМ ОКРУЖЕЊУ

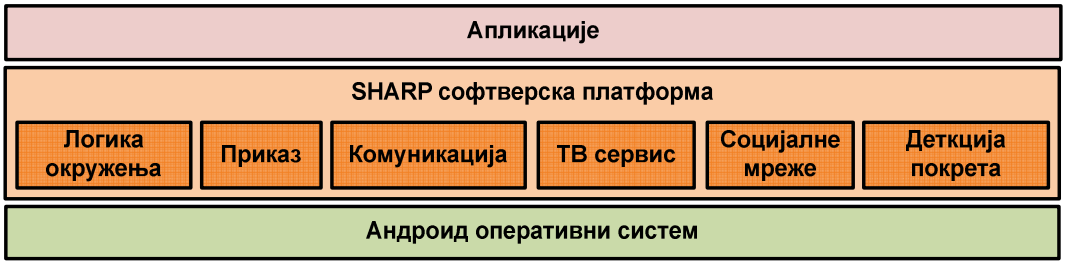
У овој глави биће представљена имплементација предложене SHARP софтверске платформе која омогућава развој дистрибуираних апликација у интегрисаном окружењу дигиталних ТВ пријемника, мобилних уређаја и интернета. Најпре ће се приказати структура SHARP софтверске платформе и представити алати изабрани за реализацију. Након тога ће реализација сваке компоненте софтверске платформе посебно бити описана. На крају ће бити приказана размена информација између различитих компонената платформе.

## 3.1 Структура SHARP софтверске платформе и избор алата за реализацију

SHARP софтверска платформа се састоји од шест компоненти: „Логика окружења“, „Приказ“, „Комуникација“, „ТВ сервис“, „Друштвени медији“ и „Детекција покрета“ [56]. Компонента „Логика окружења“ описује главну логику виртуелног окружења, које називамо дељени простор, у које људи могу да се укључе у циљу међусобне интеракције и интеракције са окружењем. Поглед на виртуелно окружење, односно исцртавање елемената дељеног простора на екранима дигиталних ТВ пријемника и мобилних уређаја обезбеђује компонента „Приказ“. Компонента „Комуникација“ обезбеђује размену информација између уређаја и сервиса који учествују у истом дељеном простору. Главна улога компоненте „ТВ сервис“ је да омогући омотач око већ постојећих Јава АПИ-ја (*Application Programming Interface*) развијених у циљу интеграције дигиталне телевизије у модерне дигиталне ТВ пријемнике [70]. Компонента „Друштвени медији“ представља интерфејс ка платформама познатих друштвених медија. Препознавање покрета корисника употребом уређаја који поседује акцелерометре и магнетометре је реализовано у оквиру компоненте „Препознавање покрета“.

Одабрано је да SHARP софтверска платформа буде реализована на уређајима са Андроид оперативним системом [1]. Из године у годину број уређаја коју поседују Андроид оперативни систем све више расте. У 2014. години 80% тржишта мобилних уређаја су представљали Андроид уређаји [60], а за њима следе мобилни уређаји са *Apple*-овим iOS оперативним системом који су обухватили око 15% тржишта. Такође, у последње време све је више и дигиталних ТВ пријемника који су опремљени истим оперативним системима као и мобилни уређаји (као што су Андроид и *Apple*-ов оперативни систем). Њихова распрострањеност и веза са интернетом омогућавају једноставан развој дистрибуираних апликација у интегрисаном окружењу.

Како би добили приступ за iOS SDK (*Software Development Kit*) потребно је да се плати годишња регистрација и право да се објаве апликације. Такође, iOS SDK је доступан само за *Mac* платформу. Андроид представља платформу отвореног кода, што значи да свако може бесплатно да приступи Андроид оперативном систему и бесплатно користи Андроид SDK. Андроид SDK је такође доступан за све платформе. Апликације које се развијају за Андроид платформу се пишу коришћењем Јава програмског језика и могуће их је бесплатно дистрибуирати. Главни разлози избора Андроид оперативног система за развој SHARP платформе су били његова велика распрострањеност, отвореност кода и једноставност при дистрибуцији апликација. Структура SHARP софтверске платформе и позиција у оквиру софтверског стека на уређајима са Андроид оперативним системом је представљена на слици 3.1.




Слика 3.1 Структура SHARP софтверске платформе и позиција у оквиру софтверског стека на уређајима са Андроид оперативним системом

## 3.2 Реализација SHARP софтверске платформе

У овом поглављу ће бити приказана реализација сваке од шест компоненти SHARP софтверске платформе: „Логика окружења“, „Приказ“, „Комуникација“, „ТВ сервис“, „Друштвени медији“ и „Детекција покрета“. Након тога ће бити објашњен начин на који се врши размена информација између дистрибуираних компоненти платформе.

### 3.2.1 Компонента „Логика окружења“

У оквиру компоненте „Логика окружења“ дефинише се дељени простор и дозвољене интеракције у њему. Како би се у оквиру дељеног простора омогућио развој различитих типова игара у оквиру ове компоненте се као подскуп дељеног простора дефинише свет игре и правила која важе у игри. Такође, дефинише се и хијерархија типова игара које је могуће реализовати и за сваки тип игре су издвојене њене специфичности. На слици 3.2 је дат UML (*Unified Modeling Language*) класни дијаграм на ком су представљене главне класе компоненте „Логика окружења“, као и њихове зависности. *SharedSpace* класа је апстрактна базна класа која дефинише дводимензионални дељени простор са којим учесници интерреагују и дефинише правила интеракције у дељеном простору. Промене стања у дељеном простору се изражавају у итерацијама стварајући привид константног протока времена. Итерације представљају мале јединице времена које конфигурише програмер. Класа *SharedSpace* садржи референце на три типа објеката: инстанцу класе *Participant*, инстанцу класе *Element* и инстанцу класе *Action*. Инстанца класе *SharedSpace* као и инстанце класа које су са њом повезане су најчешће смештене на серверу.

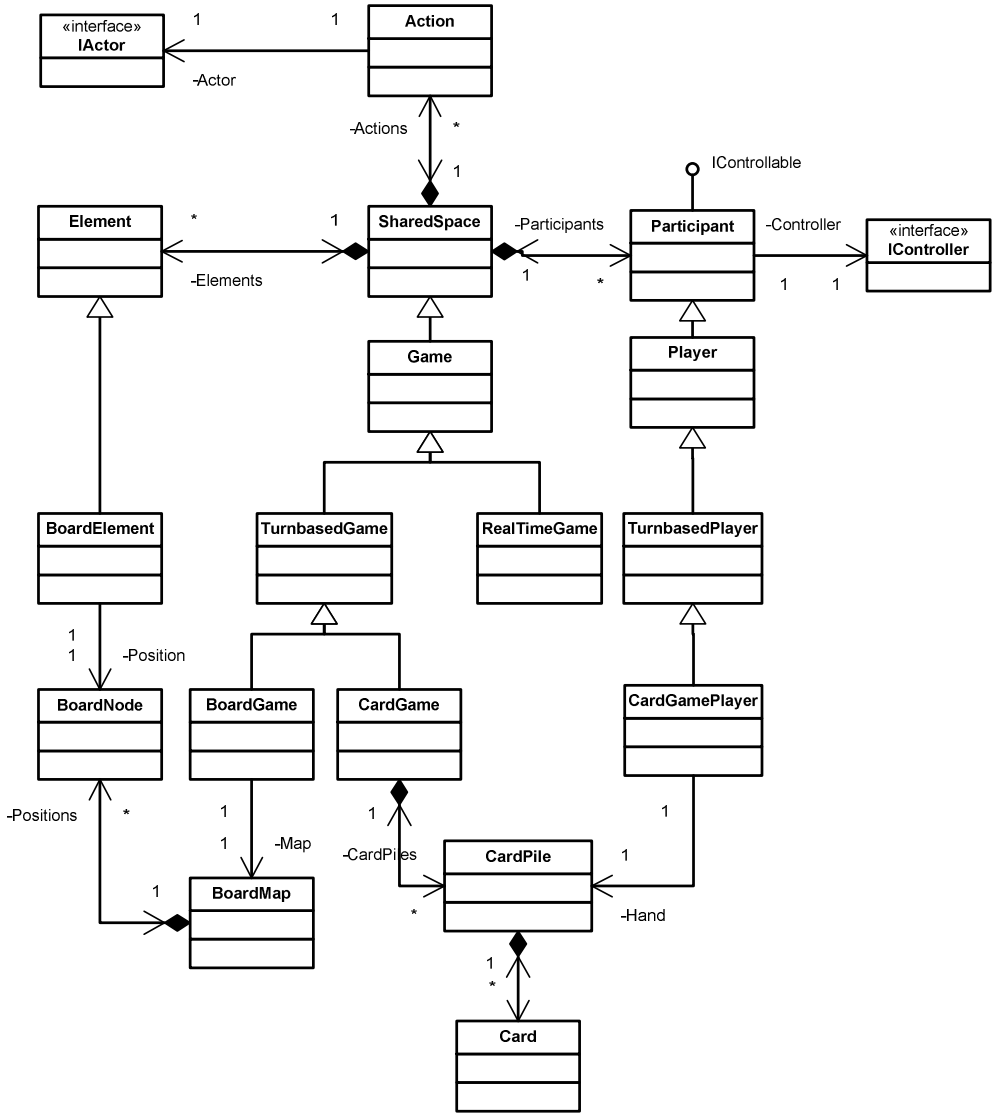
Свака инстанца класе *Participant* представља особу која има свој идентитет у виду имена, година, пола и сл. Објекат *Participant* има могућност да успостави двосмерну комуникацију са логиком контролера. Логика контролера представља део кода који се извршава на клијентској апликацији која је најчешће смештена на мобилном уређају и задужена за прикупљање улазних акција, као што је додир екрана и препознавање покрета. Након детекције акције учесника објекту *Participant* се шаљу информације о акцији коју је потребно предузети. Објекат *Participant* такође шаље поруке логици контролера. Ове поруке се могу

искористити као окидач за хаптичку повратну информацију на мобилном уређају. Пример хаптичке повратне информације у случају игара би била вибрација уређаја играча који је на потезу. Инстанца класе *Element* дефинише објекат који има своје координате и димензије у дељеном простору, а инстанца класе *Action* представља акцију која утиче на стање дељеног простора одређени временски период. У случају игара пример елемента може бити пион на табли, а пример акције померање пиона са једне позиције на табли на другу. Крај и исход акције се сигнализирају позивом методе инстанце класе која имплементира интерфејс *IActor*.

*Game* класа је изведена из *SharedSpace* класе и дефинише дводимензионални свет игре, као и базична правила која одређују понашање игара. Класа *Player* је изведена из класе *Participant* и представља играча који осим идентитета има и број освојених поена. Платформа подржава развој потезних игара и игара у реалном времену. Потезне игре могу бити игре на табли и игре картама. Класе *Game*, *Player* и *Element* се наслеђивањем проширују и формирају хијерархију подржаних категорија игара које се могу играти у интегрисаном окружењу. Класе *TurnBasedGame* и *TurnbasedPlayer* су апстрактне подкласе класа *Game* и *Player* респективно и заједно описују потезну игру и рунде. Када започне потезна игра, односно када почне први круг игре, потез се предаје првом регистрованом *TurnbasedPlayer* објекту у игри. Када играч заврши свој потез, потез се предаје следећем регистрованом објекту *TurnbasedPlayer* и предаја потеза се даље наставља док сви играчи не заврше. Када у првом кругу сви играчи заврше свој потез завршава се рунда и процес се понавља из почетка. Инстанца класе *RealTimeGame* описује игру у реалном времену у којој се за разлику од потезних игара свим играчима дозвољава да истовремено контролишу игру.

Игре на табли су представљене класом *BoardGame*, која је апстрактна подкласа класе *TurnbasedGame*. Класа *BoardMap* дефинише мапу табле са дискретним позицијама и дозвољеним прелазима из једне позиције у другу. Свака позиција на табли је представљена инстанцом класе *BoardNode*. Ова класа поседује методу која проналази могуће путеве између две позиције. Елементи дељеног простора који могу да заузму неку од дискретних позиција на табли су описани класом *BoardElement*, која је апстрактна подкласа класе *Element*. Типичан

пример елемента који може да заузме неку од дискретних позиција је пион који може да се помера по табли. Класа *BoardElement* се користи заједно са класом *BoardGame*. Игра картама је представљена класом *CardGame*, која је апстрактна подкласа класе *TurnbasedGame*. Скуп карата као што је шпил карата, карте у руци или карте на столу су представљене класом *CardPile*.




Слика 3.2 Класни дијаграм компоненте „Логика окружења“

### 3.2.2 Компонента „Приказ“

Компонента „Приказ“ је одговорна за исцртавање елемената дељеног простора на екранима ТВ пријемника и мобилних уређаја. У току једне игре инстанце компоненте „Приказ“ се могу налазити на више различитих екрана и могу приказивати различите слике. На пример, на великом екрану ТВ пријемника се приказују елементи који су јавни за све, а на екранима мобилних уређаја само елементи који су приватни. Инстанца компоненте „Приказ“ чува референце само на елементе који ће се приказивати на екрану којем припада. За ажурирање слике је одговорна компонента „Логика окружења“ која приликом промена у дељеном простору компоненти „Приказ“ шаље минималан скуп података потребан да би се слика ажурирала.

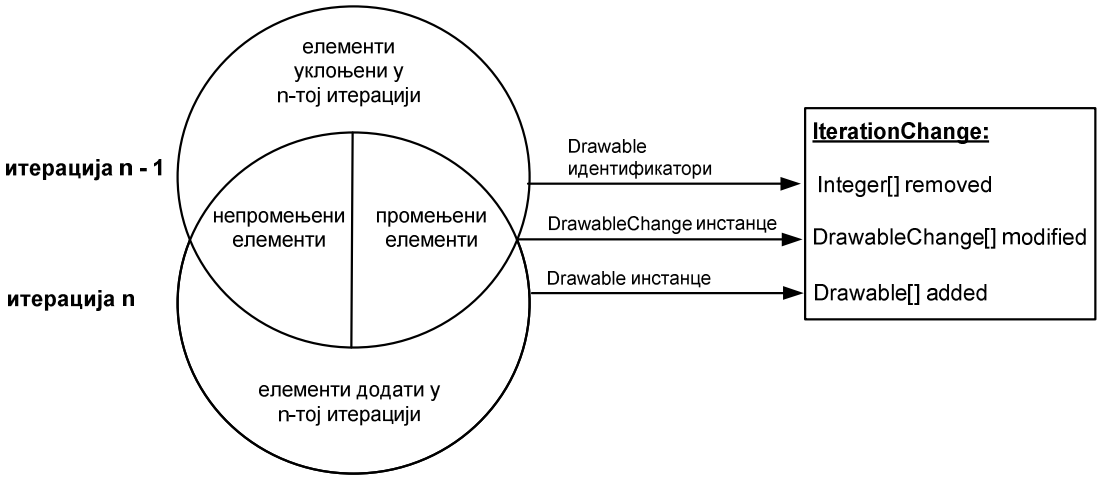
#### 3.2.2.1 Утицај промена у дељеном простору на исцртавање графике

Елемент који може да се исцрта на екрану се чува у оквиру инстанце класе *Drawable* и садржи атрибуте класе *Element* који су битни за исцртавање. Атрибути могу бити променљиви (нпр. позиција и стање) или непроменљиви (нпр. идентификатор *Element* објекта). *Drawable* објекте производи компонента „Логика окружења“, а компонента „Приказ“ их користи. Свака инстанца компоненте „Приказ“ чува листу инстанци *Drawable* класа које одговарају објектима који се приказују на том екрану. Приликом итерација у дељеном простору долази до промена стања *Element* објеката. Ове промене се одражавају и на стање *Drawable* објеката које користи компонента „Приказ“. Један од начина да се ажурира стање компоненте „Приказ“ је да се након сваке итерације овој компоненти шаљу сви *Drawable* објекти које треба да прикаже. Други, много економичнији начин, коришћен у овом раду, је да се компонента „Приказ“ обавештава само о променама које су се између две итерације догодиле у листи *Drawable* објеката, искључујући могући велики број објеката који су између две итерације остали непромењени.

Промене које могу да се догоде у дељеном простору између две итерације су: додавање, уклањање и промена *Element* објеката. Ове врсте промена одговарају додавању, уклањању и променама *Drawable* објеката у оквиру инстанце компоненте „Приказ“. *IterationChange* класа садржи информације које се са

компоненте „Логика окружења“ прослеђују компоненти „Приказ“, а компонента „Приказ“ их користи за ажурирање слике. Компонента „Приказ“ користи информације које добија од *IterationChange* објекта како би ажурирала листу својих *Drawable* објеката. У току сваке итерације у игри креира се по једна инстанца *IterationChange* класе и прослеђује компоненти „Приказ“.

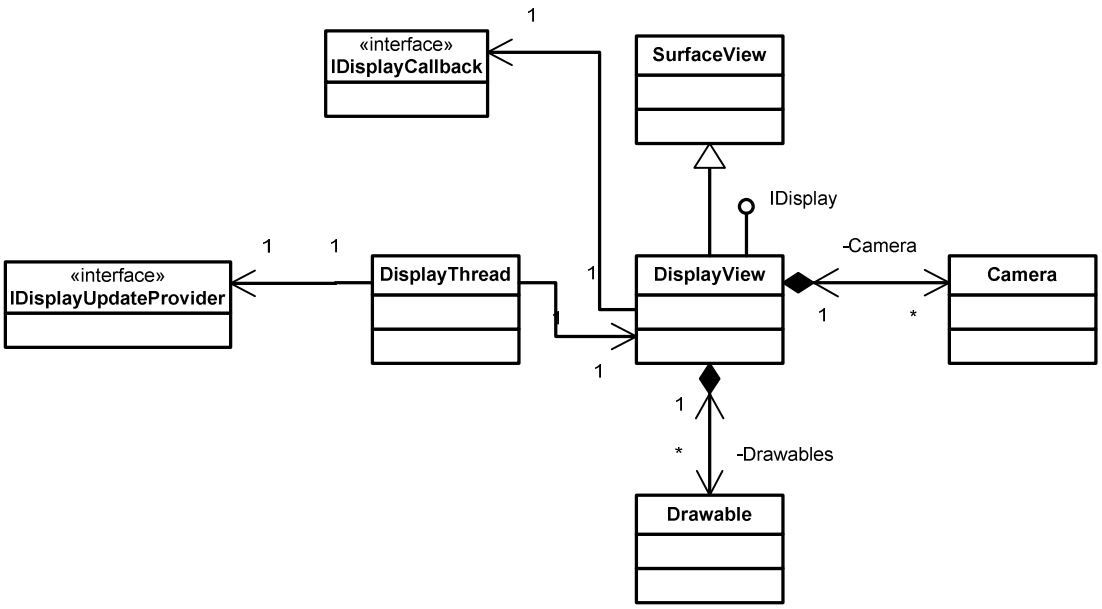
На слици 3.3 је представљено креирање *IterationChange* објекта. Ако је у дељеном простору креиран нови елемент, инстанци компоненте „Приказ“ је потребно додати нови *Drawable* објекат. Уколико је неки елемент уклоњен из дељеног простора, компоненти „Приказ“ се доставља листа идентификатора уклоњених елемената. У случају када се постојећи елемент промени између две итерације, компоненти „Приказ“ се достављају само променљиви атрибути класе *Drawable* који се чувају у оквиру *DrawableChange* класе. Класа *Drawable* имплементира метод који дозвољава њеним променљивим атрибутима да буду ажурирани инстанцом класе *DrawableChange*. У случају када се компоненте „Приказ“ и „Логика окружења“ налазе на различитим уређајима инстанце *IterationChange* класе се могу серијализовати и проследити као JSON (*JavaScript Object Notation*) поруке.




Слика 3.3 Креирање *IterationChange* објекта

#### 3.2.2.2 Исцртавање графике у оквиру компоненте „Приказ“

На слици 3.4 је представљен UML класни дијаграм најважнијих класа компоненте „Приказ“. Главна класа компоненте „Приказ“ је класа *DisplayView* и представља подкласу постојеће *SurfaceView* класе Андроид оперативног система. *DisplayView* класа је одговорна за исцртавање објеката на екрану и садржи листу *Drawable* објеката које је потребно приказати. Класа *DisplayThread* је задужена за периодично дохватање *IterationChange* објеката које прослеђује *DisplayView* класи. Када прими *IterationChange* објекат *DisplayView* класа врши ажурирање своје листе *Drawable* објеката и након тога активира исцртавање. Исцртавање графике се обавља коришћењем стандардних Андроид примитива које омогућавају да се графика прикаже на слоју преко емитованог ТВ програма. Такође, обезбеђено је да се графика исцртава по нетранспарентној позадини у случају када се не користи компонента “ТВ сервис“.




Слика 3.4 Класни дијаграм компоненте „Приказ“

Класа *DisplayView* садржи референцу на *Camera* објекат који дефинише поглед на дељени простор кроз правоугаоник који има своју позицију, висину и ширину. *Camera* је одговорна за процес приказивања координата дељеног простора на екрану уређаја и подржава два режима рада. У првом режиму рада

цео дељени простор се приказује на екрану и скалира се према димензијама екрана. Овај режим се најчешће користи код јавних екрана. У другом режиму рада се приказује само део дељеног простора који прати одабрани објекат. Објекат који се прати остаје у центру правоугаоника уколико му се координате у дељеном простору не мењају. Када објекат промени позицију тако да пређе дефинисане маргине, прилагођава се и позиција правоугаоника у дељеном простору како би објекат сачувао минимално растојање од ивице екрана.

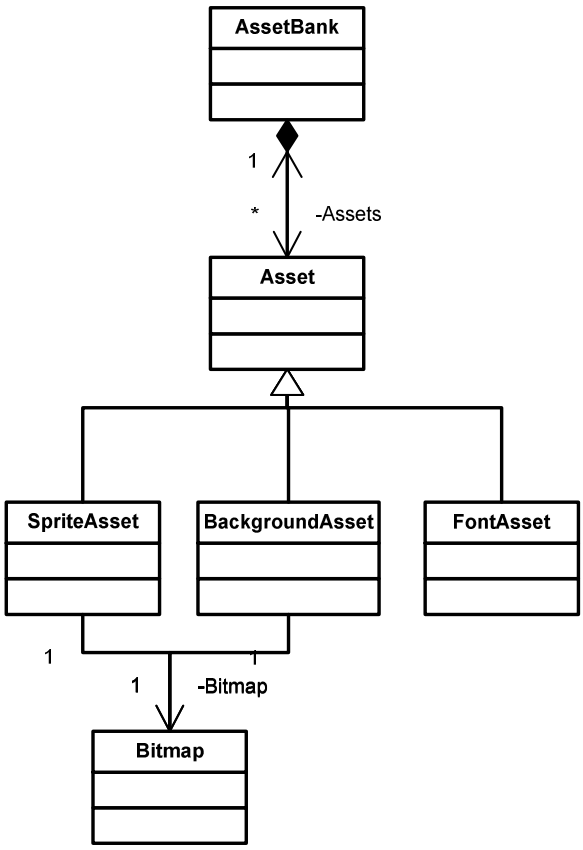
Могуће је дефинисати и објекте који ће се увек налазити на одређеном месту на екрану без обзира који део мапе игре се приказује на екрану. Пример елемента који се не помера без обзира на који део мапе показује камера у случају игара је резултат играча. Компонента „Приказ“ региструје догађаје додира, даје информације о координатама где се додир догодио и даје листу свих идентификатора елемената који се налазе на тој позицији. Када региструје догађај компонента „Приказ“ позива инстанцу класе која имплементира *IDisplayCallback* интерфејс.

#### 3.2.2.3 Управљање графичким репрезентацијама

На UML класном дијаграму на слици 3.5 представљени су различити типови графичких репрезентација који се користе приликом исцртавања дељеног простора. Класа *AssetBank* је Синглетон класа која служи за складиштење графичких репрезентација. Приликом иницијализације компоненте „Приказ“ *AssetBank* се попуњава скалираним графичким репрезентацијама које одговарају резолуцији екрана на ком ће се приказивати. Класа *Asset* је апстрактна базна класа која има јединствен идентификатор и може се проширити тако да образује једну од следеће различите класе: *SpriteAsset*, *BackgroundAsset* или *FontAsset*.

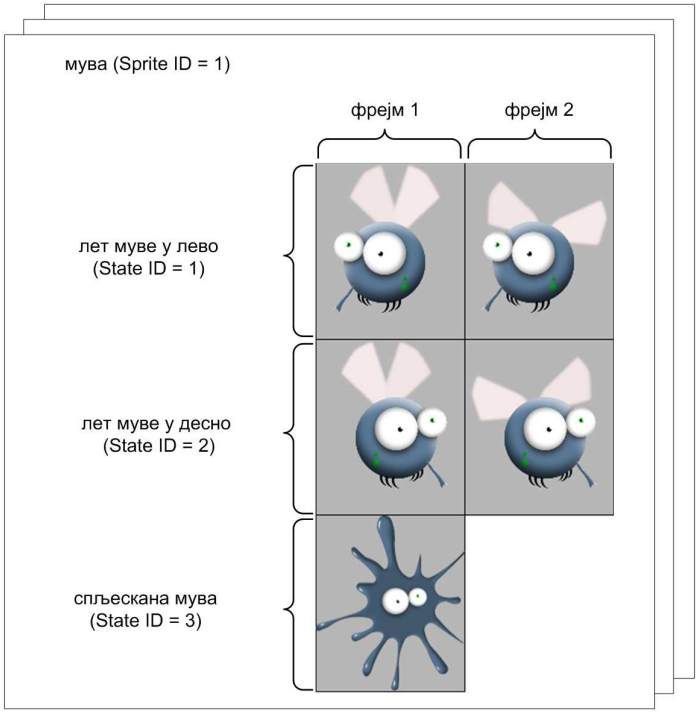
Класа *SpriteAsset* представља репрезентацију једног објеката дељеног простора. Скуп слика које образују анимације представљају инстанцу *SpriteAsset* класе и груписане су по идентификатору стања. Ова класа садржи референце на *Bitmap* објекте који дефинишу слике које се користе приликом анимација. Једна група слика представља анимацију једног стања објекта. Група може да садржи и само једну слику. На слици 3.6 је дат пример у ком су приказане све графичке репрезентације муве које се користе приликом њене анимације. Све графичке репрезентације муве имају јединствен идентификатор (*SpriteID*). Стања у којима

се мува може наћи су: мува која лети у леву страну, мува која лети у десну страну и мува која је спљоштена. Идентификатор стања (*StateID*) се мења у зависности од тога у ком стању се мува налази. Докле год је мува у једном стању при свакој итерацији смењују се графичке репрезентације које одговарају том стању.




Слика 3.5 Класни дијаграм управљања графичким репрезентацијама

Класа *BackgroundAsset* поседује референцу на битмап који се користи за презентацију позадине дељеног простора. Позадина може постојати или може бити и транспарентна. Андроид оперативни систем поседује уграђене функционалности за управљање текстом којима се приступа преко класе *Paint*. Класа *FontAsset* представља омотач око класе *Paint*.

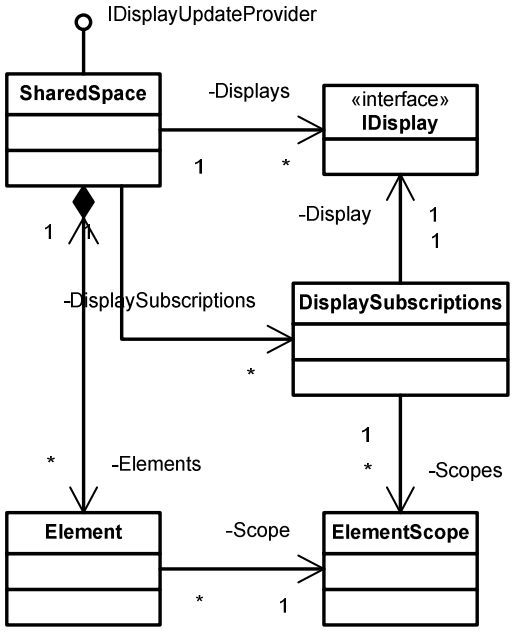



Слика 3.6 Графичке репрезентације муве која може да лети у леву, десну страну или да буде спљоштена

#### 3.2.2.4 Област приказа

Како би се дефинисала видљивост објеката који се приказују на различитим екранима користи се област приказа. На пример, елемент ће можда бити видљив само на приватном екрану једног учесника, а на осталим екранима се неће приказивати. На слици 3.7 је представљен класни дијаграм најважнијих класа које се користе приликом дефинисања области приказа. Области приказа су представљене класом *ElementScope*. Када се креира инстанца класе *Element* потребно је назначити којој области приказа припада. Елемент може да припада само једној области приказа, а једном екрану се може придружити више области приказа. Информације о томе ком екрану је која област приказа придружена се чувају у оквиру класе *DisplaySubscriptions*. Ако је нека област приказа придружена екрану, на том екрану се приказују сви елементи који припадају тој области приказа. Пре него што се *IterationChange* објекат проследи свакој инстанци компоненте „Приказ“ врши се филтрирање базирано на областима

приказа којима одговарајућа инстанца компоненте „Приказ“ припада. Резултујући *IterationChange* објекат ће садржати само информације о одговарајућим елементима које је потребно додати, променити или обрисати на том екрану.




Слика 3.7 Дијаграм најважнијих класа приликом дефинисања области приказа

Примена области приказа ће бити објашњена на примеру игре на табли која се игра са четири играча распоређена у два тима, односно по два играча у сваком тиму. Игра се састоји од померања пиона по табли након бацања коцкица и постављања мина по табли које су видљиве само за чланове истог тима, док чланови супротног тиме не знају где се налазе. Укупно се користи 5 екрана, један велики јавни екран и по један мали приватни екран за сваког од играча. У табели 3.1 је приказано осам могућих области приказа: универзална, приватна играча 1, приватна играча 2, приватна играча 3, приватна играча 4, приватна тима 1, приватна тима 2 и јавна. За сваку област приказа дати су примери елемената који јој припадају. Пиони и експлозије припадају универзалној области приказа. Инвентар играча припада одговарајућој приватној области приказа тог играча.

Мине постављене од стране једног члана тима припадају одговарајућој области приказа тог тима. Резултат игре припада само јавној области приказа.

Табела 3.1 Области приказа на примеру игре на табли са више играча

| **Област приказа** | **Примери елемената** |
|---|---|
| универзална | пиони, експлозија |
| приватна играча 1 | инвентар играча 1 |
| приватна играча 2 | инвентар играча 2 |
| приватна играча 3 | инвентар играча 3 |
| приватна играча 4 | инвентар играча 4 |
| приватна тима 1 | мине тима 1 |
| приватна тима 2 | мине тима 2 |
| јавна | резултат |

Табела 3.2 Коришћени екрани, области приказа које су им придружене и елементи видљиви на сваком екрану појединачно

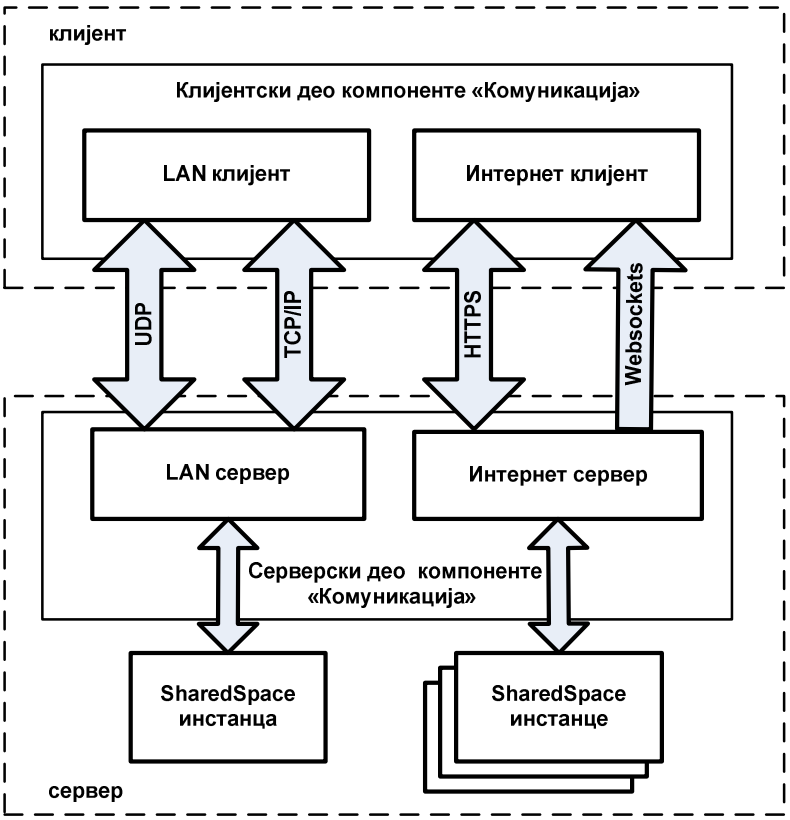
| **Екран** | **Области приказа придружене екрану** | **Елементи видљиви на екрану** |
|---|---|---|
| јавни | {универзална, јавна } | {пиони, експлозија, резултат} |
| играча 1 | {универзална, приватна играча 1, приватна тима 1} | {пиони, експлозија, инвентар играча 1, Мине тима1} |
| играча 2 | {универзална, приватна играча 2, приватна тима 1} | {пиони, експлозија, инвентар играча 2, мине тима1} |
| играча 3 | {универзална, приватна играча 3, приватна тима 2} | {пиони, експлозија, инвентар играча 3, мине тима2 } |
| играча 4 | {универзална, приватна играча 4, приватна тима 2} | {пиони, експлозија, инвентар играча 4, мине тима2 } |

За сваки од пет постојећих екрана, области приказа које су им придружене и елементи видљиви на том екрану су дати у табели 3.2. На пример, свим екранима је придружена универзална област приказа, па се пиони и експлозије приказују на сваком од пет постојећих екрана. Једино екрану играча 1 и играча 2 је додељена приватна област приказа тима 1, што значи да ће мине које постави један од та два играча бити видљиве и на екрану другог.

### 3.2.3 Компонента „Комуникација“

Компонента „Комуникација“ омогућава да више корисника активно учествује и утиче на промене у дељеном простору. Корисници могу да се налазе у оквиру исте дневне собе и да интерреагују са корисницима који су на удаљеним локацијама. У случају игара, ова компонента омогућава развој игара са више играча. Одабрано је да компонента „Комуникација“ омогући развој апликација коришћењем клијент-сервер архитектуре која пружа већу контролу над игром у односу на *peer-to-peer* архитектуру. Такође, сва израчунавања везана за логику игре ће се обављати на серверу, док ће се исцртавање објеката игре обављати на самом уређају. Структура компоненте „Комуникација“ је приказана на слици 3.8. Компонента се састоји од клијентског и серверског дела који пружају могућност за два начина имплементације комуникације. Један од начина је комуникација у оквиру локалне кућне мреже на коју су повезани сервер и сви клијенти, а други је коришћењем интернет сервиса са којим могу да комуницирају сви клијенти независно од своје локације.

Комуникација у оквиру локалне кућне мреже је имплементирана коришћењем TCP/IP (*Transmission Control Protocol/Internet Protocol*) сокета уз додатак аутоматског проналажења сервера. Клијент иницира UDP (*User Datagram Protocol*) мултикаст поруку на коју сервер у оквиру локалне кућне мреже може да одговори. Када корисник постане свестан свих сервера на мрежи отвара се TCP/IP сокет конекција према одабраном серверу. Након успостављања везе подаци могу да се размењују између клијента и сервера. На серверу је у овом случају у једном тренутку активна само једна инстанца конкретне имплементације дељеног простора.




Слика 3.8 Структура компоненте „Комуникација“

Комуникација са интернет сервисом је имплементирана коришћењем HTTP протокола (*Hypertext Transfer Protocol*) [8] и *WebSocket* протокола [31]. Ова компонента обезбеђује програмеру да на једноставан начин барата са захтевима и одговорима између клијента и сервера. Компонента управља различитим типовима имплементација дељеног простора, као и различитим инстанцама исте апликације. На примеру игара компонента управља различитим партијама исте игре, водећи рачуна о креирању и завршетку сваке партије, као и о сваком клијенту који се придружи игри или је напусти.

HTTP протокол представља основну и најчешће коришћену методу преноса података преко интернета, па је његова имплементација омогућена на великом броју различитих уређаја. *WebSocket* протокол имплементира двосмерну везу са сервером, па осим могућности слања захтева серверу од стране клијента пружа и могућност да сервер без експлицитног захтева шаље клијенту податке (*push*

*technology*). Овакав вид комуникације омогућава велику уштеду у перформансама јер се уместо честих упита корисника о промени стања врши само једна акција слања података од стране сервера. Иако овај протокол у односу на HTTP нуди боље перформансе и већу ефикасност, ради се о новијој технологији, па не постоји потпуна подршка за велики број уређаја. Због оваквих карактеристика одлучено је да у оквиру компоненте „Комуникација“ оба протокола буду подржана, при чему би се прво покушала употреба *WebSocket* протокола, а у случају неуспеха користио би се HTTP протокол.

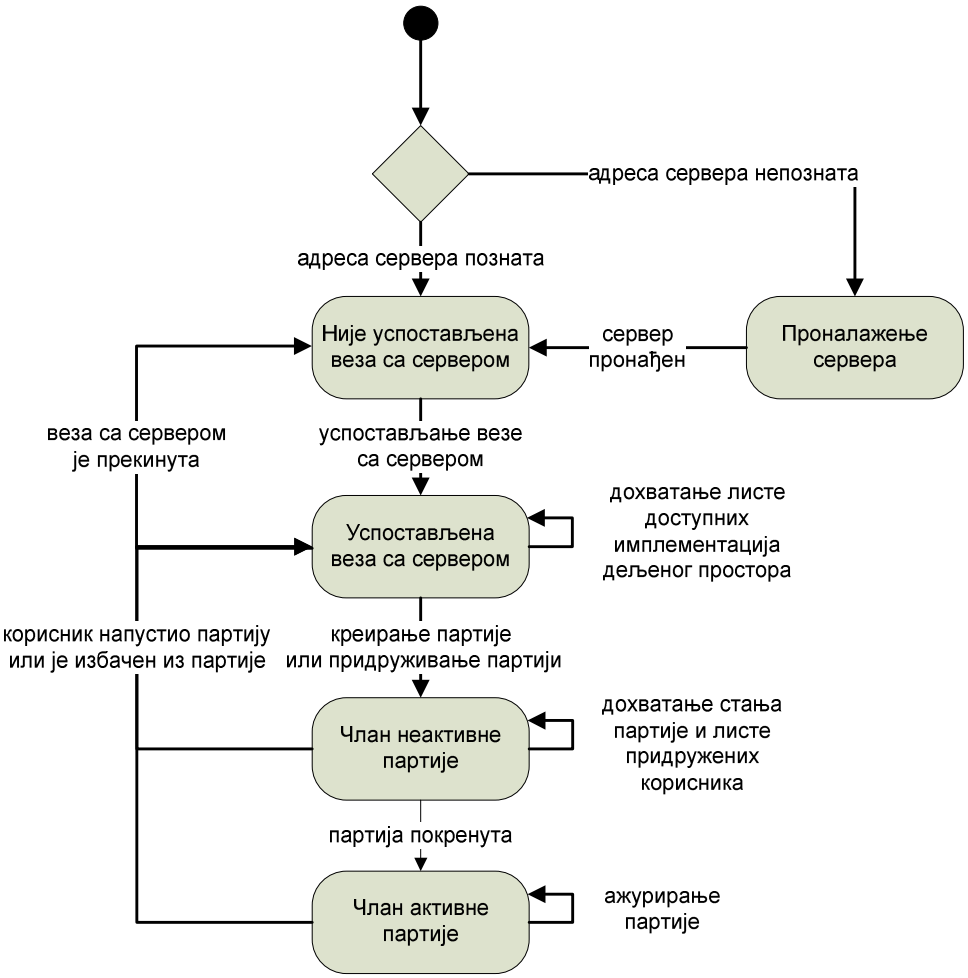
Приликом избора најпогоднијих технологија за развој сервера посебна пажња је била усмерена на технологије које ће својим функционалностима обезбедити флексибилност, проширивост и стабилност система а својим приступачним дизајном убрзати сам процес имплементације. Као један од најпопуларнијих Јава бесплатних сервера који поседује детаљну документацију, отвореност кода и добру подршку изабран је JBOSS 7 (*JavaBeans Open Source Software Application Server*) апликативни сервер [37]. Овај сервер такође поседује и велики број додатака од који су коришћени: *Infinispan* кеш, *Hibernate* библиотека, EJB 3.0 (*Enterprise JavaBeans*) и RESTEasy оквир. За имплементацију *WebSocket* протокола на серверској страни коришћен је *Netty* сервер [53], који је могуће поставити као додатак (*plug-in*) на JBOSS сервер. Подаци који се преносе су у JSON формату [22].

#### 3.2.3.1 Реализација клијентског дела компоненте „Комуникација“

У оквиру клијентског дела компоненте „Комуникација“ енкапсулиране су све функционалности потребне за комуникацију са сервером, без обзира на то који вид комуникације је коришћен. Приликом креирања конкретне апликације потребно је са клијентске стране компоненте „Комуникација“ обезбедити: надимак корисника, тип коришћене комуникације, опционо IP адресу или URL (*Uniform Resource Locator*) сервера. Надимак корисника ће бити коришћен у циљу представљања корисника у оквиру дељеног простора без обзира на то коју апликацију корисник покреће. У случају када апликација која се покреће користи комуникацију у оквиру локалне кућне мреже покушаће се са успостављањем везе са сервером чија је IP адреса унета. Ако IP адреса сервера није доступна почеће се аутоматска претрага сервера. Када покренута апликација комуницира са интернет

сервисом, потребно је доставити URL интернет сервиса, а аутоматско проналажење у том случају није могуће.

Главна класа клијентског дела компоненте „Комуникација“ је *SharedSpaceComClient*. Ова класа омогућава позиве истом скупу метода које су имплементиране на серверској страни за оба начина имплементације комуникације у случају комуникације коришћењем локалне кућне мреже, као и у случају коришћењем интернет сервиса. *SharedSpaceComClient* класа је имплементирана као аутомат стања, а стања у којима се може наћи су представљена на слици 3.9: „проналажење сервера“, „није успостављена веза са сервером“, „успостављена веза са сервером“, „члан неактивне партије“ и „члан активне партије“.




Слика 3.9 Дијаграм стања клијентског дела компоненте „Комуникација“

У случају када се покушава успостављање везе са сервером у оквиру локалне кућне мреже, коришћењем аутоматског проналажења сервера, инстанца класе *SharedSpaceComClient* се налази стању „проналажење сервера“. Серверски део компоненте „Комуникација“ периодично шаље UDP мултикаст поруке које садрже IP адресу интернет сервера. По пријему поруке на клијентском делу компоненте „Комуникација“ врши се екстраховање IP адресе.

Случај када је сервер познат, али веза са сервером још није успостављена означава стање „није успостављена веза са сервером“. У овом стању инстанца класе *SharedSpaceComClient* на основу задате конфигурације покушава успостављање везе са сервером. Метода *loginToServer*се користи приликом регистрације корисника на сервер. Улазни параметри ове методе су име корисника као и његова улога која може бити учесник или посматрач. У случају успешно успостављене везе сервер враћа јединствен кориснички идентификатор који ће се користити у току даље комуникације са сервером и стање се из „није успостављена веза са сервером“ мења у „успостављена веза са сервером“. Када веза са сервером није успешно успостављена стање остаје непромењено.

Када се налази у стању „успостављена веза са сервером“ инстанца класе *SharedSpaceComClient* од сервера захтева листу свих доступних имплементација дељеног простора и партија којима је могуће придруживање. Ова листа се добија позивом методе *getHostedSharedSpaces*. Имплементације дељеног простора су све конкретне имплементације *SharedSpace* класе, а свака партија представља инстанцу *SharedSpace* класе која се извршава на серверу. Листа постојећих имплементације дељеног простора и конкретних партија се приказује на екрану. Корисник тада има могућност или да се придружи постојећој партији или да креира нову. Креирање активне партије једне од могућих имплементација дељеног простора је могуће позивом методе *createSharedSpaceInstance*. Као резултат позива методе *createSharedSpaceInstance* добија се идентификатор партије, а корисник који је креирао партију постаје домаћин. Метода *joinSharedSpaceInstance* омогућава кориснику да се придружи претходно креираној партији. У случају придруживања партији потребно је назначити улогу корисника која може бити учесник или посматрач. Учесник представља корисника који активно учествује у дељеном простору и свака имплементација

дељеног простора дефинише минималан и максималан број учесника. Посматрач је корисник који се најчешће односи на уређаје са великим екраном који служе само за приказ. У партији не постоји ограничење за број посматрача. У случају када се учесник придружује партији, на серверу се врши провера да ли се одговарајућој партији већ придружио максималан број учесника.

У оквиру стања „члан неактивне партије“ инстанца класе *SharedSpaceComClient* периодично захтева стање и листу свих корисникапријављених за ту партију. Стање партије и листа придружених корисника партији се добија позивом методе *getSharedSpaceInstanceInfo.* Добијене информације се приказују на екрану корисника. Уколико се партији придружио потребан број корисника домаћин може да је покрене. За покретање партије користи се *startSharedSpaceInstance* метода. Када је партија успешно покренута стање ће се променити у „ члан активне партије “. У случају неуспелог покретања партије корисник ће добити поруку о грешци и стање ће остати непромењено.

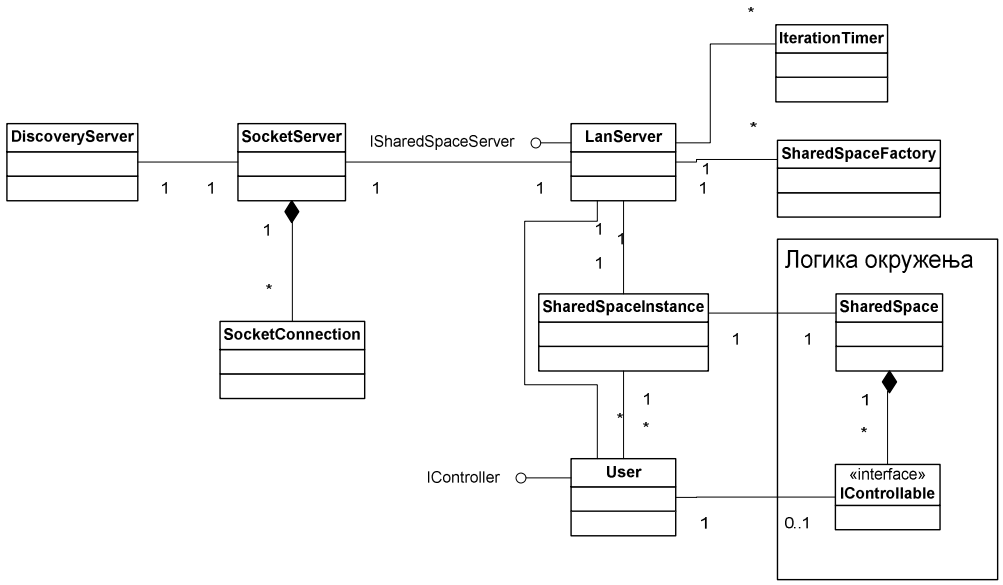
Када је у стању „члан активне партије“ клијентски део компоненте „Комуникација“ комуникациона добија листу *IterationChange* објеката и листу *ServerMessage* објеката које представљају поруке од сервера. Да би се решио проблем кашњења, који мрежа може да унесе приликом комуникације коришћењем интернет сервера, имплементиран је механизам баферисања. Бафери се пуне променама стања добијених од сервера, а затим се једна по једна промене шаљу компоненти „Приказ“. Алгоритам слања захтева серверу се прилагођава броју порука које се већ налазе у баферу. Уколико је утврђено да се бафер убрзано пуни, алгоритам успорава слање захтева, тј. повећава време између два захтева. Уколико је пак утврђено да се бафер убрзано празни, алгоритам смањује време између захтева како би што пре напунио бафер. Време између захтева је ограничено апсолутном доњом и горњом границом. Метода *getUpdates* враћа *IterationChange* објекте и *ServerMessage* објекте намењене одређеном кориснику. Слање контролних порука од клијента ка серверу омогућава метода *sendCommand.* Домаћин има могућност да искључи другог учесника из партије позивом методе *kickUser*. Метода *leaveSharedSpaceInstance* омогућава кориснику да напусти партију. Уколико домаћин напусти партију то ће резултовати завршетком партије. Из овог стања корисник може да се врати у стање

„успостављена веза са сервером“ у случају када напусти партију или у стање „није успостављена веза са сервером“ у случају када се веза са сервером прекине.

#### 3.2.3.2 Реализација серверског дела компоненте „Комуникација“ коришћењем локалне кућне мреже

На слици 3.10 је представљен класни дијаграм серверског дела компоненте „Комуникација“ која обезбеђује комуникацију у оквиру локалне кућне мреже. *DiscoveryServer* представља класу која одговара на клијентске поруке које имају за циљ проналажења сервера. *SocketServer* класа је задужена за успостављање везе са клијентом и за сваког повезаног корисника креираће се нови *SocketConnection* објекат. Поруке примљене од клијента се прослеђују *LanServer* класи која је одговорна за парсирање пристиглих порука. Сваки тип поруке поседује јединствен идентификатор на основу ког се одређује одговарајућа акција која треба да се предузме. Класа *LanServer* је такође задужена и за креирање потребног одговора клијенту и бива прослеђен одговарајућем клијенту коришћењем *SocketServer* класе.

Класа *User* представља корисника повезаног са сервером и поседује променљиву која означава улогу корисника која може бити учесник или посматрач. У случају када је улога корисника учесник класа *User* ће поседовати и референцу на *IControllable* интерфејс у оквиру компоненте „Логика окружења“, чиме омогућава „Логици окружења“ да прихвати командне поруке од клијента. Класа *User* такође имплементира и *IController* интерфејс који омогућава „Логици окружења“ прослеђивање порука клијенту. Задужење класе *SharedSpaceFactory* је креирање инстанци различитих имплементација дељеног простора. Класа *SharedSpaceInstance* представља омотач око креираних инстанци различитих имплементација дељеног простора. Ова класа поседује листу *User* објеката који представљају кориснике прикључене партији, као и информацију о томе који од корисника је домаћин партије. Такође, у оквиру ове класе се чувају и информације о томе да ли је партија почела или не. Класа *IterationTimer* је задужена за активирање периодичних итерација свих инстанци класе *SharedSpaceInstance* које представљају партију која је у току.




Слика 3.10 Класни дијаграм серверског дела компоненте „Комуникација“ која обезбеђује комуникацију у оквиру локалне кућне мреже

У оквиру *LanServer* класе имплементирано је девет различитих метода које клијенту обезбеђују одговоре: *loginToServer*, *getHostedSharedSpaces*, *createSharedSpaceInstance*, *joinSharedSpaceInstance*, *getSharedSpaceInstanceInfo*, *startSharedSpaceInstance*, *sendCommand*, *kickUser* и *leaveSharedSpaceInstance* метода. Када сервер прими од клијента поруку са захтевом за повезивање, у којој му клијент прослеђује своје име и улогу, позива се метода *loginToServer*. У оквиру ове методе секреира нова инстанца класе *User* која представља повезаног корисника. Инстанци класе *User* се придружује јединствен идентификатор који се прослеђује клијенту. Када сервер прими од клијента захтев за листом постојећих имплементација дељеног простора и постојећих креираних партија, у ком му клијент прослеђује свој кориснички идентификатор,позива се метода *getHostedSharedSpaces*. У оквиру ове методе се конструише структура података која садржи информације о свим имплементацијама дељеног простора и постојећим партијама за сваки од њих. Класа *SharedSpaceInfo* садржи име имплементације дељеног простора и додатне информације о свакој од креираних партија те имплементације дељеног простора. Информације о креираним партијама се чувају у виду листе инстанци *SharedSpaceInstanceInfo* класа.

*SharedSpaceInstanceInfo* класа садржи име партије, променљиву која означава да ли је партија активна и листу придружених корисника. Информације о придруженим корисницима се чувају у виду листе инстанци *UserInfo* класа. *UserInfo* класа садржи идентификатор клијента, његово име и улогу. Клијенту се прослеђује листа инстанци постојећих *SharedSpaceInfo* класа**.**

Када сервер прими од клијента поруку са захтевом за креирањем нове партије, у којој му клијент прослеђује свој кориснички идентификатор и име партије коју жели да креира, позива се метода *createSharedSpaceInstance*. У оквиру ове методе се креира инстанца класе *SharedSpaceInstance* коришћењем *SharedSpaceFactory* објекта, а корисник који је упутио захтев се аутоматски додаје као домаћин партије. Креирана инстанца класе *SharedSpaceInstance* се додаје *LanServer* објекту и идентификатор новокреиране партије се прослеђује кориснику. Само једна инстанца класе *SharedSpaceInstance* може постојати у једном тренутку на серверу. Методa*createSharedSpaceInstance* се најчешће директно позива са сервера као део покретања серверске апликације.

Када сервер прими поруку од клијента са захтевом за придруживање партији, у којој клијент прослеђује идентификатор корисника и идентификатор партије, позива се метода *joinSharedSpaceInstance.* У оквиру ове методе се проверава улога клијента и број максимално дозвољених корисника у тој партији. У случају када су проверени услови испуњени инстанца класе *User*, која одговара кориснику који је упутио захтев, ће бити додата инстанци класе *SharedSpaceInstance* чији идентификатор одговара жељеној партији. Корисник може да се придружи само једној партији, а као резултат кориснику се враћа обавештење о успеху придруживања жељеној партији. Када сервер прими поруку од клијента са захтевом за дохватање стања партије којој се придружио, у којој клијент прослеђује идентификатор корисника и идентификатор партије, позива се метода *getSharedSpaceInstanceInfo*. У оквиру ове методе се дохвата стање партије у виду *SharedSpaceInstanceInfo* класе и прослеђује клијенту. Када сервер прими поруку од клијента са захтевом за започињање партије, у оквиру које клијент прослеђује идентификатор корисника, позива се метода *startSharedSpaceInstance*. Покретањем партије стање објекта *SharedSpaceInstance* се мења, а итерације вођене *IterationTimer* класом ће започети. Инстанца класе *SharedSpace* ће за сваког

од корисника произвести *IterationChange* објекте и генерисати поруке у виду *ServerMessage* објеката које ће бити прослеђене корисницима.

Када клијент у току партије серверу пошаље контролну поруку, у оквиру које прослеђује идентификатор корисника и команду, са серверске стране се позива метода *sendCommand*. На основу идентификатора клијента проналази се одговарајућа инстанца класе *User* којој се прослеђују команде. Када сервер прими поруку од домаћина партије са захтевом за уклањањем специфицираног корисника, у оквиру које му клијент прослеђује идентификатор корисника (домаћина) и идентификатор клијента који треба да буде избачен, позива се метода *kickUser*. Инстанца класе *User* која одговара специфицираном кориснику ће бити уклоњена из листе активних клијената која се чува у оквиру *SharedSpaceInstance* објекта. У било ком тренутку сваки учесник може послати серверу поруку са захтевом за напуштање партије, у оквиру које серверу прослеђује свој кориснички идентификатор, и у том случају се позива метода *leaveSharedSpaceInstance*. Као резултат одговарајући *User* објекат се уклања из листе повезаних клијената.

#### 3.2.3.3 Реализација серверског дела компоненте „Комуникација“ коришћењем интернет сервиса

На слици 3.11 је представљен класни дијаграм серверског дела компоненте „Комуникација“ која обезбеђује комуникацију са интернет сервисом. *WebServer* класа обезбеђује имплементацију скупа метода које се позивају од стране клијентског дела компоненте „Комуникација“. Сваки од HTTP позива клијента иницира извршавање једне од метода имплементираних у оквиру *WebContainer* класе која користи EJB технологију како би обезбедила одржавање стања између различитих HTTP позива.

Класа *User* представља корисника повезаног са сервером и поседује променљиву која означава улогу корисника која може бити учесник или посматрач. У случају када је улога корисника учесник класа *User* поседује и референцу на *IControllable* интерфејс у оквиру компоненте „Логика окружења“, чиме омогућава „Логици окружења“ да прихвати командне поруке од клијента. Класа *User* такође имплементира и *IController* интерфејс који омогућава „Логици окружења“ прослеђивање порука клијенту. У оквиру класе *User* дефинисана је

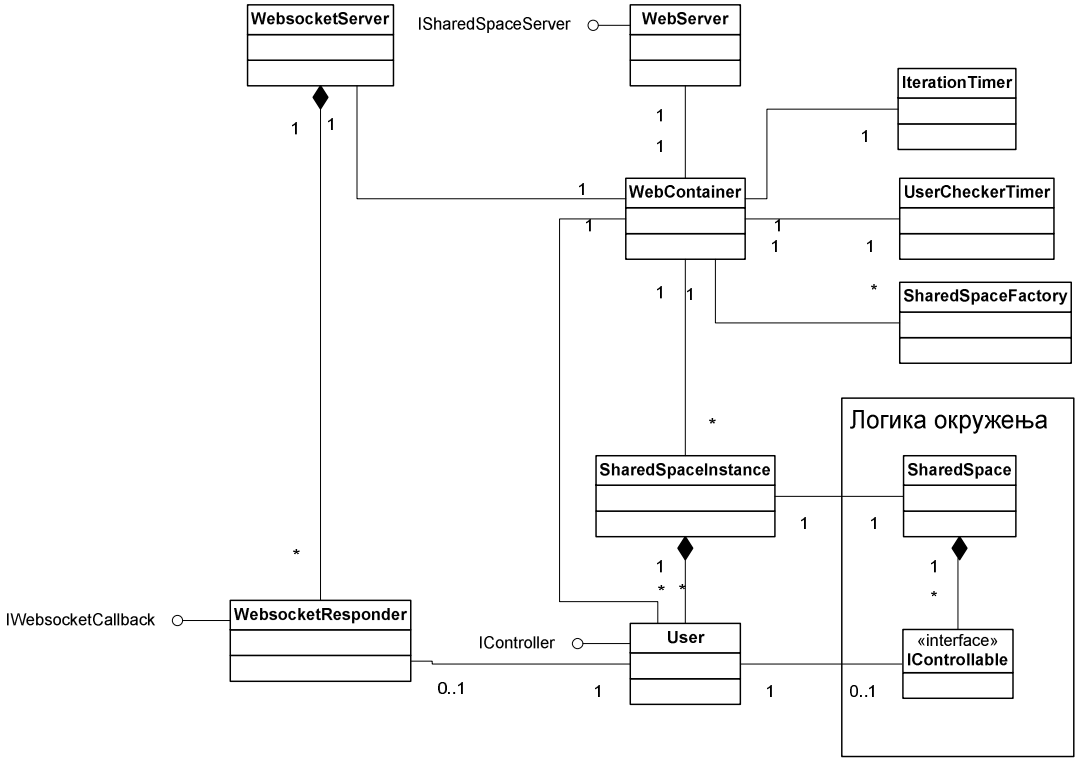
променљива која чува време последњег приступа серверу од стране клијента. Ова класа садржи референцу на објекат класе *WebsocketResponder* који се користи само у случају када постоји подршка за *WebSocket*-е.

Креирање инстанци различитих имплементација дељеног простора је задужење класе *SharedSpaceFactory*. Приликом иницијализације сервера постојеће имплементације се региструју код класе *WebContainer*. Класа *SharedSpaceInstance* представља омотач око креираних инстанци различитих имплементација дељеног простора. Ова класа поседује листу *User* објеката који представљају кориснике прикључене партији. Такође, у оквиру ове класе се чува и информација о томе да ли је партија почела или не. Класа *IterationTimer* је задужена за активирање периодичних итерација свих инстанци класе *SharedSpaceInstance* које представљају партију која је у току.

За сваког од повезаних клијената *UserCheckerTimer* класа је задужена за активацију периодичног израчунавања времена протеклог од последњег приступа серверу. У случају када је то време веће од дефинисаног дозвољеног, инстанца класе *User* која одговара неактивном клијенту се уклања. Класа *WebsocketServer* је одговорна за комуникацију коришћењем *WebSocket* протокола. Када се успостави веза коришћењем *WebSocket* потокола сваком клијенту који користи овај вид комуникације са сервером биће придружена инстанца класе *WebsocketResponder*. Ова класа је повезана и са одговарајућом инстанцом *User* класе.

У оквиру *WebContainer* класе имплементирано је десет различитих метода које обезбеђују одговоре потребне клијенту. То су: *loginToServer*, *getHostedSharedSpaces*, *createSharedSpaceInstance*, *joinSharedSpaceInstance*, *getSharedSpaceInstanceInfo*, *startSharedSpaceInstance*, *getUpdates*, *sendCommand*, *kickUser* и *leaveSharedSpaceInstance* метода. Поменуте методе имају исте функционалности као и у случају имплементације сервера у оквиру локалне кућне мреже, са додатком *getUpdates* методе која се употребљава само у случају када се за комуникацију између клијента и сервера користи HTTP протокол. У том случају у току партије клијент периодично шаље серверу захтеве за ажурирање партије и у оквиру захтева прослеђује серверу идентификатор корисника. На серверској страни се када пристигне захтев за ажурирањем позива *getUpdates* метода у оквиру које се креира листа промена од претходног захтева у виду листе

инстанци класе *Update*. Класа *Update* садржи листу инстанци класа *IterationChange* и *ServerMessage***.**




Слика 3.11 Класни дијаграм серверског дела компоненте „Комуникација“ која обезбеђује комуникацију са интернет сервисом

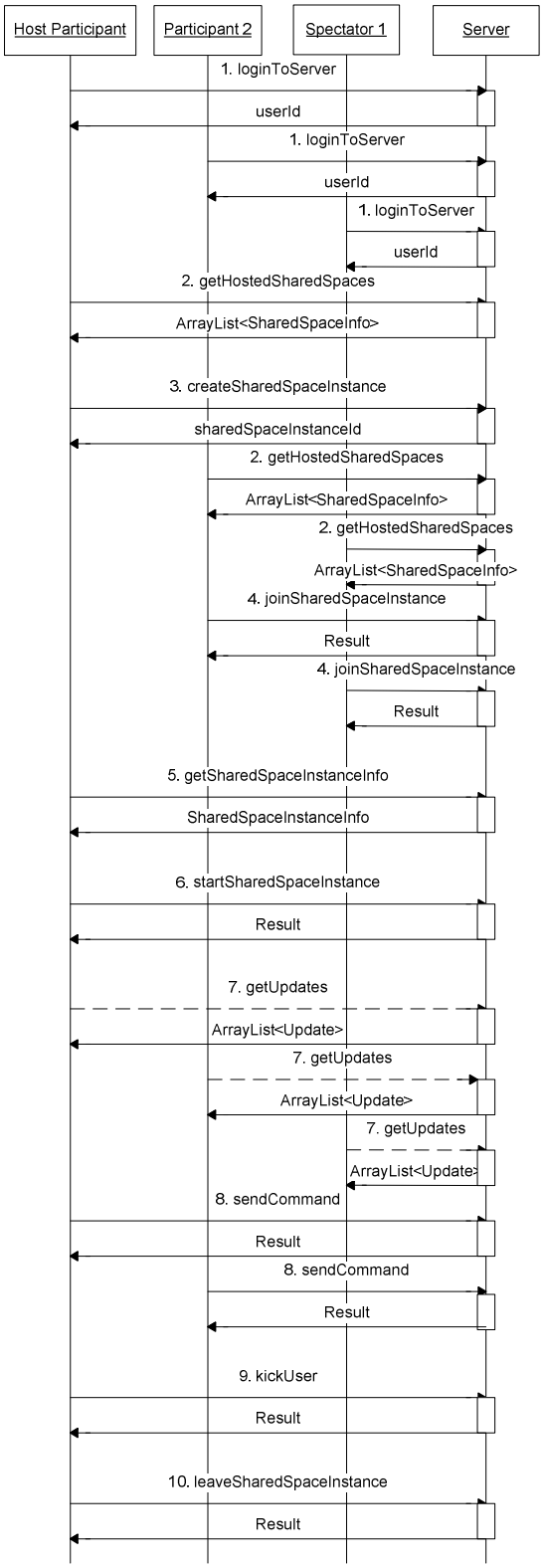
### 3.2.3.4 Интеракција између клијента и сервера

На слици 3.12 је приказан дијаграм интеракције између клијента и сервера у случају када се користи комуникација са интернет сервером и HTPP протокол комуникације. Корисник шаље серверу захтеве, а као одговор клијент добијаинстанцу класе *Result*. Ова класа садржи индикатор о успешности тражене операције, а у случају неуспеха садржи и поруку са објашњењем. У зависност од захтева класа *Result* може да садржи и додатне информације. Ове додатне информације су на слици 3.12 приказане као одговори на захтеве.

Партија из примера са слике 3.12 је предвиђена да се игра са два учесника и једним посматрачем. Сваки од корисника шаље захтев за повезивање са сервером

(*loginToServer*) и као одговор добија свој јединствени идентификатор. Први учесник шаље захтев за листом постојећих имплементација дељеног простора и креираних партија за сваку од имплементација (*getHostedSharedSpaces*) и након што добије одговор шаље захтев за креирање нове партије (*createSharedSpaceInstance*). Као резултат креирања нове партије клијенту се шаље јединствени идентификатор и клијент постаје домаћин те партије.

Након што је први учесник креирао партију, други учесник и посматрач шаљу захтев за листом постојећих имплементација дељеног простора и креираних партија (*getHostedSharedSpaces*). Када добију други учесник и посматрач шаљу захтев за придруживање партији коју је креирао први учесник (*joinSharedSpaceInstance*). Први учесник периодично шаље захтеве о стању корисника пријављених за партију коју је он креирао (*getSharedSpaceInstanceInfo*) и као резултат добија стање партије. Након што су се и други учесник и посматрач пријавили за исту партију, домаћин партије шаље захтев за покретање партије (*startSharedSpaceInstance*). Сва три корисника у току трајања партије периодично шаљу захтев за променама (*getUpdates*) и као резултат добијају листу промена од предходног захтева. Учесници партије шаљу контролне поруке (*sendCommand*). Домаћин партије шаље захтев за избацивањем другог учесника из партије (*kickUser*), а након тога и сам напушта партију (*leaveSharedSpaceInstance*) и тиме укида партију креирану на почетку секвенце.




Слика 3.12 Дијаграм интеракције између клијента и сервера

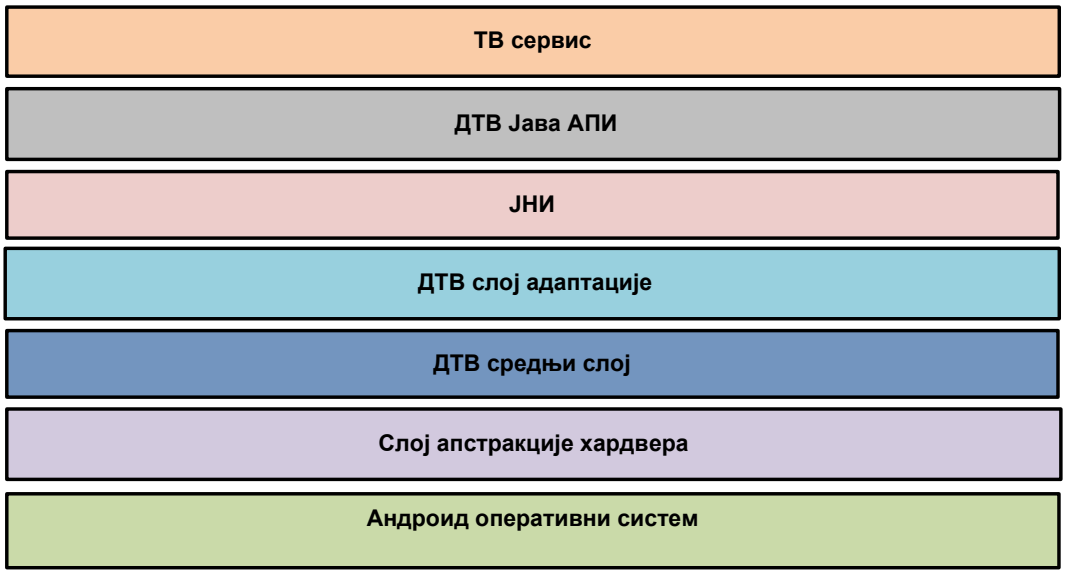
### 3.2.4 Компонента „ТВ сервис“

Апликација развијена коришћењем SHARP софтверске платформе комуницира са сервисима дигиталне телевизије коришћењем „ТВ сервис“ компоненте. Ова компонента настоји да кроз једноставан интерфејс програмеру понуди најчешће коришћени сет функција које могу да се искористе у апликацијама развијеним коришћењем SHARP софтверске платформе. Позиција развијене софтверске платформе у оквиру софтверског стека на Андроид уређајима са интегрисаном дигиталном телевизијом је приказана на слици 3.13.

Компонента „ТВ сервис“ представља омотач око већ постојећих ДТВ Јава АПИ-ја који су развијени у циљу интеграције сервиса дигиталне телевизије у модерне дигиталне телевизоре и ТВ пријемнике. ДТВ Јава АПИ има везу ка ДТВ средњем слоју у ком се налази системски код написан коришћењем C програмског језика. Системским рутинама писаним на језику C се приступа преко ЈНИ-а (*Java Native Interface*). Са друге стране, сви догађаји генерисани од стране ДТВ средњег слоја се пријављују ТВ сервису кроз JNI позивајући Јава метод одређен за пријем догађаја. Овај метод обрађује догађај и прослеђује га одговарајућем ДТВ Јава објекту. ДТВ слој адаптације представља спону између Јава АПИ-ја и ДТВ средњег слоја и пружа јединствени АПИ који омогућава приступ свим функционалностима. Његовим постојањем омогућено је да JNI слој остане не промењен. ДТВ средњи слој је задужен за надгледање транспортног низа, екстракцију специфичних табела сервисних информација и пружање лаких за приступ информација, као што су листе услуга и распоред догађаја. Овај слој омогућава и приступ одређеним хардверским компонентама дигиталног ТВ пријемника омогућавајући клијентима да на пример активирају процес снимања. Хардверске компоненте које се налазе на дигиталном ТВ пријемнику су представљене слојем апстракције ТВ хардвера.

Како би се осигурало извршавање апликације без прекида компонента „ТВ сервис“ обезбеђује неблокирајуће методе које враћају резултат захтева када је на располагању. Главне функционалности које се могу користити у оквиру развијене софтверске платформе су: пријављивање и одјављивање на сервис, захтев за активирањем и деактивирањем декодовања ТВ тока, захтев за адресом тока ТВ сигнала, промена јачине тона и добијање информације о тренутној јачини,

дохватање имена тренутног канала и промена канала, добијање информације о програму који се гледа и ономе што следи на том каналу.




Слика 3.13 Позиција компоненте„ТВ сервис“ у оквиру софтверског стека на Андроид уређајима са интегрисаном дигиталном телевизијом

Пријављивање на сервис мора бити извршено пре коришћења било које друге функционалности компоненте „ТВ сервис“. Ова операција шаље Андроид оперативном систему асинхрони захтев за повезивање, а након што добије одговор, о њему обавештава корисника. Компонента „ТВ сервис“ омогућава дохватање тренутног стања повезаности. Одјављивање не враћа никакву повратну информацију кориснику, већ само шаље захтев Андроид оперативном систему и стање сервиса мења на неповезано.

Како је обрада ТВ сигнала скупа операција она се не врши уколико не постоји експлицитан захтев неке од апликација за сигналом из ТВ тока. Због тога апликација може да захтева покретање или заустављање декодовања ТВ тока чиме се постиже уштеда у ресурсима. За приказивање ТВ слике у апликацији клијенти дохватају адресу тока сигнала. Са овом адресом могуће је приказати ток у било којој компоненти која подржава видео токове, као што је *VideoView* компонента Андроид оперативног система. Да би слика била приказана потребно је претходно активирати декодовање ТВ сигнала.

Дохватање тренутног канала омогућава апликацији да кориснику прикаже додатни садржај везан за канал, или на примеру игара да искористи ову информацију при објављивању резултата на неку од друштвених мрежа. Промена канала је операција која може трајати и преко једне секунде, па је интерфејс за њено извршавање асинхрони. Након што је канал успешно промењен посматрачки објекат бива обавештен о промени.

Како се звук ТВ сигнала аутоматски прослеђује на излаз, потребно је апликацији омогућити контролу над нивоом звука. Апликација може коришћењем компоненте „ТВ сервис“ да добије тренутну јачину звука као и да промени јачину звука за релативну вредност.

Апликација има могућност да од компоненте „ТВ сервис“ затражи назив програма који је у току, као и назив програма који ће следећи бити емитован. Како се ове информације могу мењати током времена омогућено је пријављивање на догађај промене вредности тренутног програма. Након што прими обавештење о догађају, апликација може да ишчита нове вредности тренутног и следећег програма.

### 3.2.5 Компонента „Друштвени медији“

Компонента „Друштвени медији“ пружа подршку за приступ интернет сервисима друштвених мрежа *Facebook* [33] и *Twitter* [67], као и бази метаподатака *The Movie Database* (TMDb) [62]. Због велике популарност друштвених мрежа њихова интеграција у апликације пружа предности као што су већа видљивост, популарност и лакше освајање тржишта [20]. Друштвене мреже представљају и одличну структуру за пласирање производа циљној групи људи [28]. *Facebook* и *Twitter* друштвене мреже су одабране због своје велике популарности [26]. TMDb представља базу података у којој се чувају информације о филмовима и ТВ емисијама. Из апликације је могуће вршити приступ и претраживање информација из TMDb базе, а омогућен је и унос нових информација у базу. Компонента „Друштвени медији“ представља омотач око већ постојећих јавних интерфејса које поменуте платформе пружају, омогућавајући позиве најчешће коришћеним методама. Како интернет сервиси којима се

приступа могу имати велико време одзива, понекад мерено и у секундама, компонента „Друштвени медији“ омогућава неблокирајуће позиве.

*Twitter* друштвена мрежа омогућава корисницима да објављују кратке поруке (до 140 карактера) које се називају „твитови“, да прате шта други корисници објављују и да остављају коментаре. Део компоненте „Друштвени медији“ задужен за комуникацију са *Twitter* интернет сервисом омогућава кориснику да директно из апликације развијене коришћењем SHARP платформе објави поруке и слике на налогу корисника или на налогу апликације. Имплементација користи *twitter4j* [77] библиотеку која представља имплементацију јавног интерфејса мреже *Twitter* на језику Јава. Ова библиотека је одабрана јер у поређењу са другим библиотекама, као што је *jtwitter* [74] пружа бољу документацију, програмерску подршку и слободнија права лиценцирања финалног производа. Коришћена библиотека не зависи од других пакета и може се користити без икаквих измена и на Андроид платформи. Имплементација подршке за ову друштвену мрежу не сакрива библиотеку коју користи већ представља њену надоградњу. Корисник има могућност објављивања предефинисаних порука које садрже име корисника и име програма који се тренутно гледа као и објављивања порука које садрже слику јавног, приватног или оба екрана.

Како би се обезбедио приступ *Twitter* мрежи из апликације неопходно је да корисник или апликација поседују свој налог и да се изврши регистрација апликације. Као резултат регистрације апликације добијају се кориснички кључ (*consumer key*) и корисничка тајна (*consumer secret*). Помоћу корисничког кључа и корисничке тајне од власника налога се тражи дозвола за приступ и као резултат добијају се токен за приступ (*access token*) и тајна за приступ (*access token secret*). Кориснички кључ, корисничка тајна, токен за приступ и тајна за приступ се чувају у оквиру апликације и омогућавају приступ *Twitter* мрежи. Повезивање и аутентификација се обављају само једном, или приликом покретања апликације или у одабраном тренутку у току извршавања апликације.

Подршка поседује класу која се користи за иницијализацију библиотеке подразумеваним вредностима, што омогућава апликацији да се коришћењем генеричког профила пријави на *Twitter* мрежу. Ово омогућава брзу имплементацију кода, без потребе за проласком кроз процедуре креирање налога

и регистровање нове апликације. Коришћење генеричког профила је корисно при тестирању апликације у почетној имплементацији, али је за финалну верзију производа неопходно регистровати апликацију.

Друштвена мрежа *Facebook* омогућава корисницима да креирају свој профил у оквиру ког могу да објављују слике, шаљу поруке и одржавају контакте са пријатељима, члановима породице и колегама. Омогућено је проналажење и комуникација између људи који имају заједничка интересовања као и креирање јавних страна у циљу промоције одређене теме или дешавања. Подршка за приступ *Facebook* интернет сервису користи библиотеку издату од стране *Facebook* развојног тима [61], намењену специфично за Андроид платформу. Континуиран развој, потпуна документација и независност од других пакета и библиотека су разлози због којих је ова библиотека одабрана за коришћење у оквиру SHARP платформе. Библиотека омогућава пријављивање, одјављивање корисника, слање захтева и обавештења о успешности. Подршка за приступ *Facebook* интернет сервису не покушава да сакрије библиотеку коју користи, већ само служи као њена надоградња. Корисник има могућност објављивања предефинисаних порука које садрже име корисника и име програма који се тренутно гледа као и објављивања порука које садрже слику јавног, приватног или оба екрана.

Да би корисник из апликације могао да приступи *Facebook* налогу потребно је да се претходно пријави на сајт. Приликом пријаве кориснику се приказује форма за унос корисничког имена и шифре, као и дијалог потврде давања приступа апликацији. Када корисник затвори дијалог апликација добија резултат о одлуци корисника да да дозволу за приступ налогу.

Као и код имплементације подршке за мрежу *Twitter*, и код подршке за приступ *Facebook* мрежи апликација пружа могућност пријаве коришћењем генеричког профила. Ово омогућава брзу имплементацију при тестирању и развоју. Ипак, приликом дистрибуције апликације неопходно је извршити њену регистрацију.

Подршка за приступ TMDb интернет сервису користи библиотеку JTMDB [36] написану коришћењем Јава програмског језика. У тренутку развоја, ова библиотека је била једина доступна за уређаје са Андроид оперативним системом.

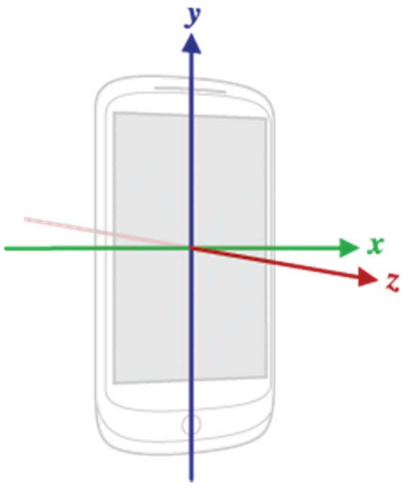
Функционалности које пружа подршка за TMDb корисницима апликација развијеним коришћењем SHARP платформе су претрага или унос додатних информацијама о програму који се прати упоредо са коришћењем апликације. Пример претраге је претрага филмова по имену, режисеру, години или глумцима где се као резултат добијају тражени детаљи филма. Упис података у базу даје могућност корисницима да оцене филмове и ТВ емисије.

Како би се омогућио приступ бази и упис података из апликације потребно је да се обави процедура аутентификације. Као резултат процедуре добија се идентификатор сесије који се користи сваки пут када апликација шаље захтев за упис података у базу. Када се покрене апликација потребно је да се затражи токен помоћу ког ће власник налога дати дозволу за приступ TMDb налогу из апликације. Процес аутентификације не захтева чување или пренос лозинке. Од корисника се тражи да се на страни TMDb сервиса пријави на свој налог, на исти начин као што би то урадио коришћењем интернет претраживача.

### 3.2.6 Компонента „Детекција покрета“

Већина мобилних уређаја поседује акцелерометре и магнетометре који обезбеђују довољно информација потребних за детекцију основних покрета као што су „замах“, „ударац“, „мућкање“, „искошење“ или „балансирање“. Ови покрети могу да се употребе за управљање садржајем који се појављује на јавним и приватним екранима у интегрисаном окружењу. Како би могле да се искористе за детекцију покрета информације добијене од сензора је потребно да се обраде.

Координатни систем мобилног уређаја је приказан на слици 3.14. X оса представља хоризонталну осу која пролази преко екрана телефона и орјентисана је ка десној страни. Y оса представља вертикална осу која пролази преко екрана телефона и орјентисана је ка врху телефона. Z оса представља осу која показује ка небу у случају када је телефон постављен на столу са екраном на горе. Акцелерометар омогућава мерење убрзања уређаја по свакој од три осе координатног система изражено у $m/s^2$. Када уређај лежи мирно на столу убрзање по осама x и y је нула, а по оси z је -9.81 $m/s^2$ због реакције на гравитацију. Магнетометар мери магнетно поље по свакој од три осе координатног система изражено у uT.

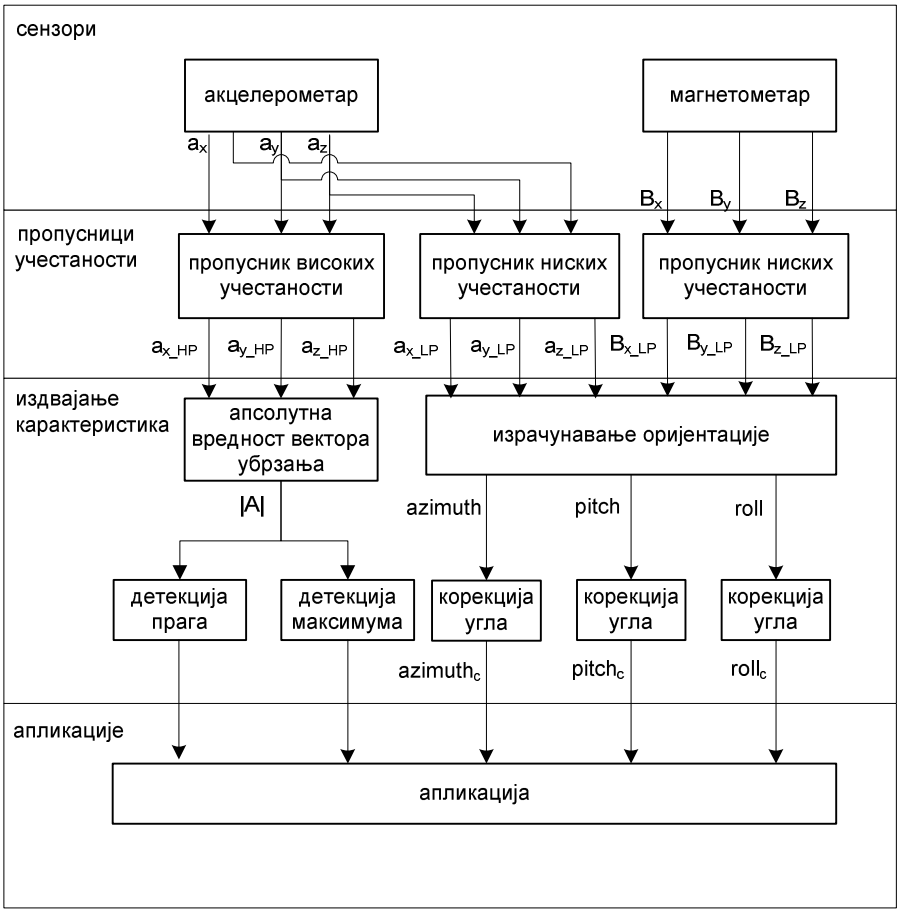



Слика 3.14 Координатни систем мобилног уређаја

Вредностима које се добијају са излаза сензора програмер може да приступи коришћењем постојећих АПИ-ја Андроид оперативног система који омогућавају пријављивање на догађаје жељеног сензора. **SensorManager** је системска класа која омогућава приступ сензорима, односно омогућава регистрацију и одјаву, а понекад и обраду података. Код **SensorManager** класе је потребно регистровати **SensorEventListener** који ће примати **SensorEvent** податке, а при регистровању је потребно навести и учесталост мерења. Учесталост мерења се може дефинисати или у милисекундама или се може одабрати из листе предефинисаних вредности: SENSOR_DELAY_NORMAL, SENSOR_DELAY_UI, SENSOR_DELAY_GAME, SENSOR_DELAY_FASTEST. Узимајући у обзир контекст изабраних апликација у нашем случају је одабрана SENSOR_DELAY_GAME константа, која је вршила мерења на 30 милисекунди, односно око 33 мерења по секунди.

На слици 3.15 је представљена структура компоненте „Детекција покрета“. Вредности које се добијају на излазу сензора (акцелерометра и магнетометра) се филтрирају пропусником високих и ниских учестаности. Како би се елиминисала гравитација, вредности са излаза акцелерометра се пропуштају кроз пропусник високих учестаности, а у циљу елиминације шума излази акцелерометра и магнетометра се доводе на улаз пропусника ниских учестаности. На основу добијених вредности са излаза пропусника могуће је издвојити основне карактеристике у виду: оријентације уређаја, апсолутне вредности вектора

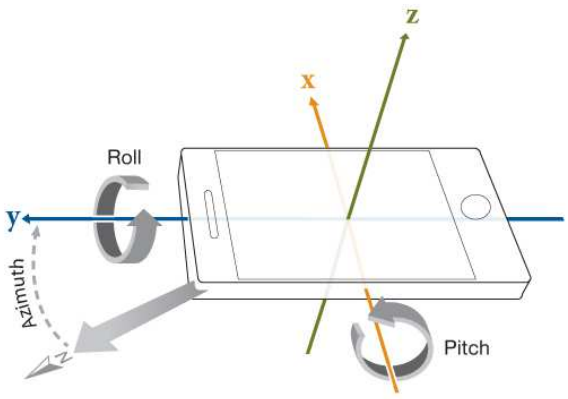
убрзања уређаја, детекције прага и детекције максимума. Ове карактеристике се могу употребити у апликацијама развијеним коришћењем SHARP платформе како би се детектовао жељени покрет.




Слика 3.15 Структура компоненте „Детекција покрета“

На основу вредности са излаза акцелерометра и магнетометра софтверским путем, позивом системске функције **SensorManager.getOrientation()**, добија се оријентације уређаја. Оријентација се одређује на основу ротације око сваке од оса координатног система и приказана је на слици 3.16. Ротација око Z осе (*azimuth*) се мери угловима од 0 до 360°, где 0 представља север, 90° исток, 180° југ и 270° запад. Ротација око x осе (*pitch*) се мери угловима од -180° до 180° са позитивним вредностима у случају када се z оса помера ка у оси. Ротација око у

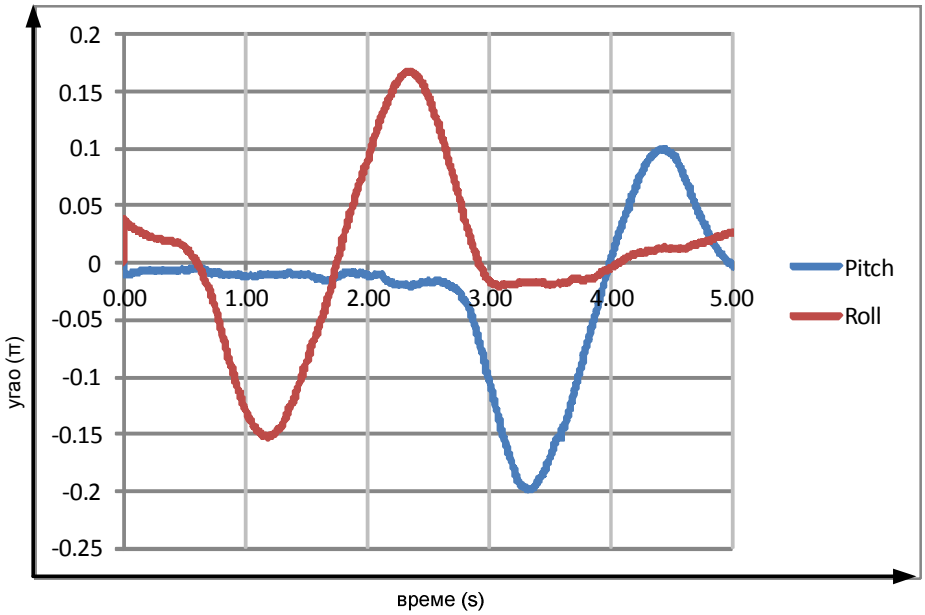
осе (*roll*) се мери угловима од -90° до 90° са позитивним вредностима у случају када се оса z помера ка x оси.




Слика 3.16 Оријентација мобилног уређаја

Ограничење распона угла за ротације око оса координатног система доводи до потенцијалних наглих скокова приликом мерења угла уређаја који се окреће око неке од оса. На пример ако се телефон окрене око x осе за више од 360° вредност *pitch* угла ће постепено достићи вредности од 180° (π) након чега ће нагло скочити на -180°(-π). Како би се добила континуална вредност угла ротације приликом мерења користи се корекција угла која омогућава да се детектује промена већа од дефинисане вредности и коригује додавањем угла величине 2π или -2π.

Измерене вредности углова *azimuth*, *pitch* и *roll* током времена, односно колико је мобилни уређај нагнут на коју страну се могу искористити у апликацијама приликом симулације континуалних покрета. Неки од таквих покрета могу бити „балансирање куглице на подлози“ или „управљање возилом“. Код „балансирања куглице на подлози“ мобилни уређај представља подлогу коју је потребно ротирати око x или y осе како би се куглица откотрљала у жељену страну. Приликом „управљања возилом“ држање мобилног уређаја симулира држање волана. На слици 3.17 је представљен дијаграм на ком су приказане вредности углова *pitch* и *roll* током времена приликом коришћења мобилног телефона као подлоге за балансирање куглице која се котрља.

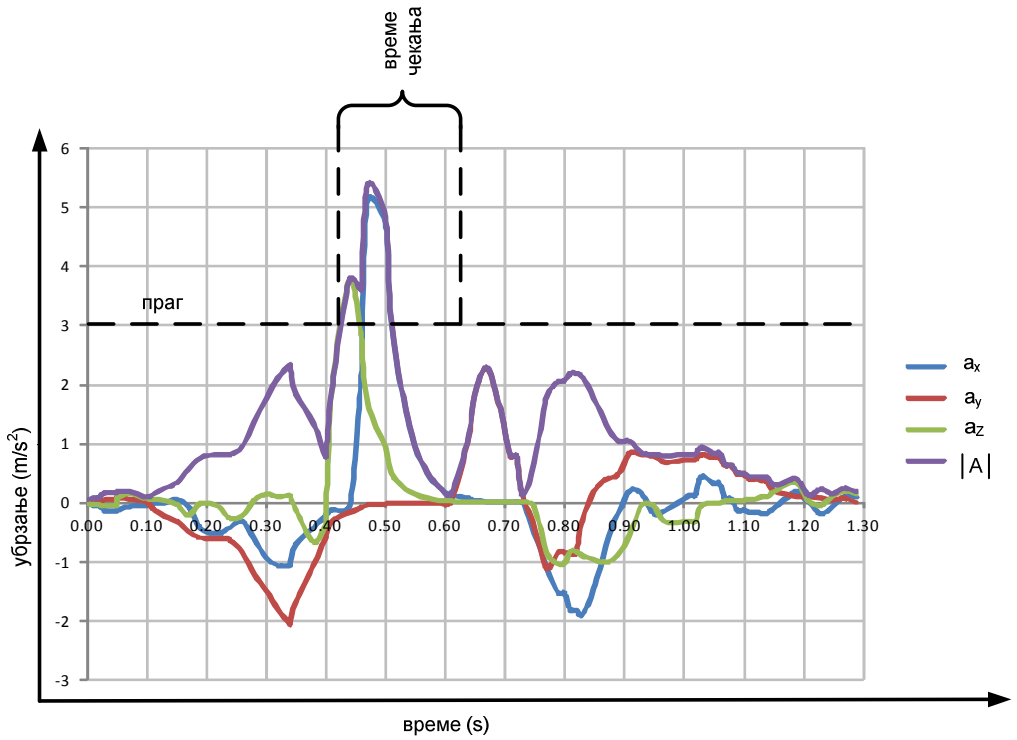



Слика 3.17 Дијаграм углова *pitch* и *roll* током времена приликом симулације „балансирања куглице на подлози“

На основу добијених вредности убрзања за сваку од оса координатног система могуће је израчунати и апсолутну вредност вектора убрзања на основу формуле $|A| = \sqrt{{a_x}^2 + {a_y}^2 + {a_z}^2}$, где $a_x$ , $a_y$ и $a_z$ представљају убрзања по осама x, y и z координатног система, а $|A|$ апсолутну вредност вектора убрзања. Детектор прага омогућава да се детектује тренутак када апсолутна вредност вектора убрзања пређе дефинисани праг. Коришћењем детектора прага могуће је регистровати покрете као што су „ударац“ или „замах“. Приликом једног покрета „ударца“ апсолутна вредност вектора убрзања више пута пређе дефинисани праг, па се дефинише и време које је потребно да се сачека након детекције пређеног прага које називамо време чекања. Након што време чекања истекне поново је могуће детектовати тренутак када апсолутна вредност вектора убрзања пређе дефинисани праг.

На слици 3.18 је представљен дијаграм који у случају покрета „ударца“ приказује убрзања за сваку од оса координатног система ($a_x$, $a_y$, $a_z$) као и апсолутну вредност вектора убрзања ($|A|$) мерене током времена. Праг је дефинисан као убрзање од 3 m/s$^2$ , а време чекања 0,2 s. У временском тренутку 0,42 s апсолутна вредности вектора убрзања је прешла дефинисани праг и у том

тренутку ће се генерисати догађај који означава детекцију покрета „ударца“. Након што истекне време од 0,2 s у тренутку када апсолутна вредности вектора убрзања  пређе праг од 3 $m/s^2$ опет ће се генерисати догађај који означава детекцију покрета „ ударца“.

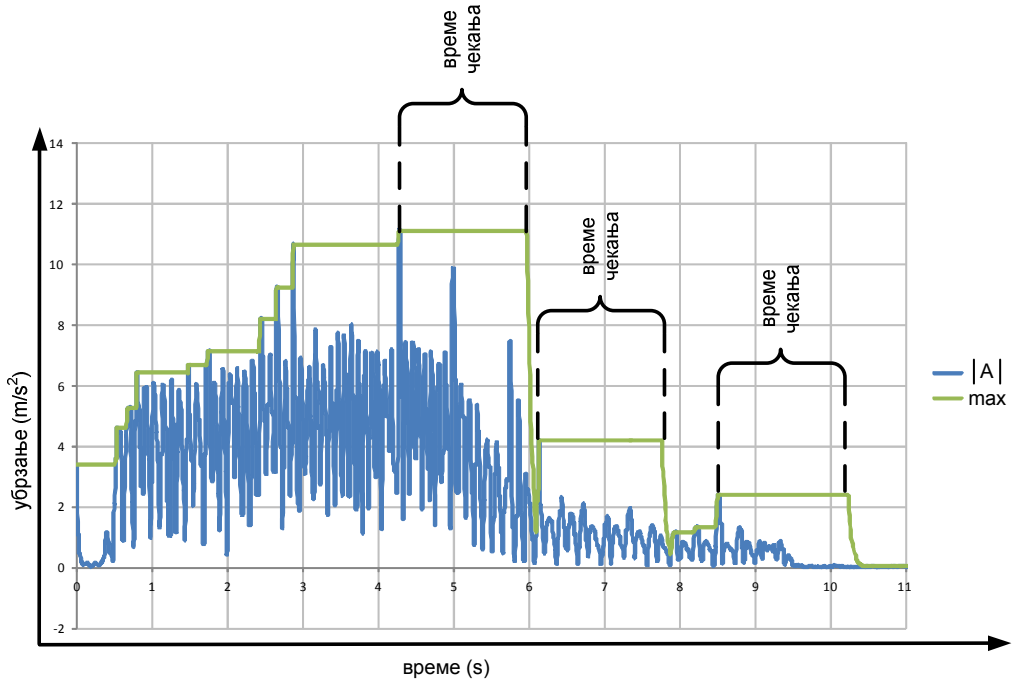



Слика 3.18 Дијаграм убрзања током времена приликом симулације покрета „ударца“

Детектор максимума бележи максималну вредност апсолутне вредности вектора убрзања за дефинисани временски период који називамо време чекања. У случају када је тренутно измерена вредност апсолутне вредности вектора убрзања већа од претходно запамћене максималне вредности, тада максимална вредност постаје тренутна вредност апсолутне вредности вектора убрзања, а време чекања се враћа на почетну вредност. У случају да у дефинисаном интервалу времена чекања вредност магнитуде није прешла максималну, максимална вредност почиње постепено да опада.

Детектор максимума се може искористити за детекцију јачине „муњкања“ у случају када се дефинише кратко време чекања или јачине „ударца“ у случају када

се дефинише дуже време чекања. На слици 3.19 је представљен дијаграм убрзања током времена приликом детекције покрета „мућкања“. Време чекања је 1,8 s.




Слика 3.19 Дијаграм убрзања током времена приликом симулације покрета „мућкања“

## 3.3 Размена информација између дистрибуираних компоненти SHARP софтверске платформе

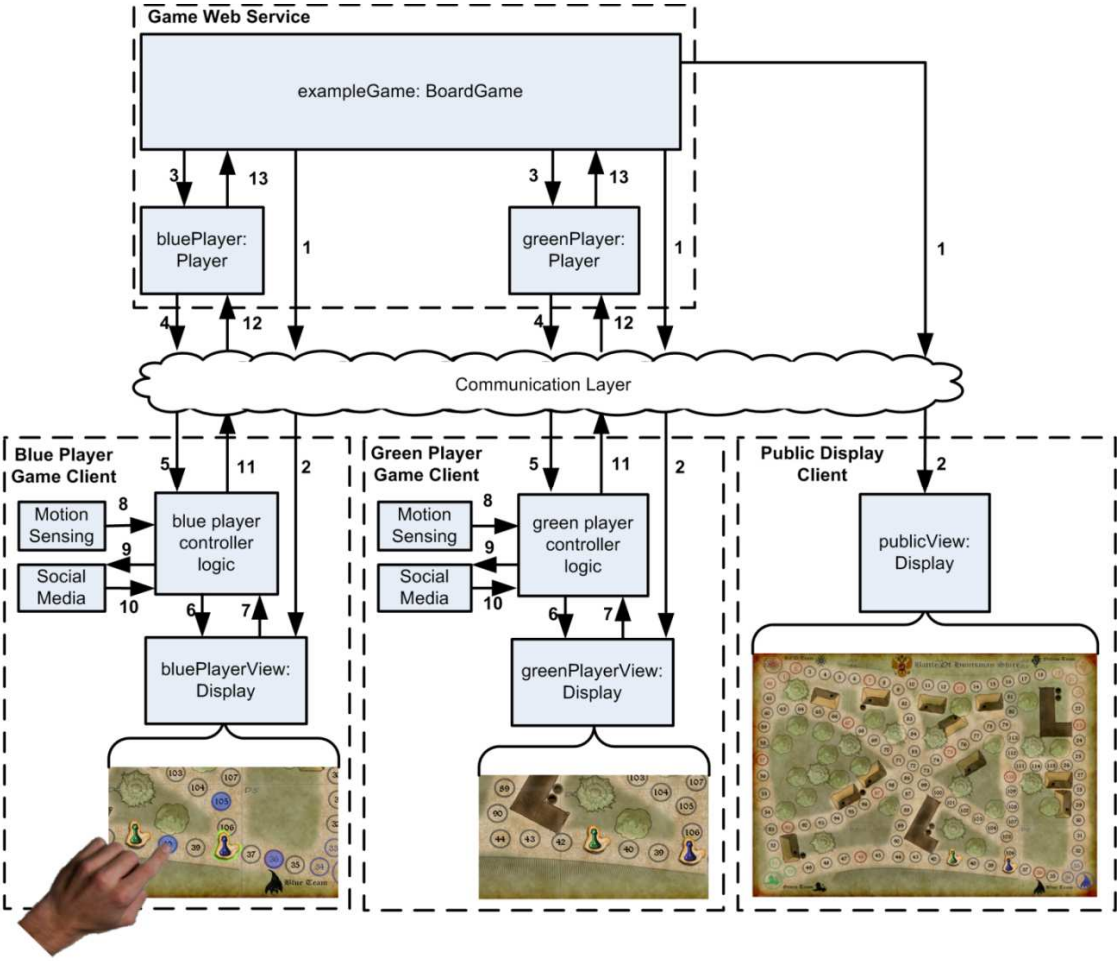
Размена информација између дистрибуираних компоненти система је приказана на примеру игре на табли, као што је дато на слици 3.20. На јавном екрану телевизора видљива је цела табла која садржи једног плавог пиона који припада плавом играчу и једног зеленог пиона који припадају зеленом играчу. Сваки од играча има мобилни уређај на чијем екрану се приказује део табле игре центриран на пиону који припада том играчу. Плави играч је управо бацио коцкицу и добио број два. Поља осенчена у плаво представљају позиције на које је могуће померити пиона. Играч додиром екрана на једно од плаво осенчених поља  бира позицију на који жели да помери пиона. Овај пример илуструје и

ситуацију у којој се екран не користи само за приказ података, него и за контролу игре. Испрекиданим линијама на слици су представљене апликације које се извршавају на различитим уређајима: *GameWebService*, *PlayerGameClient* и *PublicDisplayClient*. Апликација *GameWebService* се извршава на интернет серверу, *PlayerGameClient* на приватном уређају играча и *PublicDisplayClient* на уређајима са јавним екраном.

Апликација *GameWebService* садржи инстанцу компоненте „Логика окружења“ и имплементира логику игре која се састоји од инстанце класе *BoardGame* и две регистроване инстанце класе *Player*. Клијент користи апликацију *PlayerGameClient* за контролу инстанце класе *Player*. Ова апликација је идентична за плавог и зеленог играча. Апликација *PlayerGameClient* садржи инстанцу компоненте „Приказ“ конфигурисану да омогући приказ одговарајућег дела табле, инстанцу компоненте „Детекције покрета“ која омогућава детекцију покрета и инстанцу компоненте „Друштвени медији“ која омогућава слање порука на друштвене мреже. Логика контролера (*controller logic*) представља део програмског кода који је независтан од софтверског пакета и одговоран је за прикупљање корисничких улаза, размену информација са инстанцама компонената, присутним на клијентској апликацији и размену информација са придруженом инстанцом класе *Player*, присутној у логици игре. Апликација *PublicDisplayClient* садржи инстанцу компоненте „Приказ“ конфигурисану да омогући приказ целе табле. Компонента „Комуникација“ је приказана као транспарентни комуникациони слој.

Екран уређаја плавог играча показује део табле тако да се плави пион налази на средини екрана. Плаве сенке на пољима су инстанце класе *GameElement* и креиране су од стране игре и трају онолико колико потез траје, након чега ће их игра уклонити. Плаве сенке на пољима припадају приватној области приказа плавог играча, па се не појављују ни на једном другом екрану. Када се додирне нека од позиција која је осенчена плавом бојом логика контролера се обавештава о предузетој акцији. Логика контролера даље обавештава објекат *Player* о предузетој акцији што резултира померањем пиона на жељену позицију. Екран уређаја зеленог играча приказује само део табле са зеленим пионом у центру. Плави пион је видљив десно, а плаве сенке на пољима нису видљиве. Јавни екран

приказује целу таблу која садржи оба пиона, плавог и зеленог, без приказа осенчених позиција.




Слика 3.20 Размена информација између дистрибуираних компоненти система

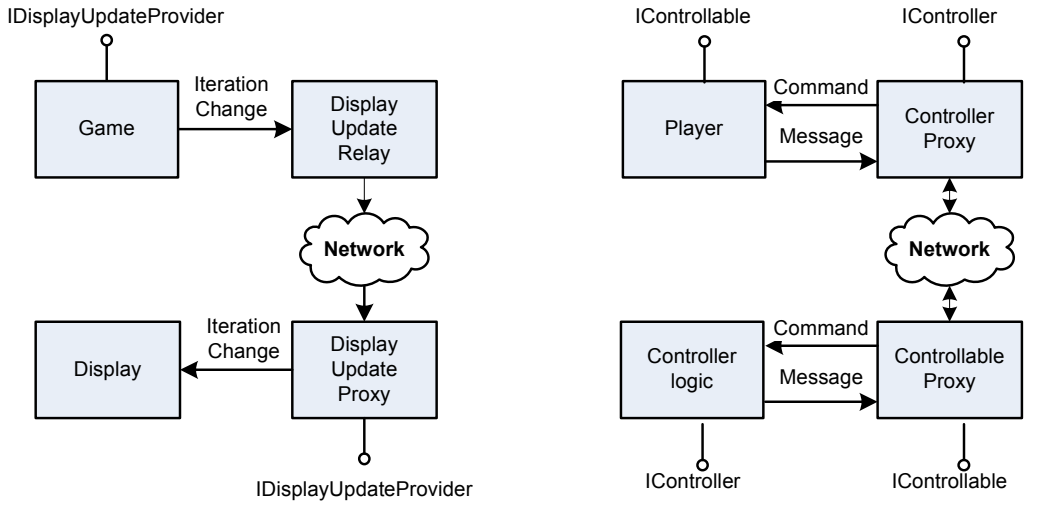
Нумерисане линије између компоненти на слици 3.20 указују на размену информација која се обавља на следећи начин:

1. У току сваке итерације у игри инстанца класе *BoardGame* производи по једну инстанцу класе *IterationChange*. У зависности од одговарајућих области приказа које су придружене сваком од екрана, инстанце класе *IterationChange* се филтрирају и прослеђују комуникационом слоју, где се врши серијализација.

2. У оквиру комуникационог слоја врши се десеријализација инстанце класе *IterationChange* и прослеђивање одговарајућој инстанци компоненте „Приказ“ у циљу исцртавања света игре.
3. Инстанца класе *BoardGame* обавештава инстанцу класе *Player* о променама у стању игре. Пример обавештења може бити информација да је играч на потезу.
4. Инстанца класе *Player* шаље контролеру логике поруке о стању игре које имају утицај на логику игре. Један пример поруке је обавештење да је играч на потезу што ће активирати вибрацију на мобилном уређају играча. Други пример поруке је команда за центрирање камере на пиона који се креће. Ове поруке се шаљу комуникационом слоју где се серијализују и прослеђују даље.
5. Комуникациони слој десеријализује поруке примљене од стране инстанце класе *Player* и прослеђује их логици контролера уређаја на ком је потребно да се предузме одговарајућа акција.
6. Логика контролера прослеђује информације компоненти „Приказ“ у циљу контролисања камере. Као пример може се навести потреба да се камера центрира на другог пиона.
7. Компонента „Приказ“ прослеђује контролеру логике информацију о детектованим догађајима додира.
8. Компонента „Препознавање покрета“ прослеђује контролеру логике информацију о детектованим покретима.
9. Логика контролера прослеђује резултате игре компоненти „Друштвени медији“ са циљем да се резултати игре објаве на друштвеним мрежама.
10. Компонента „Друштвени медији“ јавља контролеру логике да је порука успешно објављена.
11. Логика контролера шаље коришћењем комуникационог слоја поруке о контроли одговарајућој инстанци класе *Player*. У оквиру комуникационог слоја поруке се серијализују.
12. Комуникациони слој десеријализује поруке на страни сервера и прослеђује их инстанци класе *Player* где се предузимају одговарајуће промене стања игре.

13. Инстанца класе *Player* интерпретира примљене поруке контроле од играча и по потреби мења стање игре.

На слици 3.20 линије које улазе у или излазе из комуникационог слоја имају додатни корак имплементиран коришћењем помоћних класа које имају за циљ да сакрију присуство комуникационог слоја и раздвоје софтверске компоненте у оквиру дистрибуираног система. У ситуацијама, као што је у поменутом примеру игре на табли, када инстанце компоненте „Приказ“ нису на истом уређају као и инстанца класе *Game*, инстанца класе *IterationChange* се шаље коришћењем помоћних *DisplayUpdateRelay* и *DisplayUpdateProxy* класа. Инстанца *DisplayUpdateRelay* класе прикупља инстанце класе *IterationChange*, серијализује их и шаље свим клијентима који поседују инстанцу компоненте „Приказ“. Инстанца *DisplayUpdateProxy* класе прима поруке потребне за ажурирање слике на екрану, десеријализује их и доставља их инстанци компоненте „Приказ“. Интерфејс *IDisplayUpdateProvider* дефинише ко креира *IterationChange* објекте. Овај интерфејс имплементирају и инстанца класе *Game* и инстанца класе *DisplayUpdateProxy*. Размена информација између инстанце класе *Game* и компоненте „Приказ“ је представљена на слици 3.21а).




Слика 3.21а) Размена информација између објекта *Game* и компоненте „Приказ“ б) Размена информација између објекта *Player* и логике контролера

Контролна логика прикупља акције играча и прослеђује их *Player* објекту у облику командне поруке. Класа *Player* имплементира *IControllable* интерфејс чиме показује да је спремна да прихвати командне поруке од контролне логике. Контролна логика имплементира *IController* интерфејс чиме показује да је може да прихвати поруке од *Player* објекта. Инстанце класе *Player* најчешће нису присутне на истом уређају као и контролна логика, па се размена порука врши коришћењем објеката који комуницирају коришћењем компоненте „Комуникација“. Размена информација између објекта *Player* и контролне логике је приказана на слици 3.21б).

# 4. ТВ ЦЕНТРИЧНЕ ИГРЕ РАЗВИЈЕНЕ КОРИШЋЕЊЕМ SHARP СОФТВЕРСКЕ ПЛАТФОРМЕ

Игре које је могуће развити у интегрисаном окружењу су назване ТВ центричне игре. Коришћењем SHARP софтверске платформе имплементирано је пет различитих ТВ центричних игара од којих су две потезне игре и три игре у реалном времену. Од потезних игара имплементирана је једна игра на табли и једна игра картама. Обе игре су имплементиране на два начина коришћењем локалне кућне мреже и интернет сервиса. Три развијене игре у реалном времену су назване микро игре и имплементиране су тако да се играју у току гледања телевизијског програма. Све три микро игре су развијене тако да се играју само у оквиру локалне кућне мреже. Свака од игара се састоји од више апликација, а свака апликација имплементира различите компоненте софтверске платформе. Такође, свака од игара пружа корисницима различите комбинације идентификованих сценарија интеракције. У поглављима које следе свака од игара ће бити детаљније описана. За сваку игру ће бити наведени сценарији интеракције које подржавају и објашњена дистрибуција компонената коришћењем различитих апликација. У последњем поглављу биће представљени резултати испитивања корисника приликом употребе ТВ центричних игара.

## 4.1 Игра *Dice of Blood*

Имплементирана игра на табли, названа *Dice of Blood*, представља стратегијску игру сличну играма „Не љути се човече“ и „Ризико“. Игру може да игра између два и четири играча, који играју или сваки за себе или у тиму од по два играча. Сваки играч поседује два пиона, која покреће по табли. Игра се састоји од 20 рунди. У свакој рунди играч баца коцкицу и помера једног од својих пиона за број поља који зависи од резултата бацања коцкице. На ТВ екрану се приказује табла и резултат игре, а мобилни уређаји се користе за контролисање

пиона, бацање коцкице и распоређивање инвентара (мине, мачеви и самострели). Играчи добијају поене за убијање противничког пиона, што се може постићи доласком на поље противничког пиона, али и различитим врстама оружја која имају деловање на одређен број околних поља. Играчи нова оружја добијају на почетку сваког круга. Кретањем по мапи могуће је покупити додатке, који играчевом пиону дају специјалне моћи као што је шанса да преживи напад противника, брже кретање и више оружја [66]. Игра *Dice of Blood* је приказана на слици 4.1.

Слика 4.1 Игра *Dice of Blood*

У оквиру игре *Dice of Blood* корисницима су пружени следећи сценарији интеракције: подела приказа на више екрана, употреба мобилног уређаја за управљање акцијама на централном екрану, детекција покрета коришћењем сензора, додатне чулне повратне информације, корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака и интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама. Сценарији интеракције човек-рачунар у интегрисаном окружењу и начин на коју су подржани у оквиру *Dice of Blood* игре развијене коришћењем интернет сервиса су представљени у табели 4.1, а за игру *Dice of Blood* развијену коришћењем локалне кућне мреже су представљени у табели 4.2.

Табела 4.1 Сценарији интеракције човек-рачунар у интегрисаном окружењу и начин на који су подржани код *Dice of Blood* игре развијене коришћењем интернет сервиса

| **Сценарији интеракције** | **Игра *Dice of Blood* развијена коришћењем интернет сервиса** |
|---|---|
| подела приказа на више екрана | ТВ екрани су искоришћени за приказ целе мапе игре, а екрани мобилних уређаја за приказ дела мапе и додатних приватних информација |
| употреба мобилног уређаја за управљање акцијама на централном екрану | управљање пионима и постављање инвентара на мапи |
| детекција покрета коришћењем сензора | акција „ мућкања“ и „бацања“ коцкице се детектује употребом мобилног уређаја |
| додатне чулне повратне информације | вибрацијом мобилног уређаја играч се обавештава да је на потезу |
| пасивно гледање телевизије обогаћено паралелним интерактивним активностима | - |
| корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака | резултате игре је могуће објавити на друштвеним мрежама |
| интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама | више људи у оквиру једне просторије учествује у истој игри са људима који су смештени у другој просторији |

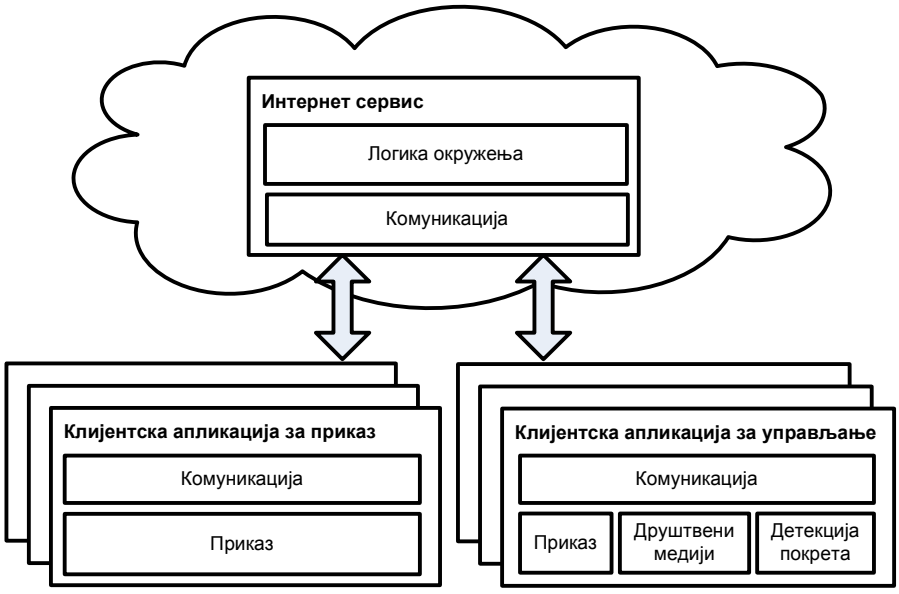
Приказ игре је подељен на јавне ТВ екране и приватне екране мобилних уређаја. Велики ТВ екран је искоришћен за приказ мапе игре, односно онога што се код традиционалних друштвених игара најчешће налази на табли којој сви играчи могу да приступе и чији садржај сви могу да виде. Мали екрани мобилних уређаја су искоришћени за приказ дела мапе на ком се њихови пиони налазе, као и приватних информација о томе које оружје поседују они или чланови њиховог тима. Додиром на одговарајуће место на екрану мобилних уређаја могуће је поставити пиона на жељено место на мапи или одабрати оружје и место на мапи

на ком ће одабрано оружје бити постављено. Мобилни уређаји су искоришћени и за симулацију физичких акција у традиционалним играма на табли, као што је мућкање и бацање коцкице. Вибрацијом мобилног уређаја кориснику се даје до знања да је на потезу. Резултати игре се објављују на друштвеним мрежама дајући могућност и пријатељима који нису директно укључени у игру да буду информисани о исходу игре. Смештањем логике игре на интернет омогућава пријатељима који се налазе у различитим дневним собама да интерреагују играјући исту игру. Као додатак пријатељи имају могућност поделе на тимове, где су, на пример, чланови истог тима смештени на истој локацији.

Табела 4.2 Сценарији интеракције човек-рачунар у интегрисаном окружењу и начин на који су подржани код *Dice of Blood* игре развијене у оквиру локалне кућне мреже

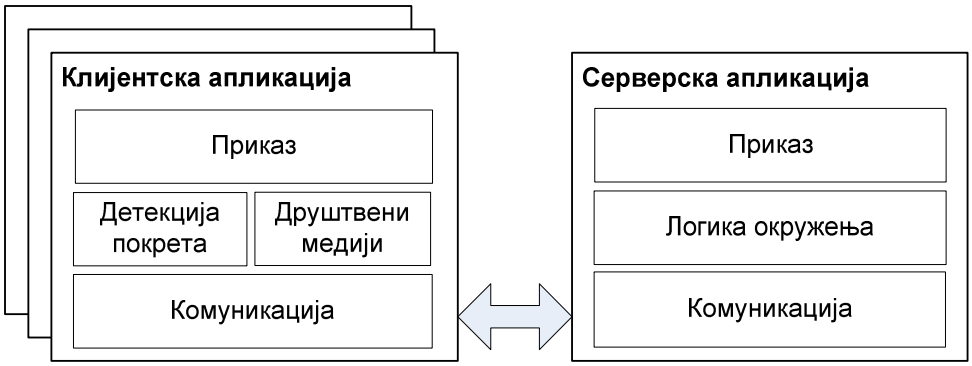
| **Сценарији интеракције** | **Игра *Dice of Blood* у оквиру локалне кућне мреже** |
|---|---|
| подела приказа на више екрана | ТВ екрани су искоришћени за приказ целе мапе игре, а екрани мобилних уређаја за приказ дела мапе и додатних приватних информација |
| употреба мобилног уређаја за управљање акцијама на централном екрану | управљање пионима и постављање инвентара на мапи |
| детекција покрета коришћењем сензора | акција „ мућкања“ и „бацања“ коцкице се детектује употребом мобилног уређаја |
| додатне чулне повратне информације | вибрацијом мобилног уређаја играч се обавештава да је на потезу |
| пасивно гледање телевизије обогаћено паралелним интерактивним активностима | - |
| корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака | резултате игре је могуће објавити на друштвеним мрежама |
| интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама | више људи у оквиру једне просторије може да учествује у истој игри |

Дистрибуција компонената у игри *Dice of Blood* која се игра коришћењем интернет сервиса је представљена на слици 4.2. На интернет сервису се налази једина инстанца компоненте „Логика окружења“, а постоје и две различите клијентске апликације. Једна од клијентских апликација се користи за управљање играчима и приказ приватних података и најчешће се извршава на мобилним уређајима. Док се друга клијентска апликација користи искључиво за приказ јавних информација и најчешће извршава на дигиталним ТВ пријемницима. Компонента „Логика окружења“ је одговорна за логику игре. На клијенту са јавним приказом налази се инстанца компоненте „Приказ“ која служи за приказ целе мапе и свих њених јавних делова, а на клијентима који управљају играчима имамо компоненту „Приказ“ која служи за приказ играчевог приватног погледа на игру. Компонента „Препознавање покрета“ се користи приликом бацања коцкица у игри и омогућава кориснику да коришћењем свог мобилног уређаја симулира виртуелно мућкање и бацање коцкице. Компонента „Друштвени медији“ се користи како би се резултати игре објавили на друштвеним мрежама (*Facebook* или *Twitter*). Компонента „Комуникација“ је присутна на свим уређајима омогућавајући им размену података. Компонента „ТВ сервис“ се не користи.




Слика 4.2 Дистрибуција компонената у игри *Dice of Blood* која се игра коришћењем интернет сервиса

Дистрибуција компонената у игри *Dice of Blood* која се игра у оквиру локалне кућне мреже је представљена на слици 4.3. У овом случају постоје две апликације, серверска која се најчешће извршава на дигиталним ТВ пријемницима и клијентска апликација која се најчешће извршава на мобилним уређајима. На серверу се налазе компонента „Логика окружења“ на којој се извршава главна логика игре и компонента „Приказ“ која служи за графички приказ целе мапе игре. На клијенту се налазе компоненте „Приказ“, „Друштвени медији“ и „Детекција покрета“ које су реализоване на исти начин као и у случају клијентске апликације за управљање код *Dice of Blood* игре развијене коришћењем интернет сервиса. Компонента „Комуникација“ је присутна на свим уређајима и у овом случају омогућава размену података у оквиру локалне кућне мреже.

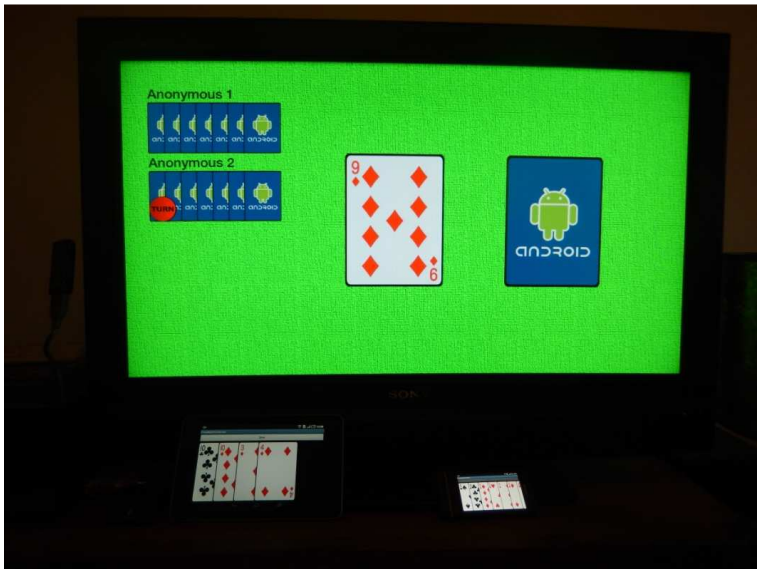



Слика 4.3 Дистрибуција компонената у игри *Dice of Blood* која се игра у оквиру локалне кућне мреже

## 4.2 Игра *Crazy Eights*

Игра картама, која се зове *Crazy Eights*, се игра стандардним шпилом од 52 карте, а број играча може бити од 2 до 4. У случају када играју два играча свакоме се дели по седам карата, а у случају када играју три или четири играча сваки добија по пет карата. Остатак шпила се ставља на сто лицем на доле и одатле играчи касније узимају карте. Прва карта са шпила се окреће лицем на горе и представља почетак талона на који ће се избацивати карте. Играчи избацују карте на талон лицем на горе и то, или карту са истим бројем или карту у истом знаку, у односу на ону која је последња избачена на талон. Карта са бројем осам може увек

да се игра и играч који је одигра захтева боју у којој мора бити следећа карта. Уколико нема одговарајућу карту коју може да избаци на талон играч је дужан да узме једну карту са шпила. Играч, уколико жели да тактизира, може да узме карту са шпила иако има код себе одговарајућу карту за избацивање. Победник партије је онај играч који први избаци све карте. Игру је могуће играти и у два тима од по два играча. У том случају је победник онај тим чија оба играча први остану без карата. Игра је приказана на слици 4.4.




Слика 4.4 Игра *Crazy Eights*

У оквиру игре *Crazy Eights* корисницима су пружени следећи сценарији интеракције: подела приказа на више екрана, употреба мобилног уређаја за управљање акцијама на централном екрану, додатне чулне повратне информације, корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака и интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама. Сценарији интеракције човек-рачунар у интегрисаном окружењу и начин на коју су подржани у оквиру *Crazy Eight* игре развијене коришћењем интернет сервиса су представљени у табели 4.3, а за игру *Crazy Eights* развијену коришћењем локалне кућне мреже су представљени у табели 4.4.

Табела 4.3 Сценарији интеракције човек-рачунар у интегрисаном окружењу и начин на који су подржани код *Crazy Eights* игре развијене коришћењем интернет сервиса

| **Сценарији интеракције** | **Игра *Crazy Eights* развијена коришћењем интернет сервиса** |
|---|---|
| подела приказа на више екрана | ТВ екрани су искоришћени за приказ карата на талону, а екрани мобилних уређаја за приказ карата у руци |
| употреба мобилног уређаја за управљање акцијама на централном екрану | управљање картама |
| детекција покрета коришћењем сензора | - |
| додатне чулне повратне информације | вибрацијом мобилног уређаја играч се обавештава да је на потезу |
| пасивно гледање телевизије обогаћено паралелним интерактивним активностима | - |
| корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака | резултате игре је могуће објавити на друштвеним мрежама |
| интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама | више људи у оквиру једне просторије учествује у истој игри са људима који су смештени у другој просторији |

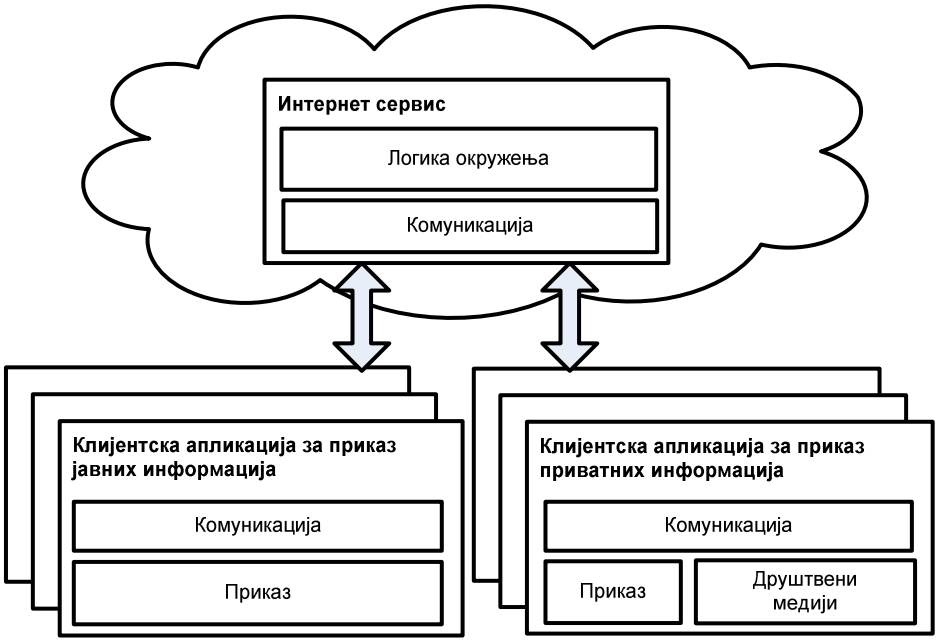
Приказ игре је подељен на јавне ТВ екране и приватне екране мобилних уређаја. Велики ТВ екран је искоришћен за приказ карата на талону, броја карата које сваки играч држи у руци и ко је на потезу. Мали екрани мобилних уређаја су искоришћени за приказ и управљање картама у руци. Резултати игре се објављују на друштвеним мрежама пружајући могућност и пријатељима који нису директно укључени у игру да буду информисани о исходу игре. Смештањем логике игре на интернет омогућава се слично као и код *Dice of Blood* игре да пријатељи који су физички смештени у различитим дневним собама интерреагују играјући исту игру.

Табела 4.4 Сценарији интеракције човек-рачунар у интегрисаном окружењу и начин на који су подржани код *Crazy Eights* игре развијене у оквиру локалне кућне мреже

| **Сценарији интеракције** | **Игра *Crazy Eights* у оквиру локалне кућне мреже** |
|---|---|
| подела приказа на више екрана | ТВ екрани су искоришћени за приказ карата на талону, а екрани мобилних уређаја за приказ карата у руци |
| употреба мобилног уређаја за управљање акцијама на централном екрану | управљање картама |
| детекција покрета коришћењем сензора | - |
| додатне чулне повратне информације | вибрацијом мобилног уређаја играч се обавештава да је на потезу |
| пасивно гледање телевизије обогаћено паралелним интерактивним активностима | - |
| корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака | резултате игре је могуће објавити на друштвеним мрежама |
| интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама | више људи у оквиру једне просторије може да учествује у истој игри |

Дистрибуција компонената у игри *Crazy Eights* која се игра коришћењем интернет сервиса је представљена на слици 4.5. На интернет сервисима се налази једина инстанца компоненте „Логика окружења“, а постоје и две клијентске апликације. Једна од клијентских апликација се користи за приказ јавних информација и најчешће се извршава на дигиталним ТВ пријемницима. Док се друга клијентска апликација користи за приказ приватних информација и најчешће се извршава на мобилним уређајима. Компонента „Логика окружења“ је одговорна за логику игре. На клијенту са јавним приказом налази се инстанца компоненте „Приказ“ која служи за приказ карата на талону. На клијентима са

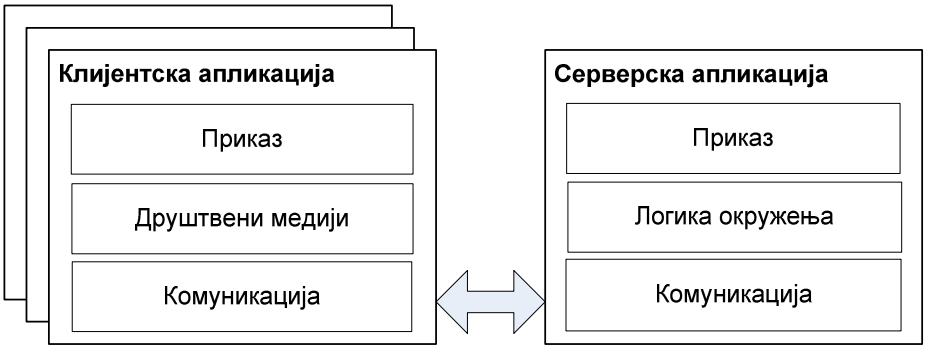
приватним приказом имамо компоненту „Приказ“ која служи за приказ карата у руци. Компонента „Друштвени медији“ се користи како би се резултати игре објавили на друштвеним мрежама (*Facebook* или *Twitter*), а „Комуникациона“ компонента је присутна у свим уређајима омогућавајући им размену података. Компоненте „Препознавања покрета“ и компонента „ТВ сервис“ се у овој игри не користе.




Слика 4.5 Дистрибуција компонената у игри *Crazy Eights* која се игра коришћењем интернет сервиса

Дистрибуција компонената у игри *Crazy Eights* која се игра у оквиру локалне кућне мреже је представљена на слици 4.6. У овом случају постоје две апликације, серверска која се најчешће извршава на дигиталним ТВ пријемницима и клијентска апликација која се најчешће извршава на мобилним уређајима. На серверу се налазе компонента „Логика окружења“ на којој се извршава главна логика игре и компонента „Приказ“ која служи за графички приказ карата на талону. На клијенту се налазе компоненте „Приказ“ и „Друштвени медији“ које су реализоване на исти начин као и у случају клијентске апликације за управљање код *Crazy Eights* игре развијене коришћењем интернет сервиса. Компонента

„Комуникација“ је присутна на свим уређајима и у овом случају омогућава размену података у оквиру локалне кућне мреже.




Слика 4.6 Дистрибуција компонената у игри *Crazy Eights* која се игра у оквиру локалне кућне мреже

## 4.3 Игра *Egg-and-Spoon*

Игра *Egg-and-spoon* је микро игра у реалном времену коју је могуће играти у току трајања ТВ програма или у паузи између гледања програма, на пример док трају рекламе. Игра представља трку између два играча од којих сваки носи по једно јаје у кашици. Трка се завршава у случају када неко од играча стигне први до циља са јајетом на кашици или у тренутку када неком од играча испадне јаје. На ТВ екрану се приказује тркачка стаза, тркачи и време преостало до краја трке, а на сваком мобилном уређај се приказују јаје на кашици и два дугмета. Играч мора да што брже наизменично притиска дугмиће како би покретао тркача, а у исто време мора и да врши балансирање јајета на кашици [40] . Игра *Egg-and-spoon* је приказан на слици 4.7.

У оквиру игре *Egg-and-spoon* корисницима су пружени следећи сценарији интеракције: подела приказа на више екрана, употреба мобилног уређаја за управљање акцијама на централном екрану, детекција покрета коришћењем сензора, додатне чулне повратне информације, пасивно гледање телевизије обогаћено паралелним интерактивним активностима, корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака. Сценарији интеракције човек-рачунар у интегрисаном

окружењу и начин на који су подржани у оквиру *Egg-and-spoon* игре су представљени у табели 4.5.


Слика 4.7 Игра *Egg-and-spoon*

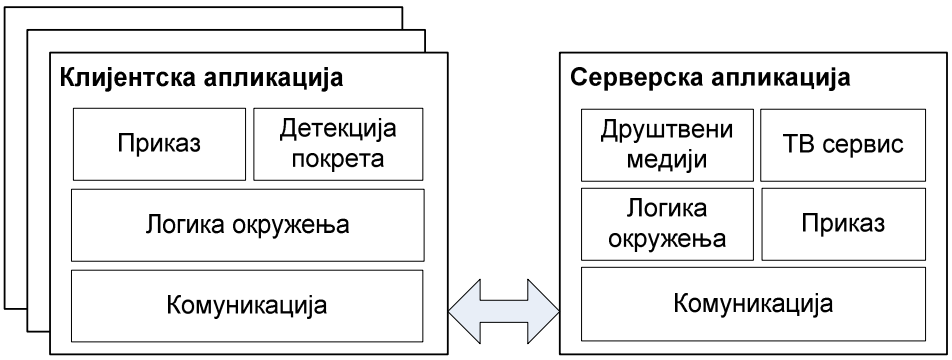
Приказ игре је подељен на јавне ТВ екране и приватне екране мобилних уређаја. На великом ТВ екрану се на слоју преко телевизијског у доњој четвртини приказује тркачка стаза, тркачи који носе јаје на кашици и преостало време за завршетак трке. Мали екрани мобилних уређаја су искоришћени за приказ друге мини игре која служи за балансирање јајета на кашици и чији исход директно утиче на резултат главне тркачке игре. На мобилним уређајима се осим јајета у кашици приказују и два дугмета која служе за управљање тркачима на великом ТВ екрану. Вибрација мобилног уређаја обавештава играча да му је јаје испало из кашике и да је изгубио трку. Како се игра извршава у току трајања ТВ емисије пасивно гледање телевизије је обогаћено паралелним интерактивним активностима. Резултат игре, као и у току које ТВ емисије и на ком каналу је игра играна, се објављује на друштвеним мрежама дајући могућност и пријатељима који нису директно укључени у игру да буду информисани.

Табела 4.5 Сценарији интеракције човек-рачунар у интегрисаном окружењу и начин на који су подржани код *Egg-and-spoon* игре развијене у оквиру локалне кућне мреже

| **Сценарији интеракције** | **Игра *Egg-and-spoon*** |
|---|---|
| подела приказа на више екрана | ТВ екрани су искоришћени за приказ тркачке стазе, а екрани мобилних уређаја за приказ „јајета у кашици“ |
| употреба мобилног уређаја за управљање акцијама на централном екрану | управљање тркачем |
| детекција покрета коришћењем сензора | употребом мобилног уређаја врши се „балансирање јајета на кашици“ |
| додатне чулне повратне информације | вибрацијом мобилног уређаја играч се обавештава да му је „испало јаје“ |
| пасивно гледање телевизије обогаћено паралелним интерактивним активностима | игра се приказује на слоју преко телевизијског програма |
| корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака | резултате игре, као и у току ког програма је играна, је могуће објавити на друштвеним мрежама |
| интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама | више људи у оквиру једне просторије може да учествује у истој игри |

Дистрибуција компонената које се користе за развој ове игре врши се преко локалне кућне мреже и приказана је на слици 4.8. Постоје две одвојене апликације серверска која се најчешће извршава на дигиталним ТВ пријемницима и клијентска апликација која се најчешће извршава на мобилним уређајима. На серверу се налазе компонента „Логика окружења“ на којој се извршава главна логика игре и компонента „Приказ“ која служи за графички приказ тркачке стазе и тркача. Компонента „ТВ сервис“ врши контролу над ТВ емисијом која се приказује у позадини игре, а компонента „Друштвени медији“ је одговорна за проглашавање победника на крају трке, наводећи имена играча и резултате са ТВ

контекстом (на пример, "Пера је победио Мику за време трајања емисије А на ТВ станици Б"). На клијенту се такође налази компонента „Логика окружења“ која служи за имплементацију игре балансирања јајета на кашици. Исход игре на клијенту контролише главну игру која се извршава на серверу. Компонента „Детекција покрета“ се користи за балансирање јајета на кашици, а компонента „Приказ“ за приказ света игре балансирања који се састоји од јајета, кашике и дугмића. Компонента „Комуникација“ се налази и на серверској и на клијентској апликацији омогућавајући размену података.



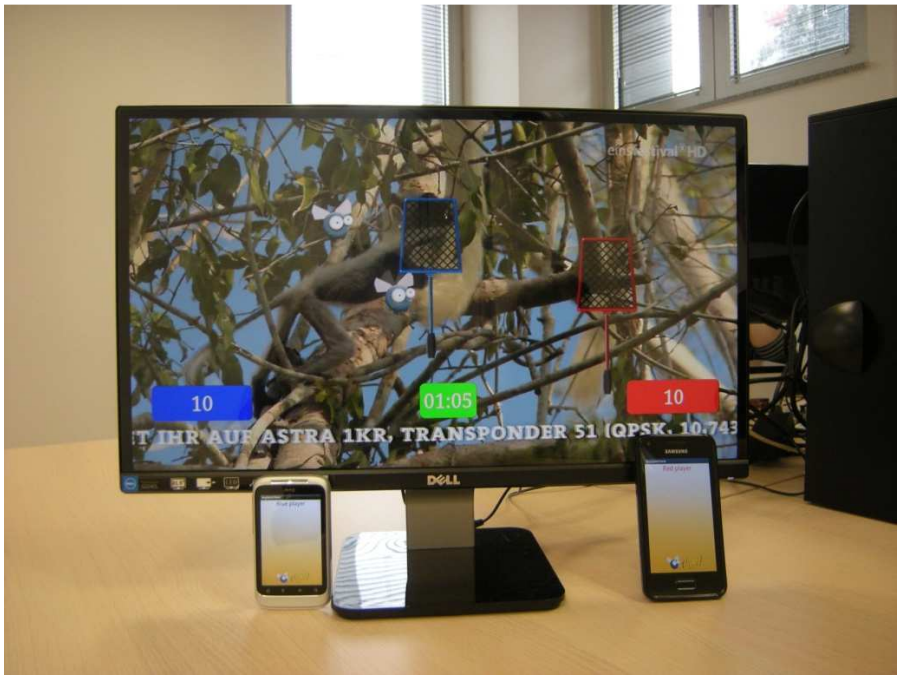

Слика 4.8 Дистрибуција компонената у игри *Egg-and-spoon* која се игра у оквиру локалне кућне мреже

## 4.4 Игра *Flyswatter*

Игра *Flyswatter* представља акцију ловљења мува која се одиграва за време гледања телевизијског програма. Игра се одвија на слоју изнад ТВ програма како би корисник једним делом и даље могао да прати програм. На екрану телевизора се приказују освојени поени сваког од играча, преостало време, две мухоловке, за сваког играча по једна и две муве које се насумично крећу по екрану [38]. На екрану телефона се приказује ознака који је играч. Мухоловком на великом ТВ екрану се управља померањем прста по екрану телефона. Циљ играча је да мухоловке са великог ТВ екрана постави преко муве, након тога коришћењем телефона симулира „ударац“ и на тај начин убије муву. Мува ни у једном тренутку не престаје да се креће тако да играч мора да буде брз и прецизан. Када играч успе да убије муву, на екрану се приказује спљоштена мува и играч добија 10 поена. Након пар секунди, спљоштена мува нестаје са екрана а уместо ње се

појављује нова са насумичног места на екрану. Циљ игре је да се убије што више мува за време од 2 минута а победник је играч који убије највише мува односно има највише поена. Игра *Flyswatter* је приказана на слици 4.9.

.




Слика 4.9 Игра *Flyswatter*

У оквиру игре *Flyswatter* корисницима су пружени следећи сценарији интеракције: употреба мобилног уређаја за управљање акцијама на централном екрану, детекција покрета коришћењем сензора, додатне чулне повратне информације, пасивно гледање телевизије обогаћено паралелним интерактивним активностима, корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака. Сценарији интеракције човек-рачунар у интегрисаном окружењу и начин на који су подржани у оквиру *Flyswatter* игре су представљени у табели 4.6. Преко целог ТВ екрана на слоју преко телевизијског приказује се игра које се састоји од мува које се насумично крећу по екрану и мухоловки. Екрани мобилних уређаја осетљиви на додир су искоришћени за управљање мухоловком, а симулацијом „ударца“ коришћењем мобилног уређаја могуће је контролисати ударац мухоловке на

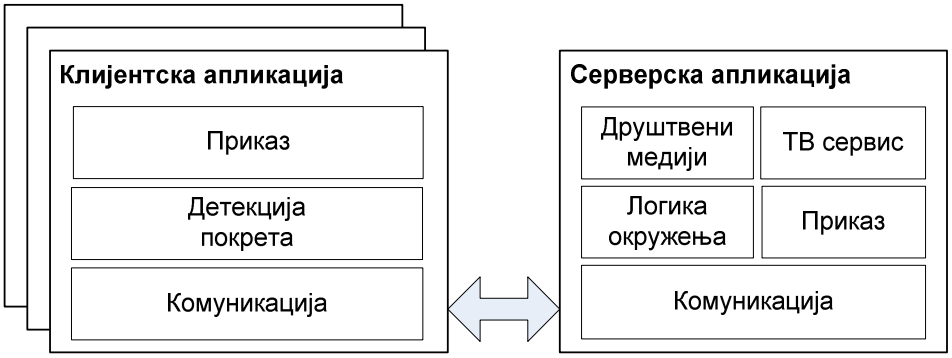
великом ТВ екрану. У случају када је ударац муве био успешан, играч се обавештава и вибрацијом мобилног уређаја. Како се игра извршава у току трајања ТВ емисије пасивно гледање телевизије је обогаћено паралелним интерактивним активностима. Резултат игре, као и у току које ТВ емисије и на ком каналу је игра играна се објављује на друштвеним мрежама, дајући могућност и пријатељима који нису директно укључени у игру да буду информисани.

Табела 4.6 Сценарији интеракције човек-рачунар у интегрисаном окружењу и начин на који су подржани код *Flyswatter* мреже

| **Сценарији интеракције** | **Игра *Flyswatter*** |
|---|---|
| подела приказа на више екрана | - |
| употреба мобилног уређаја за управљање акцијама на централном екрану | управљање мухоловкама |
| детекција покрета коришћењем сензора | употребом мобилног уређаја детектује се покрет „ударца“ |
| додатне чулне повратне информације | вибрацијом мобилног уређаја играч се обавештава да је ударац муве био успешан |
| пасивно гледање телевизије обогаћено паралелним интерактивним активностима | игра се приказује на слоју преко телевизијског програма |
| корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака | резултате игре, као и у току ког програма је играна, је могуће објавити на друштвеним мрежама |
| интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама | више људи у оквиру једне просторије може да учествује у истој игри |

Дистрибуција компонената које се користе за развој ове игре врши се преко локалне кућне мреже и приказана је на слици 4.10. Постоје две одвојене апликације серверска која се најчешће извршава на дигиталним ТВ пријемницима

и клијентска апликација која се најчешће извршава на мобилним уређајима. На серверу се налазе компонента „Логика окружења“ на којој се извршава главна логика игре и компонента „Приказ“ која служи за графички приказ игре. Компонента „ТВ сервис“ врши контролу над ТВ емисијом која се приказује у позадини игре, а компонента „Друштвени медији“ је одговорна за проглашавање победника на крају игре, наводећи имена играча, број убијених мува и у току које емисије се игра одиграла. На клијентској апликацији се компонента „Детекција покрета“ користи за препознавање покрета ударца. Компонента „Комуникација“ се налази и на серверској и на клијентској апликацији омогућавајући размену података.




Слика 4.10 Дистрибуција компонената у игри *Flyswatter* која се игра у оквиру локалне кућне мреже

## *4.5* Игра *Tomato rating*

Игра названа *Tomato rating* је такође осмишљена да се игра у току трајања ТВ програма. Када је на програму филм, серија или сцена која се кориснику не допада он има могућност да гестом симулира „бацање“ парадајза на програм који гледа. У зависности од јачине „бацање“ на телевизору се на слоју преко ТВ програма приказује од једног до пет парадајза. Што је „бацање“ јачег интензитета на телевизору ће се приказати већи број парадајза. Опционо, корисник има могућност да у зависности од броја бачених парадајза оцени програм који

тренутно гледа у јавној TMDb бази филмова. Игра *Tomato rating* је приказана на слици 4.11.

Слика 4.11 Игра *Tomato rating*

У оквиру игре *Tomato rating* корисницима су пружени следећи сценарији интеракције: употреба мобилног уређаја за управљање акцијама на централном екрану, детекција покрета коришћењем сензора, додатне чулне повратне информације, пасивно гледање телевизије обогаћено паралелним интерактивним активностима, корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака. Сценарији интеракције човек-рачунар у интегрисаном окружењу и начин на који су подржани у оквиру *Tomato rating* игре су представљени у табели 4.7. Мобилни уређаји су искоришћени за препознавање јачине геста „бацања“ и на основу тога на великом ТВ екрану се приказује одређен број бачених парадајза. Када се на ТВ екрану прикажу „бачени“ парадајзи играчу вибрира мобилни телефон, као повратна информација. Како је парадајзе могуће бацити у току трајања ТВ емисије и приказују се на слоју преко телевизијског пасивно гледање телевизије је обогаћено паралелним интерактивним активностима. Из дигиталног видео тока могуће је доћи до информације која емисија се тренутно гледа и на

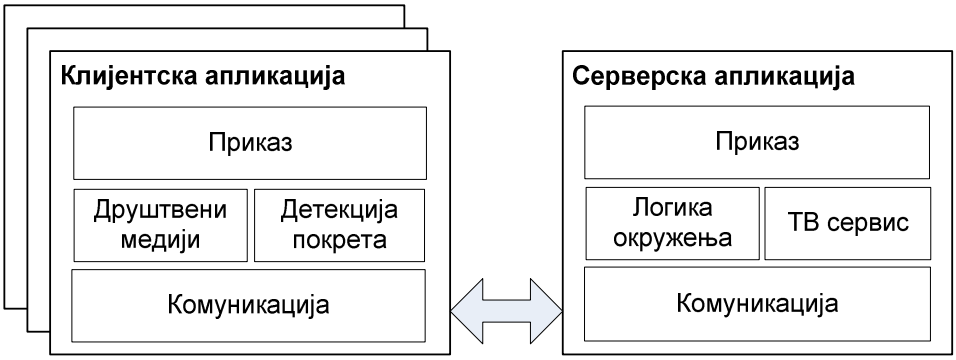
основу тога и резултата игре оцену гледаног програма је могуће унети у јавну базу филмова.

Табела 4.7 Сценарији интеракције човек-рачунар у интегрисаном окружењу и начин на који су подржани код *Tomato rating* мреже

| **Сценарији интеракције** | **Игра *Tomato rating*** |
|---|---|
| подела приказа на више екрана | - |
| употреба мобилног уређаја за управљање акцијама на централном екрану | управљање парадајзима |
| детекција покрета коришћењем сензора | употребом мобилног уређаја детектује се покрет „бацања“ |
| додатне чулне повратне информације | вибрацијом мобилног уређаја играч се обавештава да је одређен број парадајза „бачен“ |
| пасивно гледање телевизије обогаћено паралелним интерактивним активностима | игра се приказује на слоју преко телевизијског програма |
| корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака | из дигиталног видео тока се долази до информације која емисија се тренутно гледа и на основу тога и резултата игре оцену гледаног програма је могуће унети у јавну базу филмова |
| интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама | више људи у оквиру једне просторије може да учествује у истој игри |

Дистрибуција компонената које се користе за развој ове игре врши се преко локалне кућне мреже и приказана је на слици 4.12. Постоје две одвојене апликације серверска која се најчешће извршава на дигиталним ТВ пријемницима и клијентска апликација која се најчешће извршава на мобилним уређајима. На серверу се налазе компонента „Логика окружења“ на којој се извршава главна логика игре и компонента „Приказ“ која служи за графички приказ парадајза на

слоју преко ТВ програма. Компонента „ТВ сервис“ врши контролу над ТВ емисијом која се приказује у позадини игре. На клијентској апликацији се компонента „Детекција покрета“ користи за препознавање покрета ударца као и његове јачина. Коришћењем компоненте „Друштвени медији“ оцена филма, која је обрнуто пропорционална броју бачених парадајза се објављује на TMDb сервису. Компонента „Комуникација“ се налази и на серверској и на клијентској апликацији и омогућава размену података.




Слика 4.12 Дистрибуција компонената у игри *Tomato rating* која се игра у оквиру локалне кућне мреже

## 4.6 Анализа искуства корисника приликом употребе ТВ центричних игара

Како би пружили увид у искуства приликом употребе апликација које покривају идентификоване сценарије интеракције човек-рачунар у интегрисаном окружењу, развијене ТВ центричне игре је испитало 59 волонтера. Испитивање је извршено у две фазе. У оквиру прве фазе поређена су искуства приликом играња потезних друштвених игара (*Dice of Blood* и *Crazy Eights*) развијених у оквиру локалне кућне мреже и приликом играња потезних друштвених игара развијених коришћењем интернет сервиса. Ове игре су такође поређене и са традиционалним играма на табли и друштвеним играма које се играју коришћењем десктоп рачунара. У оквиру друге фазе испитано је искуство корисника приликом играња микро игара у реалном времену (*Egg-and-Spoon*, *Flyswatter* и *Tomato Rating*) које

су се приказивале на слоју преко телевизијског програма. У оба случаја испитивање је спроведено прикупљањем података из упитника и одговора на питања из отворених интервјуа [64].

У тестирању је учествовало 59 волонтера, старости између 25 и 40 година. Сви учесници су имали претходна искуства у игрању класичних друштвених игара на табли и игара картама. Учесници су такође имали и повремена искуства у игрању игара на десктоп рачунарима, играчким конзолама и на мобилним уређајима. Такође, свако од испитаника уобичајено гледа телевизију барем један сат дневно.

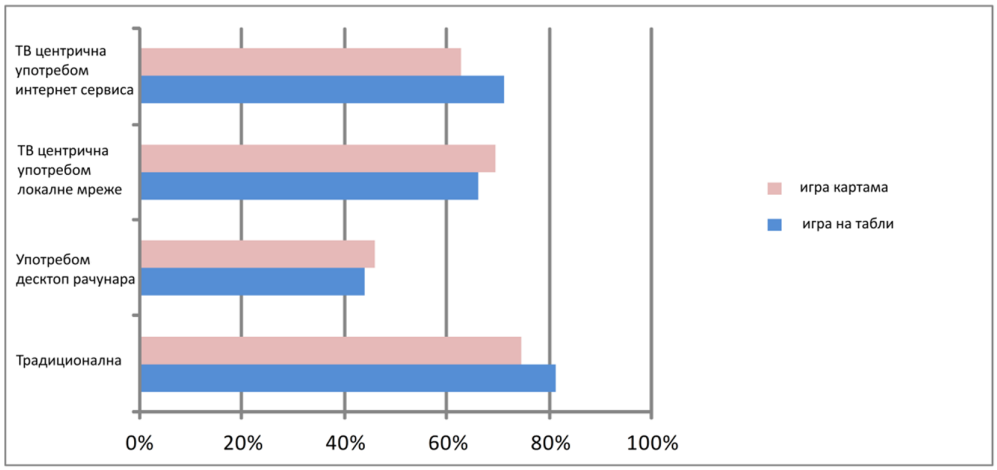
За испитивање игара коришћене су две просторије које су поседовале дигиталне ТВ пријемнике. Разлог због ког су коришћене две просторије је раздвајање две групе играча који учествују у истој партији игре развијене коришћењем интернет сервиса. Сви коришћени мобилни уређаји су били сличних хардверских карактеристика, што је важило и за све коришћене дигиталне ТВ пријемнике. Свака игра је обезбеђивала минимум од 30 фрејмова у секунди [18]. Обезбеђено је и да време између акције играча и резултата акције приказаном на ТВ екрану буде максимално 100ms за игре у реалном времену, односно 500 ms за потезне игре [17].

Испитивање искуства корисника приликом играња игре *Dice of Blood* и игре *Crazy Eights* у оквиру локалне мреже и коришћењем интернет сервиса је извршено у групама од по 3 или 4 особе. Особе у оквиру исте групе су се претходно познавале. Свака група је играла сваку од игара у различитим петнаестоминутним сесијама. Процедура се састојала у поређењу искуства приликом играња ТВ центричне игре у оквиру локалне кућне мреже и игре коришћењем интернет сервиса са искуством приликом играња потезних игара на традиционалан начин, као и коришћењем десктоп рачунара. Након сваке сесије учесници су требали да оцене колико се слажу са тврдњом о *корисничком искуству*: „Уживао сам играјући игру“ на Ликерт скали од 1 до 5 (1. уопште се не слажем, 2. не слажем се, 3. немам мишљење, 4. слажем се, 5. потпуно се слажем). Након сваке сесије са свим члановима групе је спроведен и интервју отвореног типа.

Искуство приликом играња сваке од три микро игре је испитано тако што су по два учесника играли сваку од игара у одвојеним петнаестоминутним сесијама.

Приликом играња на телевизији се приказивао документарни програм за игру *Egg-and-Spoon* и *Flyswatter*, а приликом играња *Tomato Rating* приказиван је филмски програм. Након сваке сесије учесници су требали да оцене колико се слажу са тврдњом о *корисничком искуству*, са додатком тврдње о *могућношћу праћења ТВ програма*: „У стању сам да другој особи препричам шта сам гледао на телевизији док сам играо игру“. Тврдња о *могућношћу праћења ТВ програма* је искоришћена како би се испитало колико је особа способна да истовремено игра игру и прати шта се приказује на телевизији. Након сваке сесије са учесницима је обављен интервју отвореног типа.

Резултати испитивања корисника приликом играња потезних игара су сакупљени и анализирани одређивањем процената људи који су се сложили са тврдњом о *корисничком искуству*, односно одабрали одговоре „слажем се“ и „у потпуности се слажем“ за сваку варијанту игара *Dice of Blood* и *Crazy Eights*. На слици 4.13 на x оси је представљен проценат учесника који се сложио са тврдњом о *корисничком искуству*, а на у оси су представљене све верзије потезних игара на табли и игара картама.




Слика 4.13 Проценат учесника који се сложио са тврдњом о *пријатном играчком искуству* за потезне игре

Резултати су показали да су испитаници више уживали у игрању традиционалних верзија и игара на табли и картама у односи на дигиталне верзије. Ипак, ТВ центрични концепт је оцењен повољније у односу на потезне

игре које се играју коришћењем десктоп рачунара, показујући да се по питању корисничког искуства приближио традиционалном начину играња. Ако поредимо међусобно потезне ТВ центричне игре, испитаници су више уживали у игри картама у оквиру локалне кућне мреже, док им је за игру на табли било пријатније приликом играња коришћењем интернет сервиса.

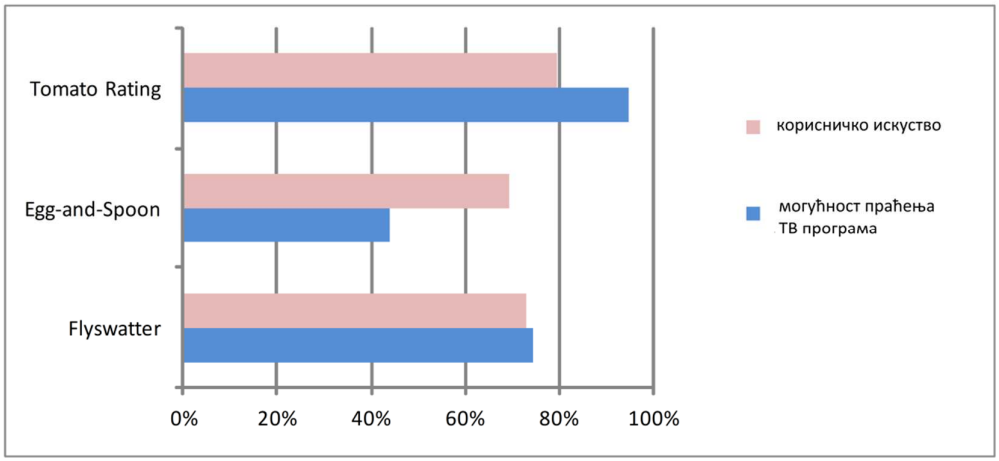
Приликом спровођења отворених интервјуа различити испитаници су имали сличне коментаре. Испитаници су напоменули да им у дигиталним верзијама игара највише недостају физички елементи традиционалних игара, као што је мешање карата и бацање праве коцкице, иако су се ТВ центричне верзије игара приближиле традиционалним омогућавајући на пример да корисник „држи карте у рукама“. Предности које су дате дигиталним верзијама игара су биле анимације које су омогућиле да свет игре не буде статичан. Такође, код ТВ центричних игара социјална интеракција је остала не промењена у односу на традиционалне верзије игара, док је код игара коришћењем десктоп рачунара то изгубљено. Код ТВ центричних верзија игара испитаници су напоменули да су уживали у комуникацији са осталим играчима, као и могућношћу да посматрају изразе лица осталих играча у соби приликом играња ТВ центричних верзија.

Резултати испитивања корисника приликом играња микро игара су сакупљени и анализирани одређивањем процената људи који су се сложили са тврдњом о *пријатном корисничком искуству*, као и процената људи који су се сложили са тврдњом о *могућношћу праћења ТВ програма*. На слици 4.14 је на x оси представљен проценат учесника који се сложио са обе тврдње, а на у оси су приказане три микро игре у реалном времену.

Већина испитаника (између 69% и 79%) се сложила са тврдњом о *пријатном корисничком искуству*. Испитаници су највише уживали играјући *Tomato Rating* игру. Такође, у току *Tomato Rating* игре су имали и најмање проблема да прате ТВ програм. Елементи игре који се приказују на ТВ екрану у случају *Tomato Rating* игре су минимални, а и циљ игре је у директној вези за садржајем ТВ програма.

Искуство приликом играња *Egg-and-Spoon* игре испитаници су оценили као најмање пријатно у поређењу са остале две игре. Највећа замерка испитаника је била подела пажње на екране мобилних уређаја и ТВ екрана што је проузроковало отежано праћење ТВ програма. Такође, корисници су чак и када су гледали у ТВ

екран више били фокусирани на доњи део екрана, на ком се игра одвијала. Више корисника је изјавило да би радије играли *Egg-and-Spoon* игру у току реклама или када чекају да се на телевизији појави емисија од значаја.




Слика 4.14 Проценат испитаника који су се сложили са тврдњама о *пријатном корисничком искуству* и *могућношћу праћења ТВ програма* за микро игре у реалном времену

У неколико случаја док су играли *Flyswatter* игру испитаници су ТВ програм посматрали као део игре, укључивањем објеката које виде на ТВ програму у игру, покушавајући на пример да их „ударе“ мухоловком. Испитаницима се свидело што је за игру искоришћен цео телевизијски екран назначавајући и да им графика игре није заклонила телевизијски програм.

# 5. ВЕРИФИКАЦИЈА СОФТВЕРСКЕ ПЛАТФОРМЕ

Игра на табли *Dice of Blood* и микроигра *Flyswatter* су првобитно развијене без употребе софтверске платформе. Након развоја SHARP софтверске платформе, ове две игре су развијене поново коришћењем платформе што је омогућило поређење различитих имплементација ових игара. У оквиру ове главе прво ће се представити резултати поређења величине, сложености и густине сложености апликација развијених без и са коришћењем SHARP софтверске платформе. Након тога ће бити приказани резултати поређења времена одзива апликација развијених са и без коришћења платформе.

## 5.1 Поређење величине, сложености и густине сложености апликација развијених без и са коришћењем SHARP софтверске платформе

Како би се боље представиле разлике у величини и сложености апликација за игре *Dice of Blood* и *Flyswatter* прво ће се представити резултати поређења целокупног програмског кода апликација развијених са и без употребе платформе. Након тога, посебно за игру *Dice of Blood* и посебно за игру *Flyswatter*, поредиће се величина, сложеност и густина сложености програмског кода, категоризованог по свакој од коришћених компонената софтверске платформе.

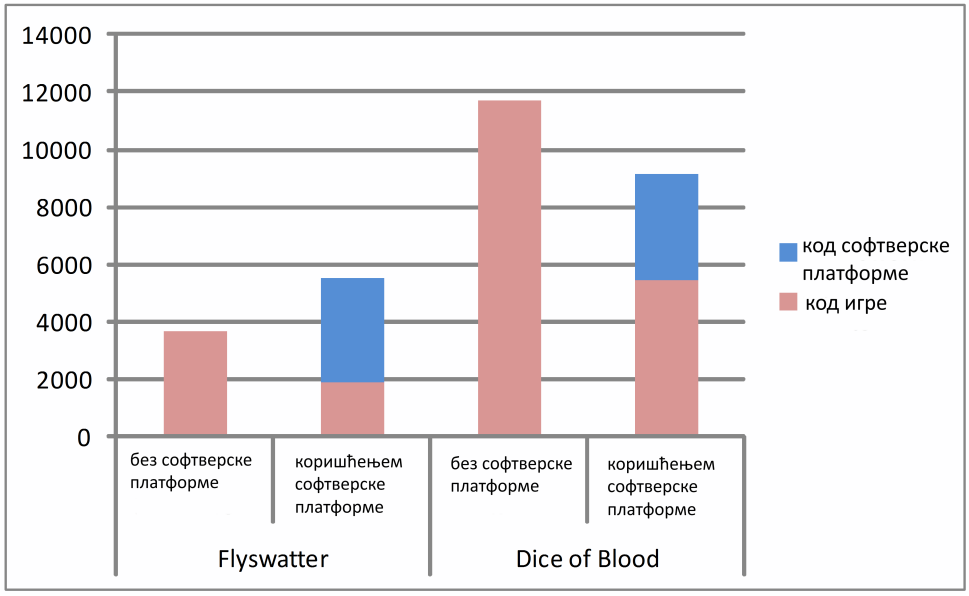
### 5.1.1 Поређење величине и сложеност целокупног програмског кода развијених апликација

Како игре развијене без употребе софтверске платформе садрже подскуп функционалности игара развијених коришћењем софтверске платформе, са посебном пажњом се приступило обезбеђивању коректног поређења. За сваку од имплементација игара, укупан Јава код клијентских и серверских апликација је прикупљен и груписан. За верзије игара које користе софтверску платформу

програмски код је подељен на код који припада самој софтверској платформи и на додатни код специфичан за ту игру. Такође, само класе платформе које су коришћене у конкретној имплементацији игара су укључене у поређење. На пример, игра *Flyswatter* не користи ни једну од функционалности које се односе на потезне игре, па су те класе искључене из поређења у категорији кода који припада софтверској платформи. Поређење је вршено само за функционалности које су развијене у обе верзије игара без и са коришћењем софтверске платформе. На пример, верзија игре *Dice of Blood* која је развијена без коришћења софтверске платформе не укључује функционалности које одговарају функционалностима компоненте „Препознавање покрета“ и подржава само комуникацију у оквиру локалне кућне мреже. Из тих разлога компонента „Препознавање покрета“, као и део кода софтверске платформе који подржава комуникацију коришћењем интернет сервиса је искључен из поређења. Код верзије игре *Flyswatter* развијене без коришћења софтверске платформе графика игре није приказивана на слоју преко телевизијског, него преко предефинисаног видео тока, без директног приступа ТВ сервисима. Из тог разлога је компонента „ТВ сервис“ искључена из поређења код ове игре. Алат *LocMetrics* [46] је коришћен за мерење величине, цикломатске сложености и густине сложености кода. Величина кода је мерена као укупан број физичких линија кода без празних линија и коментара.

На слици 5.1 је представљен број линија кода, а на слици 5.2 цикломатска комплексност игара *Flyswatter* и *Dice of Blood* посебно за игре развијене без коришћења софтверске платформе и за игре развијене коришћењем платформе. На слици 5.1 на x оси су представљене различите врсте имплементираних игара, а на y оси је представљен број линија кода. На x оси на слици 5.2 су представљене различите врсте имплементираних игара, а на y оси је представљена цикломатска комплексност. На примеру *Flyswatter*, као и на примеру *Dice of Blood* игре укупна величина и цикломатска комплексност програмског кода игре развијене коришћењем софтверске платформе је мања у односу на величину и цикломатску комплексност кода игре развијене без употребе платформе. За игру *Flyswatter* и величина и цикломатска комплексност кода игре развијене коришћењем платформе представља 53% програмског кода игре развијене без употребе платформе. У случају *Dice of Blood* игре програмски код развијен употребом

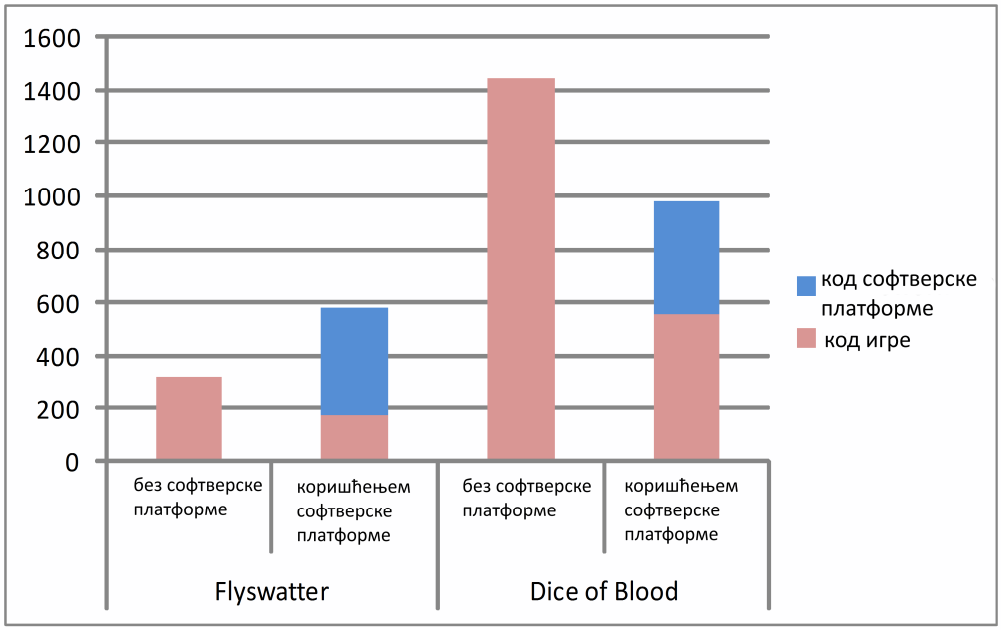
платформе представља 47% програмског кода игре развијене без употребе платформе, док је цикломатска комплексност кода игре развијене употребом платформе 39% цикломатске комплексности кода игре развијене без употребе платформе. У случају када посматрамо програмски код игре *Flyswatter* заједно са кодом платформе, онда он представља 151% величине програмског кода игре развијене без употребе платформе, док је комплексност 179% комплексности игре развијене без употребе платформе. У случају игре *Dice of Blood* ситуација је другачија и програмски код игре заједно са кодом платформе представља 78% програмског кода игре развијене без употребе платформе, док је комплексност 68% комплексности игре развијене без употребе платформе.




Слика 5.1 Број линија кода игара *Flyswatter* и *Dice of Blood* развијених без употребе и коришћењем SHARP софтверске платформе

Већа и комплекснија *Dice of Blood* игра је имала више добити у смањењу величине и комплексности кода у односу на *Flyswatter* игру. Разлика у смањењу величине и комплексности кода ове две игре се може објаснити величином обавезног додатног кода игре која се развија употребом софтверске платформе. Овај код се односи на постављање одговарајућих вредности потребних структура података платформе, као и имплементације потребних интерфејса. Овај код је уочљивији у случају када се софтверска платформа користи за имплементацију

једноставнијих игара. У случају када је игра комплекснија овај додатни код је мање уочљив.




Слика 5.2 Цикломатска комплексност кода игара *Flyswatter* и *Dice of Blood* развијених без употребе и коришћењем софтверске платформе

### 5.1.2 Поређење величине, сложености и густине сложености програмског кода категоризованог по компонентама SHARP софтверске платформе

Категоризација програмског кода за игре развијене са и без SHARP софтверске платформе је извршена на следећи начин. За игре развијене коришћењем софтверске платформе категоризација је извршена у зависности од компоненте платформе која се користила. За игре развијене без коришћења софтверске платформе код је подељен у зависности од задужења одређених секција кода и сврстан је у категорију коју најближе описује. Идеално, један фајл би могао да се сврста у једну категорију, али се дешавало и да различити делови кода у оквиру истог фајла могу да се сврставају у различите категорије. У том случају подела је вршена по методама, јер би свака дубља подела могла имати утицај на цикломатску комплексност. Програмски код који се односи на

иницијализацију апликације и коришћење компоненти графичког корисничког интерфејса је категоризован под компонентом „Приказ“, јер је у најближој релацији са задужењима имплементираним у оквиру компоненте „Приказ“ софтверске платформе.

#### 5.1.2.1 Поређење величине развијених апликација категоризованих по компонентама SHARP софтверске платформе

Величина *Flyswatter* игре, изражена у физичком броју линија кода, по компонентама за сваку од игара развијену без и коришћењем софтверске платформе је представљена у табели 5.1. За *Flyswatter* игру развијену коришћењем софтверске платформе подела је извршена на део кода који се односи на програмски код софтверске платформе, на додатни код који се односи искључиво на *Flyswatter* игру и суму величине та два кода.

Табела 5.1 Величина *Flyswatter* игре развијене без и са коришћењем софтверске платформе

| | ***Flyswatter* игра без софтверске платформе** | ***Flyswatter* игра развијена коришћењем софтверске платформе** | | |
|---|---|---|---|---|
| | | додати код *Flyswatter* игре | код софтверске платформе | код игре заједно са кодом софтверске платформе |
| **Комуникација** | 953 | 468 | 618 | 1086 |
| **Логика окружења** | 664 | 754 | 512 | 1266 |
| **Приказ** | 794 | 619 | 1156 | 1775 |
| **Детекција покрета** | 203 | 59 | 252 | 311 |
| **Друштвени медији** | 1025 | 26 | 1045 | 1071 |
| **Укупно** | 3639 | 1926 | 3583 | 5509 |

У случају компоненте „Комуникација“ ако се у обзир не узме програмски код који директно припада софтверској платформи, део кода који се искључиво односи на додатни код *Flyswatter* игре представља 49% програмског кода који одговара коду који се може сврстати под компоненту „Комуникација“ у оквиру *Flyswatter* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Комуникација“ који припада софтверској платформи, онда је сума кода софтверске платформе и додатног кода који се односи на саму игру за компоненту „Комуникација“ 114% пограмског кода *Flyswatter* игре развијене без употребе софтверске платформе.

Ако посматрамо компоненту „Логика окружења“ и ако се у обзир не узме програмски код који директно припада софтверској платформи, део кода који се искључиво односи на додатни код *Flyswatter* игре представља 113% програмског кода који се може сврстати под компоненту „Логика окружења“ у оквиру *Flyswatter* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Логика окружења“ који припада софтверској платформи, онда је сума кода софтверске платформе и додатног кода који се односи на саму игру за компоненту „Логика окружења“ 190% програмског кода *Flyswatter* игре развијене без употребе софтверске платформе.

Код компоненте „Приказ“ када се у обзир не узме програмски код који директно припада софтверској платформи, део кода који се искључиво односи на додатни код *Flyswatter* игре представља 78% програмског кода који одговара коду који се може сврстати под компоненту „Приказ“ у оквиру *Flyswatter* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Приказ“ који припада софтверској платформи, онда је сума кода софтверске платформе и додатног кода који се односи на саму игру за компоненту „Приказ“ 223% програмског кода *Flyswatter* игре развијене без употребе софтверске платформе.

У случају компоненте „Детекција покрета“ ако се у обзир не узме програмски код који директно припада софтверској платформи, део кода који се искључиво односи на додатни код *Flyswatter* игре представља 29% програмског кода који одговара коду који се може сврстати под компоненту „Детекција покрета“ у оквиру *Flyswatter* игре развијене без употребе софтверске платформе. Ако се у

обзир узме и програмски код компоненте „Детекција покрета“ који припада софтверској платформи, онда је сума кода софтверске платформе и додатног кода који се односи на саму игру за компоненту „Детекција покрета“ 153% програмског кода *Flyswatter* игре развијене без употребе софтверске платформе.

За компоненту „Друштвени медији“ када се у обзир не узме програмски код који директно припада софтверској платформи, део кода који се искључиво односи на додатни код *Flyswatter* игре представља 2,5% програмског кода који одговара коду који се може сврстати под компоненту „Друштвени медији“ у оквиру *Flyswatter* игре развијене без употребе софтверске платформе. Ако се у обзир узме и код компоненте „Друштвени медији“ који припада софтверској платформи, онда је сума кода софтверске платформе и додатног кода који се односи на саму игру за компоненту „Друштвени медији“ 104% програмског кода *Flyswatter* игре развијене без употребе софтверске платформе.

Табела 5.2 Величина *Dice of Blood* игре развијене без и са коришћењем софтверске платформе

| | ***Dice of Blood* игра без софтверске платформе** | ***Dice of Blood* игра развијена коришћењем софтверске платформе** | | |
|---|---|---|---|---|
| | | додати код *Dice of Blood* игре | код софтверске платформе | код игре заједно са кодом софтверске платформе |
| **Комуникација** | 1192 | 305 | 618 | 923 |
| **Логика окружења** | 4850 | 3687 | 858 | 4545 |
| **Приказ** | 4444 | 1460 | 1156 | 2616 |
| **Друштвени медији** | 1204 | 31 | 1045 | 1076 |
| **Укупно** | 11690 | 5483 | 3677 | 9160 |

Величина *Dice of Blood* игре изражена у физичком броју линија кода по компонентама за сваку од игара развијену без и коришћењем софтверске платформе је представљена у табели 5.2. За *Dice of Blood* игру развијену коришћењем софтверске платформе подела је извршена на део програмског кода који се односи на код софтверске платформе, на додатни код који се односи искључиво на *Dice of Blood* игру и суму величине та два кода.

У случају компоненте „Комуникација“ ако се у обзир не узме програмски код који директно припада софтверској платформи, део кода који се искључиво односи на додатни код *Dice of Blood* игре представља 25% програмског кода који одговара коду који се може сврстати под компоненту „Комуникација“ у оквиру *Dice of Blood* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Комуникација“ који припада софтверској платформи, онда је сума кода софтверске платформе и додатног кода који се односи на саму игру за компоненту „Комуникација“ 77% кода *Dice of Blood* игре развијене без употребе софтверске платформе.

Ако посматрамо компоненту „Логика окружења“ и ако се у обзир не узме програмски код који директно припада софтверској платформи, део кода који се искључиво односи на додатни код *Dice of Blood* игре представља 76% програмског кода који се може сврстати под компоненту „Логика окружења“ у оквиру *Dice of Blood* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Логика окружења“ који припада софтверској платформи, онда је сума кода софтверске платформе и додатног кода који се односи на саму игру за компоненту „Логика окружења“ 94% кода *Dice of Blood* игре развијене без употребе софтверске платформе.

Код компоненте „Приказ“ када се у обзир не узме програмски код који директно припада софтверској платформи, део кода који се искључиво односи на додатни код *Dice of Blood* игре представља 33% програмског кода који одговара коду који се може сврстати под компоненту „Приказ“ у оквиру *Dice of Blood* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Приказ“ који припада софтверској платформи, онда је сума кода софтверске платформе и додатног кода који се односи на саму игру за компоненту

„Приказ“ 59% програмског кода *Dice of Blood* игре развијене без употребе софтверске платформе.

За компоненту „Друштвени медији“ када се у обзир не узме програмски код који директно припада софтверској платформи, део кода који се искључиво односи на додатни код *Dice of Blood* игре представља 2,5% програмског кода који одговара коду који се може сврстати под компоненту „Друштвени медији“ у оквиру *Dice of Blood* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Друштвени медији“ који припада софтверској платформи, онда је сума кода софтверске платформе и додатног кода који се односи на саму игру за компоненту „Друштвени медији“ 89% програмског кода *Dice of Blood* игре развијене без употребе софтверске платформе.

Посматрањем величине програмског кода *Flyswatter* и *Dice of Blood* игара развијених без и коришћењем софтверске платформе може се закључити да су одређене функционалности скоро у потпуности пребачене у код софтверске платформе, остављајући програмеру да у коду који је директно везан за одређену игру дода само делове који се односе на иницијализацију. Добар пример представља део кода који се односи на компоненту „Друштвени медији“ код обе игре, *Flyswatter* и *Dice of Blood*, и на компоненту „Детекција покрета“ код игре *Flyswatter*. Једини пример где је број линија кода игре реализоване без коришћења софтверске платформе био мањи него код дела кода који се односи само на игру развијену коришћењем софтверске платформе јесте код компоненте „Логика окружења“ игре *Flyswatter*.

#### 5.1.2.2 Поређење цикломатске комплексности развијених игара категоризованих по компонентама SHARP софтверске платформе

Цикломатска комплексност игре *Flyswatter* за сваку од игара развијену без и коришћењем софтверске платформе је представљена у табели 5.3. За *Flyswatter* игру развијену коришћењем софтверске платформе комплексност кода је подељена на комплексност дела кода који се односи на код софтверске платформе, комплексност додатног кода који се односи искључиво на *Flyswatter* игру и комплексност суме та два кода.

У случају компоненте „Комуникација“ ако се у обзир не узме програмски код који директно припада софтверској платформи, комплексност дела кода који се искључиво односи на додатни код *Flyswatter* игре представља 57% комплексности кода који се може сврстати под компоненту „Комуникација“ у оквиру *Flyswatter* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Комуникација“ који припада софтверској платформи, онда је сума комплексности кода софтверске платформе и комплексности додатног кода који се односи на саму игру за компоненту „Комуникација“ 105% комплексности кода *Flyswatter* игре развијене без употребе софтверске платформе.

Табела 5.3 Цикломатска комплексност *Flyswatter* игре развијене без и са коришћењем софтверске платформе

| | ***Flyswatter* игра без софтверске платформе** | ***Flyswatter* игра развијена коришћењем софтверске платформе** | | |
|---|---|---|---|---|
| | | додати код *Flyswatter* игре | код софтверске платформе | код игре заједно са кодом софтверске платформе |
| **Комуникација** | 81 | 46 | 39 | 85 |
| **Логика окружења** | 74 | 95 | 73 | 168 |
| **Приказ** | 37 | 22 | 149 | 171 |
| **Детекција покрета** | 21 | 7 | 30 | 37 |
| **Друштвени медији** | 111 | 3 | 115 | 118 |
| **Укупно** | 324 | 173 | 406 | 579 |

Ако посматрамо компоненту „Логика окружења“ и не узмемо у обзир програмски код који директно припада софтверској платформи, комплексност дела кода који се искључиво односи на додатни код *Flyswatter* игре представља 128% комплексности кода који се може сврстати под компоненту „Логика

окружења“ у оквиру *Flyswatter* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Логика окружења“ који припада софтверској платформи, онда је сума комплексности кода софтверске платформе и комплексности додатног кода који се односи на саму игру за компоненту „Логика окружења“ 227% комплексности кода *Flyswatter* игре развијене без употребе софтверске платформе.

Код компоненте „Приказ“ када се у обзир не узме програмски код који директно припада софтверској платформи, комплексност дела кода који се искључиво односи на додатни код *Flyswatter* игре представља 59% комплексности кода који одговара коду који се може сврстати под компоненту „Приказ“ у оквиру *Flyswatter* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Приказ“ који припада софтверској платформи, онда је сума комплексности кода софтверске платформе и комплексности додатног кода који се односи на саму игру за компоненту „Приказ“ 460% комплексности кода *Flyswatter* игре развијене без употребе софтверске платформе.

У случају компоненте „Детекција покрета“ ако се у обзир не узме програмски код који директно припада софтверској платформи, комплексност дела кода који се искључиво односи на додатни код *Flyswatter* игре представља 33% комплексности кода који одговара коду који се може сврстати под компоненту „Детекција покрета“ у оквиру *Flyswatter* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Детекција покрета“ који припада софтверској платформи, онда је сума комплексности кода софтверске платформе и комплексности додатног кода који се односи на саму игру за компоненту „Детекција покрета“ 176% кода *Flyswatter* игре развијене без употребе софтверске платформе.

За компоненту „Друштвени медији“ када се у обзир не узме програмски код који директно припада софтверској платформи, комплексност дела кода који се искључиво односи на додатни код *Flyswatter* игре представља 2,7% комплексности кода који одговара коду који се може сврстати под компоненту „Друштвени медији“ у оквиру *Flyswatter* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Друштвени

медији“ који припада софтверској платформи, онда је сума комплексности кода софтверске платформе и комплексности додатног кода који се односи на саму игру за компоненту „Друштвени медији“ 106% комплексности кода *Flyswatter* игре развијене без употребе софтверске платформе.

Цикломатска комплексност игре *Dice of Blood* за сваку од игара развијену без и коришћењем софтверске платформе је представљена у табели 5.4. За *Dice of Blood* игру развијену коришћењем софтверске платформе комплексност кода је подељена на комплексност дела кода који се односи на код софтверске платформе, комплексност додатног кода који се односи искључиво на *Dice of Blood* игру и комплексност суме та два кода.

Табела 5.4 Цикломатска комплексност *Dice of Blood* игре развијене без и са коришћењем софтверске платформе

| | ***Dice of Blood* игра без софтверске платформе** | ***Dice of Blood* игра развијена коришћењем софтверске платформе** | | |
|---|---|---|---|---|
| | | додати код *Dice of Blood* игре | код софтверске платформе | код игре заједно са кодом софтверске платформе |
| **Комуникација** | 120 | 16 | 39 | 55 |
| **Логика окружења** | 638 | 490 | 117 | 607 |
| **Приказ** | 554 | 49 | 149 | 198 |
| **Друштвени медији** | 132 | 3 | 115 | 118 |
| **Укупно** | 1444 | 558 | 420 | 978 |

У случају компоненте „Комуникација“ ако се у обзир не узме програмски код који директно припада софтверској платформи, комплексност дела кода који се искључиво односи на додатни код *Dice of Blood* игре представља 13% комплексности кода који се може сврстати под компоненту „Комуникација“ у

оквиру *Dice of Blood* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Комуникација“ који припада софтверској платформи, онда је сума комплексности кода софтверске платформе и комплексности додатног кода који се односи на саму игру за компоненту „Комуникација“ 46% комплексности кода *Dice of Blood* игре развијене без употребе софтверске платформе.

Ако посматрамо компоненту „Логика окружења“ и не узмемо у обзир програмски код који директно припада софтверској платформи, комплексност дела кода који се искључиво односи на додатни код *Dice of Blood* игре представља 77% комплексности кода који се може сврстати под компоненту „Логика окружења“ у оквиру *Dice of Blood* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Логика окружења“ који припада софтверској платформи, онда је сума комплексности кода софтверске платформе и комплексности додатног кода који се односи на саму игру за компоненту „Логика окружења“ 95% комплексности кода *Dice of Blood* игре развијене без употребе софтверске платформе.

Код компоненте „Приказ“ када се у обзир не узме програмски код који директно припада софтверској платформи, комплексност дела кода који се искључиво односи на додатни код *Dice of Blood* игре представља 8,8% комплексности кода  који одговара коду који се може сврстати под компоненту „Приказ“ у оквиру *Dice of Blood* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте „Приказ“ који припада софтверској платформи, онда је сума комплексности кода софтверске платформе и комплексности додатног кода који се односи на саму игру за компоненту „Приказ“ 35,7% комплексности кода *Dice of Blood* игре развијене без употребе софтверске платформе.

За компоненту „Друштвени медији“ када се у обзир не узме програмски код који директно припада софтверској платформи, комплексност дела кода који се искључиво односи на додатни код *Dice of Blood* игре представља 2,2% комплексности кода  који одговара коду који се може сврстати под компоненту „Друштвени медији“ у оквиру *Dice of Blood* игре развијене без употребе софтверске платформе. Ако се у обзир узме и програмски код компоненте

„Друштвени медији“ који припада софтверској платформи, онда је сума комплексности кода софтверске платформе и комплексности додатног кода који се односи на саму игру за компоненту „Друштвени медији“ 89% комплексности кода *Dice of Blood* игре развијене без употребе софтверске платформе.

Како је цикломатска комплексност у блиској корелацији са величином кода посматрањем игара *Flyswatter* и *Dice of Blood* може се доћи до сличних закључака везаних за комплексност кода као и за величину кода. У случају компонената „Приказ“, „Детекција покрета“ и „Друштвени медији“ комплексност кода софтверске платформе је драстично већа у односу на комплексност самог кода конкретне игре развијене коришћењем платформе. Такође, осим у случају компоненте „Логика окружења“ игре *Flyswatter* комплексност кода који се односи на игру развијену коришћењем софтверске платформе је много мања у односу на комплексност кода за обе игре развијене без коришћења платформе.

#### 5.1.2.3 Поређење густине комплексности развијених игара категоризованих по компонентама SHARP софтверске платформе

Густина комплексности игре *Flyswatter* за сваку од игара развијену без и коришћењем софтверске платформе је представљена у табели 5.5. За *Flyswatter* игру развијену коришћењем софтверске платформе густина комплексности је подељена на густину дела кода који се односи на код софтверске платформе, густину додатног кода који се односи искључиво на *Flyswatter* игру и густину суме та два кода.

Густина комплексности игре *Dice of Blood* за сваку од игара развијену без и коришћењем софтверске платформе је представљена у табели 5.6. За *Dice of Blood* игру развијену коришћењем софтверске платформе густина комплексности кода је подељена на густину дела кода који се односи на код софтверске платформе, густину додатног кода који се односи искључиво на *Dice of Blood* игру и густину суме та два кода.

Густина комплексности сваке од компонената платформе је већа од густине комплексности кода игре *Flyswatter* и игре *Dice of Blood* развијене коришћењем софтверске платформе, што указује на то да код који није део софтверске платформе има мање логичких израза тока контроле по физичкој линији кода од

компонената софтверске платформе. На основу тога се очекује да је већа вероватноћа да се грешке догоде у самој платформи, него у коду развијених игара. Коришћење платформе ће утицати на убрзање времена развоја јер ће за грешке у оквиру софтверске платформе бити довољно да се само једном исправе. Код који се односи само на игре развијене коришћењем софтверске платформе садржи мање линија и мање је комплексан од кода игара развијених без употребе софтверске платформе, па је и за очекивати да је вероватноћа појаве грешака у првом случају мања.

Табела 5.5 Густина комплексности *Flyswatter* игре развијене без и са коришћењем софтверске платформе

| | ***Flyswatter* игра без софтверске платформе** | ***Flyswatter* игра развијена коришћењем софтверске платформе** | | |
|---|---|---|---|---|
| | | додати код *Flyswatter* игре | код софтверске платформе | код игре заједно са кодом софтверске платформе |
| **Комуникација** | 0.085 | 0.098 | 0.063 | 0.078 |
| **Логика окружења** | 0.111 | 0.126 | 0.143 | 0.133 |
| **Приказ** | 0.047 | 0.036 | 0.129 | 0.096 |
| **Детекција покрета** | 0.103 | 0.119 | 0.119 | 0.119 |
| **Друштвени медији** | 0.108 | 0.115 | 0.110 | 0.110 |
| **Укупно** | 0.089 | 0.090 | 0.113 | 0.105 |

Табела 5.6 Густина комплексности *Dice of Blood* игре развијене без и са коришћењем софтверске платформе

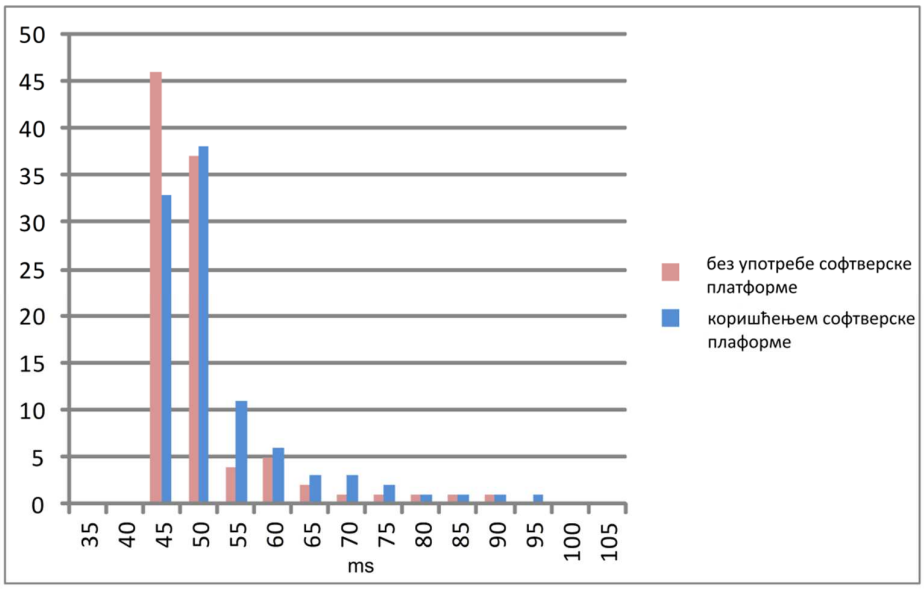
| | ***Dice of Blood* игра без софтверске платформе** | ***Dice of Blood* игра развијена коришћењем софтверске платформе** | | |
|---|---|---|---|---|
| | | додати код *Dice of Blood* игре | код софтверске платформе | код игре заједно са кодом софтверске платформе |
| **Комуникација** | 0.101 | 0.052 | 0.063 | 0.060 |
| **Логика окружења** | 0.132 | 0.133 | 0.136 | 0.134 |
| **Приказ** | 0.125 | 0.034 | 0.129 | 0.076 |
| **Друштвени медији** | 0.110 | 0.097 | 0.110 | 0.110 |
| **Укупно** | 0.124 | 0.102 | 0.114 | 0.107 |

## 5.2 Поређење времена одзива апликација развијених са и без коришћења SHARP софтверске платформе

У оквиру дневне собе, као што је наведено у најновијој литератури [9], могу се издвојити три битне границе за укупно време одзива. У случају препознавања или промене контекста 2s је најдуже прихватљиво кашњење, у случају одговора на команде 0,5s је најдуже прихватљиво кашњење и у случај одзива у реалном времену 0,2s би било најдуже прихватљиво кашњење. Што се тиче кашњења у играма, могу се издвојити две битне границе. За потезне игре 0,5s је најдуже прихватљиво кашњење, док је код неких игара у реалном времену неопходно да кашњење не буде веће од 0,1s [17].

Поређење времена одзива апликација је извршено за случај геста „ударца“ код игре *Flyswatter* развијене без употребе и употребом SHARP софтверске платформе. Тестирање је извршено на 100 узастопних гестова „ударца“ код игре развијене без коришћења платформе и 100 узастопних гестова „ударца“ за случај игре развијене коришћењем софтверске платформе. Као стартни тренутак мерења геста „ударца“ узет је тренутак у којем се играчева рука помера уназад у покушају да направи гест, а као завршни тренутак узет је моменат у коме је мухоловка на ТВ екрану у крајњем положају „ударца“. На слици 5.3 је приказан одзив геста „ударца“ у случају игре развијене без коришћења платформе и у случају игре развијене употребом платформе. На x оси приказани су одзиви у размаку од 5 ms, а на у оси је приказан број тестова чије је време одзива у одговарајућем интервалу.

Од 100 измерених тестова геста „ударца“ у случају игре развијене без коришћења платформе највећи број „удараца“ је имао одзив између 40 и 50 ms (46 је имало одзив између 40 и 45 ms, а 37 „удараца“ је имало одзив између 45 ms и 50 ms). Најдужи забележени одзив је био 90 ms. У случају игре развијене коришћењем софтверске платформе највећи број „удараца“ је имало одзив између 40 и 55 ms (33 је имало одзив између 40 и 45 ms, 38 одзив између 45 и 50 ms, а 11 одзив између 50 и 55 ms). Најдужи забележени одзив је био 95 ms. Просечно време одзива геста „ударца“ у случају игре развијене без коришћења софтверске платформе је 49.7 ms, док је у случају игре развијене коришћењем SHARP софтверске платформе 52.55 ms. Мерења показују да коришћење платформе није увело значајна кашњења. Такође, измерена времена одзива су испод свих утврђених прагова одговора на команде и одзива у играма, указујући на то да је употреба софтверске платформе могућа у наведеним сценаријима.




Слика 5.3 Одзив геста „ударца“ у случају игре развијене без коришћења платформе и у случају игре развијене употребом платформе

# 6. ЗАКЉУЧАК

Проучавањем области интеракције човек-рачунар у окружењу дневне собе са дигиталним ТВ пријемницима и мобилним уређајима установљено је да се постојећи системи у овој области развијају независно, да су ограничене флексибилности и да се углавном фокусирају на једном од могућих сценарија интеракције. У овом раду представљена је SHARP софтверска платформа која пружа програмску подршку за ефикасан и униформан развој дистрибуираних апликација у интегрисаном окружењу дигиталних ТВ пријемника, мобилних уређаја и интернета. Прегледом постојећих система у области интеракције човек-рачунар у оквиру дневне собе идентификовани су следећи могући сценарији интеракције које апликације развијене у интегрисаном окружењу требају да пруже корисницима: подела приказа на више екрана (на мобилним уређајима се приказује додатни садржај као пратећи ономе што је на великом ТВ екрану), употреба мобилног уређаја за управљање акцијама на централном екрану, детекција покрета коришћењем сензора, додатне чулне повратне информације (хаптички одговор, светлосно обавештење), пасивно гледање телевизије обогаћено паралелним интерактивним активностима, корисничко искуство побољшано приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака и интеракција групе људи у оквиру једне просторије са групом људи на другим локацијама. Због своје популарности, масовне употребе и тржишне исплативости SHARP софтверска платформа је примењена за развој ТВ центричних игара које се играју у интегрисаном окружењу нудећи корисницима идентификоване сценарије интеракције.

Развијена SHARP софтверска платформа се састоји од шест компонената: „Логика окружења“, „Приказ“, „Комуникација“, „ТВ сервис“, „Друштвени медији“ и „Детекција покрета“. Компонента „Логика окружења“ описује главну логику виртуелног окружења, које је названо дељени простор, у које корисници могу да се укључе у циљу међусобне интеракције и интеракције са окружењем. Као подврсту дељеног простора могуће је дефинисати свет игре. У случају игре, компонента „Логика окружења“ дефинише логику игре, правила која важе у игри

и хијерархију различитих типова игара: потезне (игре на табли, игре картама) и у реалном времену. Компонента „Приказ“ омогућава поглед на дељени простор. Коришћењем ове компоненте могуће је постићи поделу приказа на више екрана. Код игара ова компонента омогућава исцртавање елемената игре на екранима дигиталних ТВ пријемника и мобилних уређаја. Такође, компонента „Приказ“ омогућава да се на различитим групама екрана исцртавају различити елементи. На примеру игара, може се креирати игра код које се јавни садржај (карте на столу, тркачка стаза) приказује на ТВ екранима, док се екрани мобилних уређаја користе за приказ приватног садржаја игре (карте у руци, инвентар).

Компонента „Комуникација“ обезбеђује размену информација између уређаја и сервиса који учествују у истој игри. Ова компонента омогућава два начина повезивања уређаја у оквиру локалне кућне мреже или коришћењем интернет сервиса. Први начин омогућава само уређајима у оквиру исте локалне кућне мреже да учествују у игри, али обезбеђује мала времена одзива система, па се на овај начин могу креирати игре у реалном времену. Други начин омогућава корисницима смештеним на различитим локацијама да учествују у истој игри, али то доводи до већих кашњења у игри и овај начин комуникације се може искористити за потезне игре. Коришћењем ове компоненте могуће је обезбедити интеракцију групе људи у оквиру једне просторије са групом људи на другим локацијама.

Интеграција сервиса дигиталне телевизије је обезбеђена употребом компоненте „ТВ сервис“. Компонента „Друштвени медији“ представља интерфејс ка платформама познатих друштвених медија као што су: *Facebook*, *Twitter* и TMDb. Коришћењем компонената „ТВ сервис“ и „Друштвени медији“ обезбеђује се приступ ТВ и интернет сервисима што омогућава додавање паралелних интерактивних активности пасивном гледању телевизије. Такође, корисничко искуство је могуће побољшати приступом подацима из дигиталног видео тока, социјалним мрежама и базама метаподатака. Детекција покрета корисника употребом мобилног уређаја који поседује акцелерометре и магнетометре је реализована у оквиру компоненте „Препознавање покрета“. Ова компонента омогућава да се мобилни уређај искористи за управљање акцијама на централном екрану и за пријем додатних чулних повратних информације. На примеру игара

ова компонента омогућава да се мобилни уређаји искористе као контролери у игри.

Коришћењем SHARP софтверске платформе развијено је пет различитих игара: две потезне игре и три микро игре у реалном времену које се играју у току трајања ТВ програма. Потезне игре (*Dice of Blood* игра на табли и *Crazy Eights* игра картама) су развијене тако да могу да се играју и у оквиру локалне кућне мреже и коришћењем интернет сервиса. Микро игре (*Egg-and-Spoon*, *Flyswatter* и *Tomato rating*) су развијене тако да се играју у оквиру локалне кућне мреже. Развијене игре приказују флексибилност SHARP софтверске платформе, различите начине дистрибуције компонената платформе у оквиру интегрисаног окружења и демонстрирају употребу различитих комбинација идентификованих сценарија интеракције. SHARP софтверска платформа и апликације су намењене Андроид мултимедијалним уређајима, чије тржиште је у константном расту.

На примеру развијених игара донети су закључци о томе како корисници прихватају идентификоване сценарије интеракције испитивањем искуства 59 волонтера приликом употребе развијених ТВ центричних игара. Њихова искуства су прикупљена у виду упитника и отворених интервјуа са додатком опсервација истраживача. Што се тиче поделе приказа на више екрана како би се простор на великим јавним екранима искористио ефикасно пожељно је да димензије дељеног простора имају сличну пропорцију оној која се користи на ТВ екранима (на пример 16:9). Мобилни уређаји имају већи степен варијације у пропорцијама екрана, као и ограничен простор на екрану што их више кандидује за приказ дела дељеног простора или за приказ минималне количине додатних информација. На примеру игара подела пажње између два екрана током игара у реалном времену спречавала је кориснике да се удубе у свет игре, а такође им је ограничила и могућност праћења ТВ програма. Код потезних игара које су по природи споријег темпа употреба више екрана на којима су одвојене приватне и јавне информације је побољшала игру.

Као пример активности које се могу обављати у паралели са пасивним гледањем телевизије издвојене су микро игре. Природа микро игара и интуитивна употреба сензора за детекцију покрета комбинована са информацијама из дигиталног видео тока може да обогати искуство гледања телевизије. Корисници

су показали веће задовољство приликом играња једноставнијих игара. Такође, у току играња једноставнијих игара са минималном графиком корисници су боље пратили ТВ програм. Што се тиче управљања акцијама на великом екрану корисници су изразили веће задовољство у случају када су интерреаговали директно са објектом у свету игре са додатком препознавања покрета и пријема повратног одговора, него у случају када су употребљавани додатни дугмићи на екрану који су се користили за контролу игре. Закључак везан за интеракцију групе људи у интегрисаном окружењу је да традиционалне игре на табли и игре картама имају предности које се не могу једноставно заменити играма у дигиталној форми. ТВ центричне верзије потезних игара успевају да очувају контакт лицем у лице и додају опипљиве елементе у игру као што је држање мобилног уређаја који показује карте у руци. За компликованије игре на табли са тимским елементом игре члановима супарничког тима може бити од користи да се налазе у различитим собама (играјући верзију игре коришћењем интернет сервера), што им омогућава да између себе дискутују о стратегији игре.

Верификација SHARP софтверске платформе извршена је поређењем величине, сложености, густине сложености и времена одзива *Dice of Blood* и *Flyswatter* игре које су развијене коришћењем и без коришћења предложене платформе. Прво су поређене величине и сложености целокупног програмског кода апликација развијених са и без употребе платформе, а након тога, посебно за игру *Dice of Blood* и посебно за игру *Flyswatter*, извршено је поређење величине, сложеност и густине сложености програмског кода, категоризованог по свакој од коришћених компонената софтверске платформе.

Ако посматрамо целокупан програмски код развијених игара, игра *Dice of Blood* која је развијена употребом софтверске платформе је имала 47% линија програмског кода игре *Dice of Blood* која је развијена без употребе софтверске платформе, а комплексност јој је представљала 39 % комплексности игре која је развијена без употребе платформе. Игра *Flyswatter* је која је развијена употребом софтверске платформе је имала 53% линија програмског кода игре *Dice of Blood* која је развијена без употребе софтверске платформе, а комплексност јој је представљала 53 % комплексности игре која је развијена без употребе платформе. Посматрањем програмског кода категоризованог по компонентама софтверске

платформе уочено је да је код већине компонената, осим у случају компоненте која је задужена за логику игре, одређене функционалности скоро у потпуности пребачене у код софтверске платформе, остављајући програмеру да у коду који је директно везан за одређену игру дода само делове који се односе на иницијализацију. Добар пример су функционалности које се односе на приступ друштвеним медијима и препознавање покрета. Посматрањем густине комплексности програмског кода уочено је да је густина комплексности сваке од компонената платформе већа од густине комплексности додатног кода и за игру *Dice of Blood* и за *Flyswatter* игру. Програмски код који није део софтверске платформе има мање логичких израза тока контроле по физичкој линији кода у односу на програмски код компонената платформе. Вероватноћа да ће се грешка догодити у самој платформи је већа од вероватноће да се грешка догоди у додатом програмском коду игре што повољно утиче на време развоја и одржавање система. За грешке које се појаве у оквиру софтверске платформе биће довољно да се само једном исправе.

Резултати поређења су показали да је софтверска платформа утицала да се време потребно за развој и тестирање апликација смањи. Комплексна логика која је потребна за развој апликација као што су игре је премештена у софтверску платформу, а исте компоненте софтверске платформе се могу употребљавати за развој различитих врста апликација. У случају великих и комплексних апликација потреба за сложеном логиком која је смештена у различитим компонентама платформе се повећава, па је у овом случају и корист приликом употребе платформе већа него у случају када се развијају апликације мање величине и комплексности. Велику корист програмери су имали и од смерница које је софтверска платформа увела у структурирању логике игре. Коришћење софтверске платформе је резултовало већим степеном поделе задужења класа развијених у апликацијама које су користиле софтверску платформу у односу на апликације које је нису користиле. Коришћење софтверске платформе је спречило програмере да у оквиру исте класе спајају програмски код који припада различитим компонентама система.

Поређење времена одзива апликација је извршено за случај геста „ударца“ код игре *Flyswatter* развијене без употребе и употребом SHARP софтверске

платформе. Мерење је вршено од тренутка почетка корисничке акције, до тренутка када се акција прикаже на ТВ екрану за 100 узастопних гестова „ударца“ код обе апликације. Времена одзива прикупљена из апликација развијених без коришћења и коришћењем софтверске платформе су показала да је апликација развијена коришћењем платформе унела у просеку прихватљивих 2,95ms додатног кашњења. Такође, времена одзива у обе верзије апликација су била у дозвољеним границама перформанси свих утврђених прагова одговора на команде и одзиве у играма.

Будуће активности које би могле бити продужетак овог рада се могу поделити на више праваца. Један правац би био примена развијене SHARP софтверске платформе у развоју других типова апликација које би корисницима понудиле идентификоване сценарије интеракције у интегрисаном окружењу. Неки од примера би могли бити соба за састанке, колаборативно планирање, дељење календара, видеа или слика. Следећи правац би се односио на надоградњу саме SHARP софтверске платформе. Платформа би могла да се надогради тако да подржи размену говорних и текстуалних порука између корисника. Такође, платформа би могла да омогући да акција у емитованом ТВ програму утиче на покретање одређене акције у оквиру апликације развијене коришћењем платформе. Платформа би могла да омогући клијентима да се коришћењем интернет претраживача прикључе на интернет сервис. Ово би обезбедило развој апликација које би биле независне од платформе, али би потенцијални компромис био губитак могућности препознавања покрета.

# ЛИТЕРАТУРА


1. *Android OS*, Available at: www.android.com, accessed December 2014
2. Agrawal S, Constandache I, Gaonkar S, Roy Choudhury R, Cave K, DeRuyter F: *Using mobile phones to write in air*, Proceedings of the 9th international conference on Mobile systems, applications, and services, ACM, pp. 15-28, 2011
3. Baek J, Yun B: *A sequence-action recognition applying state machine for user interface*, IEEE Transactions on Consumer Electronics, vol. 54, no. 2, pp. 719-726, 2008
4. Bakker S, Vorstenbosch D, van den Hoven E, Hollemans G, Bergman T: *Weathergods: tangible interaction in a digital tabletop game*, Proceedings of the 1st international conference on Tangible and embedded interaction, pp 151–152, New York, NY, USA, 2007. ACM Press.
5. Bao L, Intille S. S: *Activity recognition from user-annotated acceleration data*, Pervasive computing, Springer Berlin Heidelberg, vol. 3001, pp. 1-17, 2004
6. Baughman N, Liberatore M, Levine B: Cheat-*proof playout for centralized and peer-to-peer gaming*, IEEE Transactions on Networking vol. 15, no. 1, pp. 1–13, 2007
7. Benford S, Magerkurth C, Ljungstrand P: *Bridging the physical and digital in pervasive gaming*. Communications of the ACM, vol. 48, no. 3, pp. 54–57, 2005.
8. Berners-Lee T, Fielding R, Frystyk H : *Hypertext transfer protocol--HTTP/1.0*, MIT, 1996
9. Bjelica M. Z: *Methods of implementation of context-aware platforms and context-aware user interfaces for applications in consumer electronics*, Doctoral dissertation, Faculty of Technical Sciences, University of Novi Sad, Serbia, pp.1-361, 2013
10. Blizzard entertainment, *World of Warcraft*, Available at: http://us.battle.net/wow/en/, Accessed December 2014

11. Bhandari S, Lim Y. K: *Exploring gestural mode of interaction with mobile phones*, Proceedings of the CHI'08 Extended Abstracts on Human Factors in Computing Systems, pp. 2979-2984, 2008
12. Boehm B, Abts C, Chulani S: *Software development cost estimation approaches—A survey*, Annals of Software Engineering, vol. 10 no. 1-4, pp. 177-205, 2000
13. **А** Boertjes E: *ConnecTV: Share the Experience*, Proceedings of the EuroITV 2007, pp.139-140, 2007
14. **Б** Boertjes E, Klok J, Schultz S: *ConnecTV: Results of the field trial*, Proceedings of EuroITV 2008, available at: https://soc.kuleuven.be/com/mediac/socialitv2/papers/ConnecTV_Results_of_the_Field_Trial.pdf, Accessed December 2014
15. Cesar P, Geerts D: *Past, present, and future of social TV: a categorization*, Proceedings of the Consumer Communications and Networking Conference (CCNC), IEEE, pp. 347-351, 2011
16. Chuang Y. L, Liao C. W, Chen W. S, Chang W. T, Cheng S. H, Zeng Y. C, Chan K. H (2013) Use second screen to enhance TV viewing experiences. Cross-Cultural Design. Methods, Practice, and Case Studies. 8023:366-374
17. **А** Claypool M, Claypool K: *Latency and player actions in online games*, Communications of the ACM vol. 49, no. 11, pp. 40-45, 2006
18. **Б** Claypool M, Claypool K, Damaa F: *The effects of frame rate and resolution on users playing first person shooter games*, Proceedings of the ACM/SPIE Multimedia Computing and Networking (MMCN '06) Conference, San Jose, CA, USA, pp. 1-11, 2006
19. Coppens T, Trappeniers L, Godon M: *AmigoTV: towards a social TV experience*,Proceedings of the Second European Conference on Interactive Television: Enhancing the experience, Brighton, pp. 159-162, 2004
20. Cha M, Haddadi H, Benevenuto F, Gummadi P. K*: Measuring User Influence in Twitter: The Million Follower Fallacy*, Proceedings of the 4th International AAAI Conference on Weblogs and Social Media ICWSM, no. 38, 2010
21. Cruickshank L, Tsekleves E, Whitham R, Hill A, Kondo K: *Making interactive TV easier to use: Interface design for a second screen approach*, The Design Journal vol. 10, no. 3, pp. 41-53, 2007

22. Crockford D, *The application/json media type for javascript object notation (json)*, 2006, Available at: https://tools.ietf.org/html/rfc4627, accessed December 2014
23. Dachselt R, Buchholz R: *Natural throw and tilt interaction between mobile phones and distant displays*, Proceedings of the 27th international conference extended abstracts on Human factors in computing systems, ACM, New York, NY, USA, pp. 3253–3258, 2009
24. Diele O: *State Of Online Gaming Report*, Spil Games, 2013, Available at: http://auth-83051f68-ec6c-44e0-afe5-bd8902acff57.cdn.spilcloud.com/v1/archives/1384952861.25_State_of_Gaming_2013_US_ FINAL.pdf, Accessed December 2014
25. Döring T, Shirazi A. S, Schmidt A: *Exploring gesture-based interaction techniques in multi-display environments with mobile phones and a multi-touch table*, Proceedings of the International Conference on Advanced Visual Interfaces (AVI'10), pp. 419-419, 2010
26. Duggan M, Brenner J: *The demographics of social media users – 2012*, Pew Research Center, Washington, DC. Available at: http://www.lateledipenelope.it/public/513cbff2daf54.pdf , Accessed December 2014
27. EVE Online, Available at: http://www.eveonline.com/, Accessed December 2014
28. Ellison N. B, Steinfield C, Lampe C: *The benefits of Facebook "friends:" Social capital and college students' use of online social network sites*, Journal of Computer Mediated Communication, vol. 12, no. 4, pp. 1143-1168, 2007
29. Fallahkhair S, Pemberton L, Griffiths R: *Development of a cross-platform ubiquitous language learning service via mobile phone and interactive television*, Journal of Computer Assisted Learning, vol. 23, no. 4, pp. 312-325
30. Frapolli F, Hirsbrunner B, Lalanne D: *Dynamic rules: Towards interactive games intelligence*, Proceedings of the 12th international conference on intelligent user interfaces, ACM Press, New York, NY, USA, pp. 29–32, 2007
31. Fette I, Melnikov A: *RFC 6455: The WebSocket protocol*, Internet Engineering Task Force, pp. 4-12, 2011
32. FFIV Final Fantasy 14, Available at: http://www.finalfantasyxiv.com/, Accessed December 2014
33. Facebook, Available at: www.facebook.com, Accessed December 2014

34. Gill GK, Kemerer CF: *Cyclomatic complexity density and software maintenance productivity*, IEEE Transactions on Software Engineering, vol. 17, no. 12, pp. 1284-1288, 1991
35. Huang C. Y, Hsu C. H, Chang Y. C, Chen K. T*: GamingAnywhere: An Open Cloud Gaming System*, Proceedings of the ACM Multimedia Systems 2013, pp. 1-12, 2013
36. JTMDB API, *Java wrapper for themoviedb.org*, Available at: http://jtmdb.sourceforge.net/, Accessed December 2014
37. JBOSS Aplication Server 7, Available at: http://jbossas.jboss.org/, Accessed May 2013
38. Jeličić A, Punt M, Bjelica M. Z, Vujanović V: New Possibilities of Human-Computer Interaction in Integrated Video Games for Mobile Phones, TV and Internet, Proceeding of the IEEE Third International Conference on Consumer Electronics (ICCE), Berlin, pp. 279-281, 2013
39. Joselli M, Clua E: *Grmobile: A framework for touch and accelerometer gesture recognition for mobile games*, Proceedings of the 8th Brazilian Symposium on Games and Digital Entertainment (SBGAMES), pp. 141-150, 2009
40. Jovanović S, Punt M, Bjelica M, Zdravković V, Kukolj M**:** *An Approach of Integrating Communication Services in Applications for Android-Based Digital TV Receivers*, TELFOR JOURNAL, vol. 5, no. 1, pp. 60-64, 2013
41. Kemp R, Palmer N, Kielmann T, Bal H: *Cuckoo: a computation offloading framework for smartphones*, Mobile Computing, Applications, and Services vol. 76, pp. 59-79, 2012
42. Kulkarni S, Douglas S, Churchill D: *Badumna: A decentralised network engine for virtual environments*, Computer Networks, vol. 54, no. 12, pp. 1953–1967, 2010
43. Kwapisz J. R, Weiss G. M, Moore S. A: *Activity recognition using cell phone accelerometers*, ACM SigKDD Explorations Newsletter, vol. 12, no. 2, pp. 74-82, 2011
44. Lane N. D, Miluzzo E, Lu H, Peebles D, Choudhury T, Campbell A. T: *A survey of mobile phone sensing*, Communications Magazine, IEEE, vol. 48, no. 9, pp. 140-150, 2010

45. Lanza M, Marinescu R: *Object-oriented metrics in practice: using software metrics to characterize, evaluate, and improve the design of object-oriented systems.* Springer Science & Business Media, 2007
46. LocMetrics, Available at: http://www.locmetrics.com, Accessed September 2013
47. Malfatti S. M, Ferreira dos Santos F, Rodrigues dos Santos S: *Using mobile phones to control desktop multiplayer games*, Proceedings of the Brazilian Symposium on Games and Digital Entertainment (SBGAMES), IEEE, pp. 230-238, 2010
48. McAdam C, Brewster S: *Using mobile phones to interact with tabletop computers*, Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces, pp. 232-241, 2011
49. McCabe T. J: *A complexity measure*, IEEE Transactions on Software Engineering, vol. 2, no. 4, pp. 308–320, 1976
50. Magerkurth C, Memisoglu M, Engelke T, Streitz N: *Towards the next generation of tabletop gaming experiences*, Proceedings of the Graphics Interface Conference (GI'04), pp. 73-80, 2004
51. Metcalf C, Harboe G, Tullio J, Massey N, Romano G, Huang E. M, Bentley F: *Examining presence and lightweight messaging in a social television experience*, ACM Transactions on Multimedia Computing, Communications, and Applications vol. 4, article no. 27, 2008
52. Nathan M, Harrison C, Yarosh S, Terveen L, Stead L, Amento B: CollaboraTV*: making television viewing social again*, Proceedings of the 1st international conference on Designing interactive user experiences for TV and video, pp. 85-94, 2008
53. Netty Project, Available at: http://netty.io/, Accessed May 2013
54. Nguyen V, Deeds-Rubin S, Tan T, Boehm B: *A SLOC counting standard*, Proceedings of the 22nd International Annual Forum on COCOMO and Systems/Software Cost Modeling, Los Angeles, USA, 2007
55. Nielsen Company, *Cross Platform is the New Norm*, 2011, Available at: http://www.nielsen.com/us/en/reports/2011/cross-platform-is-the-new-norm.html. Accessed 9 February 2014, Accessed February 2013
56. Punt M, Bjelica M. Z, Zdravkovic V, Teslic N: *An integrated environment and development framework for social gaming using mobile devices, digital TV and*

*Internet*, Multimedia Tools and Applications, Available at: http://link.springer.com/article/10.1007/s11042-014-2045-8, Accessed January 2015

57. Römer K, Domnitcheva S: *Smart playing cards: A ubiquitous computing game*, Personal and Ubiquitous Computing vol. 6, no. 5-6, pp. 371-377, 2002
58. Scheible J, Ojala T, Coulton P: *MobiToss: A novel gesture based interface for creating and sharing mobile multimedia art on large public displays*. Proceedings of the 16th ACM international conference on Multimedia, pp. 957-960, 2008
59. Shneiderman B: *Response time and display rate in human performance with computers*, ACM Computing Surveys (CSUR), vol. 16, no.3, pp. 265-285, 1984
60. Strategy Analytics, Available at: http://www.prnewswire.com/news-releases/strategy-analytics-android-shipped-1-billion-smartphones-worldwide-in-2014-300027707.html, Accessed January 2015
61. Steele J, To N: *The Android developer's cookbook: building applications with the Android SDK*, Pearson Education, 2010
62. The Movie Database ,TMDb, Available at: www.themoviedb.org, Accessed December 2014
63. The Statistics Portal Statista, *Distribution of time spent on apps on iOS and Android devices in April 2014, by category*, Available at: http://www.statista.com/statistics/248343/distribution-of-time-spent-ios-and-android-apps-by-category/, Accessed January 2015
64. Tullis T, Albert W: *Measuring the user experience: collecting, analyzing, and presenting usability metrics*, Morgan Kaufmann, 2010
65. TV Anytime, Available at: http://www.tv-anytime.org/, Accessed December 2014
66. Tršić L, Veljković N, Punt M, Bjelica M Z, Predojević M: *Realizacija računarskog servisa u oblaku za društvene igre za mobilne i TV uređaje*, Telfor 2013, pp. 999-1002, Beograd, 2013
67. Twitter Inc., Available at: www.twitter.com, Accessed December 2014
68. Tse E, Greenberg S, Shen C, Forlines C: *Multimodal multiplayer tabletop gaming*, Proceedings of the Third International Workshop on Pervasive Gaming Applications - PerGames 2006, pp. 141–150, Dublin, Ireland, 2006

69. Vajk T, Coulton P, Bamford W, Edwards R: *Using a mobile phone as a 'Wii-like' controller for playing games on a large public display*, International Journal of Computer Games Technology, vol. 2008, pp. 1-6, 2008
70. Vidakovic M, Maruna T, Teslic N, Mihic V: *A java API interface for the integration of DTV services in embedded multimedia devices*, IEEE Transactions on Consumer Electronics, vol. 58, no. 3, pp. 1063–1069, 2012
71. Venture Beat, *2014 Global Games Market Report*, Available at: http://insight.venturebeat.com/report/2014-global-games-market-report, Accessed January 2015
72. Wang S. C, Chung T. C, Yan K. Q: *A new territory of multi-user variable remote control for interactive TV*, Multimedia Tools and Applications, vol. 51, no. 3, pp. 1013-1034, 2011
73. Wingrave C. A, Williamson B, Varcholik P D, Rose J, Miller A, Charbonneau E, LaViola J. J: *The wiimote and beyond: Spatially convenient devices for 3d user interfaces*, Computer Graphics and Applications, IEEE, vol. 30, no. 2, 71-85, 2010
74. Winterwell Associates Ltd, "*jtwitter*", Available at: http://www.winterwell.com/software/jtwitter.php, Accessed December 2014
75. Yahoo! Inc and The Nielsen Company, *Mobile Shopping Framework: The Role of Mobile Devices in the Shopping Process*, Available at: http://advertising.yahoo.com/article/the-role-of-mobile-devices-in-shopping-process.html, Accessed December 2012
76. Yahyavi A, Kemme B: *Peer-to-peer architectures for massively multiplayer online games: A survey*, ACM Computing Surveys (CSUR), vol. 46, no.1 , article no. 9, 2013
77. Yamamoto Y, *Twitter4j*, Available: http://twitter4j.org/en/, Accessed July 2013
78. Zamith M, Montenegro A, Clua E, Joselli M, Feijo B, Leal R, Valente L: *An distributed architecture for mobile digital games based on cloud computing*, Proceedings of the Brazilian Symposium on Computer Games and Digital Entertainment (SBGames '11), pp. 79–88, 2011
79. Zhang Z: *Microsoft kinect sensor and its effect*, MultiMedia, IEEE, vol. 19, no.2, pp. 4-10, 2012

# БИОГРАФИЈА АУТОРА

Марија Пунт (девојачко Обрадовић) је рођена 1978. у Београду, где је завршила основну школу као ђак генерације. Математичку гимназију завршила је 1997. године. Током Основне и Средње школе била је учесник и освајала награде на Републичким и Савезним такмичењима из математике, као и редован учесник Летњих и Зимских школа младих математичара у организацији Друштва математичара Србије.

На Електротехнички факултет у Београду уписала се 1997. године, смер Рачунарска техника и информатика. Дипломирала је на Електротехничком факултету, јула 2004, са просечном оценом 8,69 и оценом 10 на дипломском. Магистарску тезу под насловом "Евалуација једног система за детекцију грешака базираног на техници надгледања процесора" одбранила је у децембру 2009. године на Електротехничком факултету код ментора проф. др Мила Томашевића и проф. др Јована Ђорђевића.

Од августа 2002. до августа 2003., преко ИАСТЕ-а, радила је у истраживачко развојном центру *British Telekom Adastral Park*, Ипсвич, Енглеска на истраживању и развоју *Ad-hoc Peer-to-Peer* бежичних мрежа. Од октобра 2004. до марта 2006. ради као стручни сарадник на Вишој електротехничкој школи из предмета Програмски језик Јава и Објектно програмирање, а од фебруара 2005. и на Електротехничком факултету као постдипломац. Од марта 2006. на Електротехничком факултету ради као асистент на предметима: Основе рачунарске технике 1, Основе рачунарске технике 2, Архитектура рачунара, Организација рачунара, Архитектура и организација рачунара, Веб дизајн, Практикум из коришћења рачунара и Практикум из Основа рачунарске технике.

Кандидаткиња је учествовала на два пројекта финансираних од Министарства за науку Републике Србије. Коаутор је на четири збирке задатака и приручника за лабораторијске вежбе који се користе на Електротехничком факултету у Београду. Коаутор је на два рада у међународним часописима са *impact* фактором са SCI листе од којих је један у директној вези са дисертацијом, једног рада у домаћем часопису и девет радова на међународним и домаћим стручним конференцијама.

# ПРИЛОГ 1

## Изјава о ауторству

Потписана Марија Пунт

број индекса (по старом програму је)

**Изјављујем**

да је докторска дисертација под насловом

Интеракција човек-рачунар у интегрисаном окружењу дигиталних ТВ пријемника,

мобилних уређаја и Интернета

- резултат сопственог истраживачког рада,
- да предложена дисертација у целини ни у деловима није била предложена за добијање било које дипломе према студијским програмима других високошколских установа,
- да су резултати коректно наведени и
- да нисам кршио/ла ауторска права и користио интелектуалну својину других лица.

**Потпис докторанда**

У Београду, 13.07.2015

# ПРИЛОГ 2

## Изјава о истоветности штампане и електронске верзије докторског рада

Име и презиме аутора Марија Пунт

Број индекса (по старом програму је)

Студијски програм Рачунарска техника и информатика

Наслов рада Интеракција човек-рачунар у интегрисаном окружењу дигиталних ТВ пријемника, мобилних уређаја и Интернета

Ментор проф. Др Јован Ђорђевић

Потписана Марија Пунт

Изјављујем да је штампана верзија мог докторског рада истоветна електронској верзији коју сам предао/ла за објављивање на порталу **Дигиталног репозиторијума Универзитета у Београду.**

Дозвољавам да се објаве моји лични подаци везани за добијање академског звања доктора наука, као што су име и презиме, година и место рођења и датум одбране рада.

Ови лични подаци могу се објавити на мрежним страницама дигиталне библиотеке, у електронском каталогу и у публикацијама Универзитета у Београду.

**Потпис докторанда**

У Београду, 13.07.2015

# ПРИЛОГ 3

## Изјава о коришћењу

Овлашћујем Универзитетску библиотеку „Светозар Марковић“ да у Дигитални репозиторијум Универзитета у Београду унесе моју докторску дисертацију под насловом:

Интеракција човек-рачунар у интегрисаном окружењу дигиталних ТВ пријемника,

мобилних уређаја и Интернета

која је моје ауторско дело.

Дисертацију са свим прилозима предао/ла сам у електронском формату погодном за трајно архивирање.

Моју докторску дисертацију похрањену у Дигитални репозиторијум Универзитета у Београду могу да користе сви који поштују одредбе садржане у одабраном типу лиценце Креативне заједнице (Creative Commons) за коју сам се одлучио/ла.

1. Ауторство

2. Ауторство - некомерцијално

3. Ауторство – некомерцијално – без прераде

4. Ауторство – некомерцијално – делити под истим условима

5. Ауторство – без прераде

6. Ауторство – делити под истим условима

(Молимо да заокружите само једну од шест понуђених лиценци, кратак опис лиценци дат је на полеђини листа).

**Потпис докторанда**

У Београду, 13.07.2015

1. Ауторство - Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце, чак и у комерцијалне сврхе. Ово је најслободнија од свих лиценци.

2. Ауторство – некомерцијално. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела.

3. Ауторство - некомерцијално – без прераде. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела. У односу на све остале лиценце, овом лиценцом се ограничава највећи обим права коришћења дела.

4. Ауторство - некомерцијално – делити под истим условима. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца не дозвољава комерцијалну употребу дела и прерада.

5. Ауторство – без прераде. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца дозвољава комерцијалну употребу дела.

6. Ауторство - делити под истим условима. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца дозвољава комерцијалну употребу дела и прерада. Слична је софтверским лиценцама, односно лиценцама отвореног кода.