

**SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN**

Nino Lončar

VIŠEAGENTNI SUSTAV ZA SIMULACIJU DUNGEON CRAWLER IGRE

PROJEKT

VIŠEAGENTNI SUSTAVI

Varaždin, 2026.

SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
V A R A Ž D I N

Nino Lončar

Matični broj: 0016153176

Studij: Informacijsko i programsko inženjerstvo

VIŠEAGENTNI SUSTAV ZA SIMULACIJU DUNGEON CRAWLER IGRE

PROJEKT

Mentor:

doc. dr. sc. Bogdan Okreša Đurić

Varaždin, siječanj 2026.

Izjava o izvornosti

Izjavljujem da je ovaj projekt izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor potvrdio prihvatanjem odredbi u sustavu FOI Radovi

Sažetak

Ovaj rad opisuje simulaciju dungeon crawler igre implementirane s pomoću višeagentnog sustava. U radu je najprije predstavljena teorijska pozadina višeagentnih sustava i osnovni koncepti na kojima se sustav temelji. Zatim je objašnjena implementacija same igre, s posebnim naglaskom na strukturu agenata, njihove uloge, ponašanja i međusobnu interakciju tijekom simulacije. Opisano je funkcioniranje borbenog sustava, donošenje odluka agenata te upravljanje sposobnostima i stanjima likova. Na kraju je prikazan primjer izvršavanja simulacije koji ilustrira ponašanje agenata.

Ključne riječi: višeagentni sustav; agent; dungeon crawler; borba temeljena na potezima; simulacija

Sadržaj

1. Uvod	1
2. Metode i tehnike rada	2
3. Teorijska pozadina	3
3.1. Višeagentni sustavi	3
3.1.1. Komunikacija agenata	3
3.1.2. Glasanje u višeagentnim sustavima	4
3.2. Dungeon crawler igra	4
4. Opis implementacija	5
4.1. Agenti simulacije	5
4.1.1. Sposobnosti agenata	6
4.1.2. Stilovi borbe	7
4.1.3. Stilovi glasanja	8
4.2. Inicijalizacija simulacije	8
4.3. Petlja igre	9
5. Prikaz rada aplikacije	11
5.1. Pregled postavki simulacije	11
5.1.1. Heroji	11
5.1.2. Neprijatelji	12
5.1.3. Tamnica	13
5.2. Izvođenje simulacije	14
6. Kritički osvrt	16
7. Zaključak	17
Popis literature	18
Popis slika	19
Popis tablica	20
Popis isječaka koda	21

1. Uvod

Dungeon crawler igre predstavljaju jedan od žanrova videoigara u kojemu se naglasak stavlja na istraživanje prostora, borbu s neprijateljima i razvoj likova. Borba bazirana na potezima, koju ovakve igre često koriste, daje dobro definiranih pravila, stanja i interakcije između većeg broja autonomnih likova te je ovakav tip igara pogodan za modeliranje i analizu s pomoću višeagentnih sustava.

Cilj ovog rada je osmisliti i implementirati simulaciju dungeon crawler igre koristeći principe višeagentnih sustava. Na početku je predstavljena teorijska pozadina višeagentnih sustava i dungeon crawler igara. U nastavku je opisana arhitektura i funkcioniranje sustava, zajedno s implementacijom ključnih komponenti. Na kraju se prikazuju rezultati izvođenja simulacije, daje se osvrt na implementirano te se donosi zaključak.

2. Metode i tehnike rada

U ovom poglavlju opisane su metode i tehnike korištene pri razradi teme, način provedbe istraživačkih aktivnosti te programski alati i tehnologije primijenjeni tijekom izrade rada.

Za istraživanje teorije iza ovog rada korišten je Google Scholar te su svi korišteni izvori pronađeni preko njega.

Za implementaciju je korišten isključivo programski jezik Python. Nije korištena nikakva vrsta baze podataka jer se svi podaci učitavaju iz CSV (eng. *Comma-Separated Values*) datoteki. Korištene Python knjižnice su: random, csv i tkinter.

3. Teorijska pozadina

U ovome poglavlju dana je teorijska pozadina iza implementiranog sustava. Kratko su objašnjeni osnovni principi agenata i višeagentnih sustava, komunikacija između agenata te glasanje u višeagentnim sustavima. Također je ukratko objašnjen dungeon crawler žanr.

3.1. Višeagentni sustavi

Prema Gokulanu i Srinivasan [1], najprihvaćenija definicija agenta opisuje kao "fleksibilni i autonomni entitet s mogućnošću percipiranja svoje okoline preko senzora". Agenti također mogu interagirati sa svojom okolinom, često preko aktuatora. Agente opisuju s nekoliko karakteristika; situiranost, autonomnost, sposobnost zaključivanja, responzivnost, proaktivnost i društveno ponašanje.

Situiranost označuje da se agent nalazi u nekom okruženju koje može percipirati i s kojim može interagirati. Autonomnost označuje da agent može sam donositi odluke bez poticaja ili uplitanja drugih agenata, sustava ili ljudi. Sposobnost zaključivanja označuje sposobnost da radi s općim ciljevima i da povezivanjem dostupnih informacija izvuče zaključke. Responzivnost označuje da agent može odgovarati na stanje svojeg okruženja. Proaktivnost označuje da agent može utjecati na svoju okolinu i dinamički joj se prilagođavati. Društveno ponašanje označava sposobnost agenta da surađuje s drugim agentima kako bi postigao svoje ciljeve i sposobnost dijeljenja informacija s njima. [1]

Višeagentni sustav povezuje više agenata s ciljem postizanja zajedničkog cilja. U ovakvim sustavima, postupci agenata utječu i na njegovo okruženje, ali i na druge agente, zbog čega bi agenti trebali predviđati ponašanja drugih ageneta i prilagoditi svoje ponašanje. S obzirom da pojedini agent, zbog ograničenosti svojih senzora, vjerojatno ne može percipirati cjelokupno okruženje u kojemu se sustav nalazi, njegove akcije mogu *naštetiti* drugim agentima, zbog čega agenti moraju moći rješavati konflikte kako bi radili prema zajedničkom cilju. [1]

3.1.1. Komunikacija agenata

Kako bi postigli zajedničke ili pojedinačne ciljeve, agenti moraju komunicirati i razmjenjivati informacije. Mogućnost i lakoća komunikacije ovisi o okruženju u kojem se agenti nalaze. Okruženja mogu, primjerice, varirati u tome jesu li spoznatljiva (mjera u kojoj agenti mogu spoznati svoje okruženje), predvidljiva (mjera u kojoj agenti mogu predvidjeti stanje okruženja), može li se kontrolirati (u kojoj mjeri agenti mogu mijenjati okruženje)... [2]

Huns i Stephens [2] razlikuju dvije osnovne vrste poruka koje omogućuju komunikaciju između agenata: upit (eng. *Query*) i tvrdnja (eng. *Assertion*). Upiti agentima omogućuju zahtjevanje informacija, dok tvrdnje omogućuju slanje informacija. S ove dvije poruke može se realizirati jednostavan protokol u kojemu je jedan agent aktivni sudionik i šalje upite drugom agentu koji je pasivni sudionik koji čeka upit i šalje tvrdnju kao odgovor na upit.

Pri interpretaciji poruke potrebno je uzeti u obzir njezin kontekst, koji može značajno utjecati na značenje poruke, identitet pošiljatelja te kardinalnost, odnosno je li poruka poslana privatno ili svim agentima u sustavu... [2]

3.1.2. Glasanje u višeagentnim sustavima

Glasanje u višeagentnim sustavima predstavlja način na koji agenti mogu zajednički donijeti odluku pri čemu vode brigu o tome da postignu zajedničke, ali i vlastite ciljeve. Pitt i drugi [3] za složenije vrste glasanja u višeagentnim sustavima razlikuju nekoliko različitih uloga. Glasači su agenti koji imaju pravo glasa. Ulogu predlagatelja imaju agenti koji mogu davati prijedloge. Ulogu podupiratelja imaju agenti koji su ovlašteni poduprijeti dane prijedloge. Predsjedavajući su agenti koji su osposobljeni za vođenje postupka. Ulogu nadzornika imaju agenti koji trebaju biti obaviješteni o glasovima i rezultatima glasanja.

3.2. Dungeon crawler igra

Dungeon crawler je žanr videoigara u kojemu igrač ili skup igrača istražuje tamnice (eng. *Dungeon*). Igrač ili igrači se kreću nizom povezanih prostorija koje mogu sadržavati različite zagonetke, borbe s neprijateljima ili opremu za igrače, kao što su oružje ili oklop. Tamnice su vrlo često proceduralno generirane. Ovaj žanr igre često uključuje RPG (eng. *Role-Playing Game*) elemente, što znači da igrači mogu kreirati i razvijati svoje likove. Barony i Darkest dungeon predstavljaju primjere videoigara iz ovog žanra.

4. Opis implementacija

U simulaciji postoje dvije grupe agenata s konfliktnim ciljevima; heroji i neprijatelji. Cilj heroja je izaći iz tamnice ili ju u potpunosti očistiti od neprijatelja. Cilj neprijatelja je zaustaviti heroje u njihovom naumu. Heroji prolaze kroz tamnicu, bore se s neprijateljima na koje naiđu u borbi koja se temelji na potezima (eng. *Turn-Based Combat*) te traže izlaz iz tamnice. U nastavku su opisane karakteristike agenata, inicijalno postavljanje simulacije i glavna petlja igre.

Cjelokupni izvorni kod je dostupan na <https://github.com/NinoLoncar/dungeon-crawler-simulation>.

4.1. Agenti simulacije

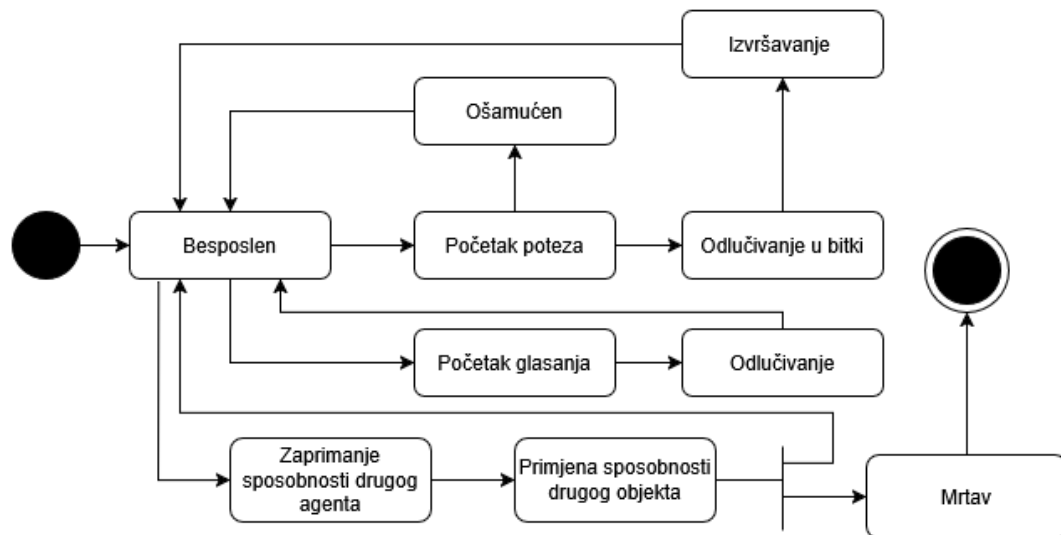
Svaki agent predstavlja jednog lika koji nije igrač ili NPC-a (eng. *Non-Player Character*). NPC-i su grupirani u dvije skupine; u heroje i neprijatelje. Heroji i neprijatelji se uglavnom uopće ne razlikuju. Jedina je razlika što će heroji svojim napadima ciljati neprijatelje, a neprijatelji će napadati heroje. Heroji također imaju sposobnost glasanja kada biraju sobu u koju će ići. Iz toga razloga, klase *Hero* i *Enemy* nasljeđuju klasu *Npc* koja implementira većinu ponašanja agenata; odabir sposobnosti, odabir mete, prihvata napada... Konstruktor klase *Npc* prikazan je u isječku koda 1. Svaki NPC ima svoj naziv, maksimalno zdravlje ili HP (eng. *Hit point*), brzinu, stil borbe i ikonu. Brzina određuje redoslijed u kojem će agenti izvršavati svoje poteze tijekom bitke. Stil borbe određuje koju će metu agent prioritizirati tijekom borbe.

```
1 class Npc:
2     def __init__(
3         self,
4         name,
5         max_hp,
6         speed,
7         fighting_style,
8         icon=None,
9     ):
10         self.name = name
11         self.icon = icon
12         self.max_hp = max_hp
13         self.hp = max_hp
14         self.speed = speed
15         self.fighting_style = fighting_style
16         self.is_alive = True
17         self.abilities = []
18         self.status_effects = []
19         self.is_stunned = False
```

Isječak koda 1: Klasa *Npc*

Slika 1 prikazuje dijagram stanja heroj agenta. Agent počinje u stanju besposlen (eng. *Idle*). Kada počne njegov potez primjenjuje na sebe sve statusne efekte koje ima, koji su detaljnije opisani u sljedećem poglavlju. Ako ima statusni efekt ošamućen, preskače svoj potez i vraća se u stanje besposlen. Inače odlučuje koji će sposobnost primijeniti i na komu, nakon

čega izvršava tu sposobnost tako da ju preda objektu klase *Game* koji prosljeđuje sposobnost njezinim metama. Ako se provodi glasanje o tome koju sobu sljedeće istražiti, heroj o tome odlučuje i daje svoj glas. Ako objekt klase *Game* prosljedi heroju sposobnost nekog drugog agenta, on ju sam primjenjuje na sebi. Postoji mogućnost da za sposobnost smanji njegov HP na 0 zbog čega agent umire i završava svoj životni ciklus. Inače se vraća u stanje besposlen. Dijagram stanja za neprijatelje je jednak samo što nema stanja za glasanje.



Slika 1: Dijagram stanja heroj agenta

4.1.1. Sposobnosti agenata

Svaki agent ima svoje sposobnosti (eng. *Ability*). Postoje tri vrste sposobnosti; sposobnosti za napadanje, sposobnosti za ozdravljivanje te statusne sposobnosti. Sposobnosti za napadanje čine štetu svojoj meti. Sposobnosti za ozdravljivanje povećavaju HP mete ako je trenutni HP manji od maksimalnog. Statusne sposobnosti svojoj meti daju različite privremene statuse koji traju određeni broj poteza. Ti statusi svojoj meti mogu davati različite efekte; efekt može svaki potez meti napraviti štetu, može joj privremene povećati neki atribut poput brzine ili ju može ošamutiti što znači da se njezin potez preskače. Svaka sposobnost ima hlađenje (eng. *Cooldown*), odnosno broj poteza koji mora proći između dva korištenja sposobnosti. Sposobnost može imati samo jednu metu ili područje djelovanja ili AoE (eng. *Area of Effect*) što znači da zahvaća cijelu grupu koja se cilja. Isječak koda 2 prikazuje osnovnu klasu *Ability* koja implementira osnovno ponašanje sposobnosti.

```

1  class Ability:
2      def __init__(self, name, cooldown_duration, aoe, priority, source, icon):
3          self.name = name
4          self.cooldown_duration = cooldown_duration
5          self.icon = icon
6          self.aoe = aoe
7          self.source = source
8          self.on_cooldown = False
9          self.cooldown_left = 0
10         self.priority = priority

11     def start_cooldown(self):
12         if self.cooldown_duration > 0:
13             self.on_cooldown = True
14             self.cooldown_left = self.cooldown_duration

15     def tick_cooldown(self):
16         if not self.on_cooldown:
17             return
18         self.cooldown_left -= 1
19         if self.cooldown_left <= 0:
20             self.cooldown_left = 0
21             self.on_cooldown = False

22     def is_ready(self):
23         return not self.on_cooldown

24     def reset_cooldown(self):
25         self.cooldown_left = 0
26         self.on_cooldown = False

```

Isječak koda 2: Klasa *Ability*

4.1.2. Stilovi borbe

Svaki heroj i neprijatelj ima svoj stil borbe. Stil borbe određuje koje će mete agent prioritzirati kod svojih napada. Postoji 5 stilova borbe koji su opisani u tablici 1.

Tablica 1: Stilovi borbe

Naziv	Opis
Kaotičan (eng. <i>Chaotic</i>)	Odabire nasumičnu metu
Pomahnitali (eng. <i>Berserker</i>)	Odabire nasumičnu metu i ima 15% da odabere saveznika
Oportunist (eng. <i>Opportunist</i>)	Odabire metu s najmanje HP-a
Odvažan (eng. <i>Bold</i>)	Odabire metu s najviše HP-a
Progonitelj (eng. <i>Chaser</i>)	Odabire najbržu metu

4.1.3. Stilovi glasanja

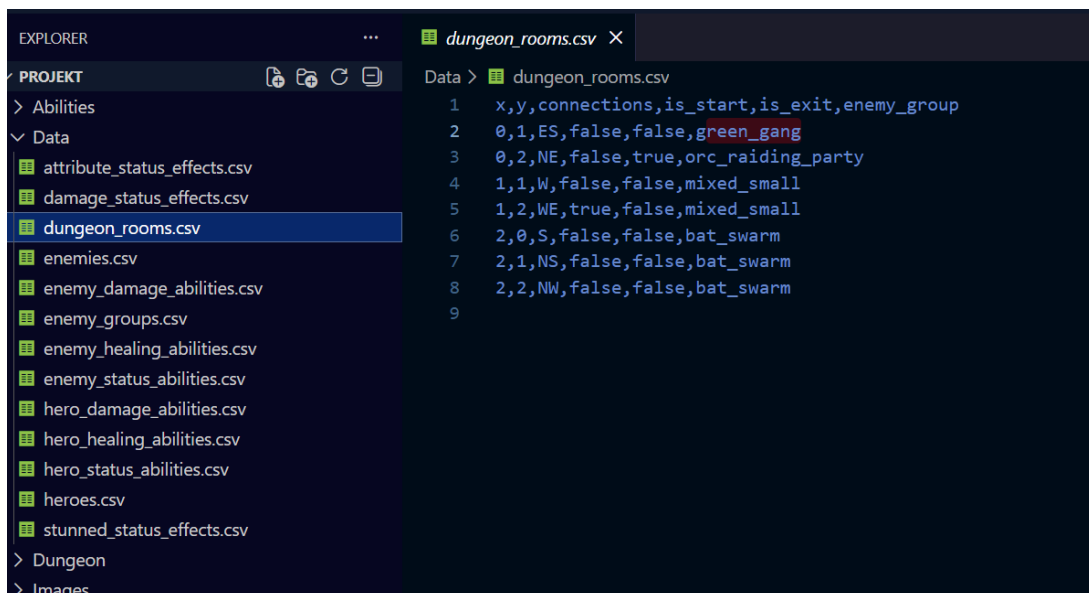
Svaki heroj ima svoj stil glasanja. Ovaj stil određuje njegove prioritete u kontekstu glasanja sljedeće prostorije u koji će heroji ići. Tablica 2 sadrži stilova glasanja s njihovim opisima.

Tablica 2: Stilovi glasanja

Naziv	Opis
Kukavički (eng. <i>Cowardly</i>)	Ako je pronađen, glasa za izlaz iz tamnice. Inače glasa za prostoriju s najmanje neprijatelja.
Avanturistički (eng. <i>Adventurous</i>)	Glasa za prostoriju s najviše neprijatelja.
Znatiželjan (eng. <i>Curious</i>)	Glasa za prostoriju s najmanje neprijatelja.

4.2. Inicijalizacija simulacije

Cijeli projekt je implementiran na način da pruža određenu razinu fleksibilnosti, prilagođavanja i personalizacije simulacije. Svi NPC-i, njihove sposobnosti te i sama tamnica se učitavaju iz većeg broja CSV (eng. *Comma-Separated Values*) datoteka, što znači da dodavanje novih NPC-a, sposobnosti ili izmjena tamnice ne zahtjeva nikakvo programiranje nego isključivo izmjenu ovih datoteka. Slika 2 prikazuje listu datoteka zajedno s datotekom za definiranje tamnice.



Slika 2: Datoteke za učitavanje podataka

Za učitavanje ovih podataka koristi se klasa *GameDataLoader* koja priprema sve podatke za ostatak simulacije; učitavanje NPC-ova i pridruživanje sposobnosti te učitavanje prostorija tamnice i njihovo povezivanje. Isječak koda 3 prikazuje metodu za učitavanje heroja.

```

1  def load_heroes(self):
2      heroes = {}
3      with open(HEROES_FILE, newline="", encoding="utf-8") as f:
4          reader = csv.DictReader(f)
5          for row in reader:
6              fighting_style_str = row.get("fighting_style", "CHAOTIC").upper()
7              try:
8                  fighting_style = FightingStyle[fighting_style_str]
9              except KeyError:
10                 fighting_style = FightingStyle.CHAOTIC
11
12             voting_style_str = row.get("voting_style", "CURIOUS").upper()
13             try:
14                 voting_sytle = VotingStyle[voting_style_str]
15             except KeyError:
16                 voting_sytle = VotingStyle.CURIOUS
17
18             hero_id = row["id"]
19             hero = Hero(
20                 name=row["name"],
21                 max_hp=int(row["max_hp"]),
22                 speed=int(row["speed"]),
23                 icon=row["icon"],
24                 fighting_style=fighting_style,
25                 voting_style=voting_sytle,
26             )
27             heroes[hero_id] = hero
28
29         self.load_damage_abilities(heroes, HERO_DAMAGE_ABILITIES_FILE)
30         self.load_healing_abilities(heroes, HERO_HEALING_ABILITIES_FILE)
31         self.load_status_abilities(heroes, HERO_STATUS_ABILITIES_FILE)
32         return list(heroes.values())

```

Isječak koda 3: Metoda za učitavanje heroja

4.3. Petlja igre

Petlja igre (eng. *Game Loop*) započinje u nasumično odabranoj ulaznoj prostoriji tamnice. Ako prostorija sadrži neprijatelje započinje borba. Ako prostorija ne sadrži neprijatelje, započinje glasanje o prostoriji u koju će heroji sljedeće ići. Borba se izvršava u potezima. NPC-i s većom brzinom imat će svoj potez prije NPC-a s manjom brzinom. Na početku poteza NPC na sebe primjenjuje relevantne statusne efekte ako ih ima. Zatim bira sposobnost. Odabrana sposobnost ovisi o više parametara. NPC ne može odabrati sposobnost koja je na hlađenju. Ako ju ima te ako postoje ozljeđeni saveznici, NPC će odabrati sposobnost ozdravljivanja. AoE sposobnost ozdravljivanja će iskoristiti samo ako postoje 2 ili više ozljeđena saveznika. Iduće po prioritetu su statusne sposobnosti. Napadačke sposobnosti su najmanjeg prioriteta. Nakon odabrane sposobnosti, NPC bira metu. Ako je odabrana napadačka sposobnost, NPC će odabrati metu u skladu sa svojim stilom borbe. Na ovaj način svaki NPC izvršava svoj potez sve dok jedan tim ne izgubi. Ako neprijatelji ubiju sve heroje, simulacija završava. Ako heroji ubiju sve neprijatelje, započinje proces glasanja.

Kada heroji uđu u prostoriju, otkriju sve njezine susjedne prostorije. Sve prostorije koje su i otkrivene i neposjećene postaju kandidati kod glasanja u koju će prostoriju sljedeće ići. Ako je otkrivena i posjećena prostoriju s izlazom iz tamnice, izlaz također postaje kandidat. Svaki heroj glasa u skladu sa svojim stilom glasanja. Ako neki kandidati dobiju jednak broj glasova, pobjednik se bira nasumično.

Heroji su pobijedili ako su posjetili sve prostoriju i porazili sve neprijatelje ili ako uspješno izađu iz tamnice. Heroji su izgubili ako svi poginu tijekom borbe s neprijateljima.

5. Prikaz rada aplikacije

Ovo poglavlje prikazuje kako izgleda simulacija dok se izvodi. Prvo su predstavljene unaprijed definirani podaci koje simulacija koristi, a zatim je ukratko prikazano sučelje i odvijanje simulacije.

5.1. Pregled postavki simulacije

Ovo poglavlje pruža pregled unaprijed definiranih podataka unutar CSV datoteka. Ovi podaci su demonstrativni i vrlo ih je lako promijeniti.

5.1.1. Heroji

Po unaprijed zadanim podacima, postoje 4 heroja. Svaki heroj ima po tri sposobnosti. Ratnik i rendžer fokusiraju se na davanje štete neprijateljima. Glavna uloga svećenika je ozdravljanje svoj suboraca. Uloga barda je motiviranje svojih saveznika pozitivnim statusnim efektima i slabljenje neprijatelja negativnim statusnim efektima.

Tablica 3: Heroji (ikone preuzete s [4])

Naziv	Ikona	Max HP	Brzina	Stil borbe	Stil glasanja
Ratnik (eng. <i>Warrior</i>)		22	5	Odvažan	Avanturistički
Rendžer (eng. <i>Ranger</i>)		18	8	Oportunist	Avanturistički
Svećenik (eng. <i>Priest</i>)		14	5	Oportunist	Znatiželjan
Bard (eng. <i>Bard</i>)		14	8	Kaotičan	Kukavički




Tablica 4: Sposobnosti heroja (ikone preuzete s [4])

Heroj	Naziv	Ikona	Hlađenje	AoE	Opis
	Osnovni na-pad		0	Ne	2-3 štete (eng. <i>Da-mage</i> , skraćeno dmg)
	Zamah sjekir-om		3	Da	2-4 dmg
	Glavosjek		3	Ne	5-6 dmg
	Osnovni na-pad		0	Ne	2-3 dmg
	Probijajuća strijela		3	Ne	4-5 dmg
	Vatrena strijela		3	Ne	2 dmg/potezu, traje 3 poteza
	Osnovni na-pad		0	Ne	1 dmg
	Sveto svjetlo		3	Da	3 dmg
	Ozdravljajuća molitva		3	Ne	+5-7 HP
	Osnovni na-pad		0	Ne	1 dmg
	Epska pje-sma		6	Da	+3 brzine na 3 poteza
	Loša melo-dija		3	Ne	Ošamujućuje neprijatelja na 1 potez













5.1.2. Neprijatelji

Po unaprijed zadanim podacima postoje 3 neprijatelja od kojih svaki ima dvije sposobnosti. Šišmiši su pojedinačno slabi, ali često dolaze u grupama i mogu usporiti neprijatelje. Goblini su srednje teški neprijatelji koji nanose štetu. Ork je teži neprijatelj koji može nanijeti puno štete i ošamutiti neprijatelje.

Tablica 5: Neprijatelji (ikone preuzete s [4])

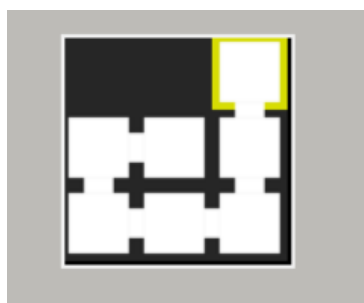
Naziv	Ikona	Max HP	Brzina	Stil borbe
Šišmiš		6	7	Kaotičan
Goblin		8	6	Oportunist
Ork		12	3	Pomahnitali

Tablica 6: Sposobnosti neprijatelja (ikone preuzete s [4])

Neprijatelj	Naziv	Ikona	Hlađenje	AoE	Opis
	Ugriz		0	Ne	2-4 dmg
	Vrisak		4	Da	-2 brzine na jedan potez
	Udarac mačem		0	Ne	2-4 dmg
	Ubod u leđa		3	Ne	4-6 dmg
	Zamah sjekirrom		0	Ne	4-7 dmg
	Iznenadni udarac		3	Ne	Ošamućuje metu na 2 poteza

5.1.3. Tamnica

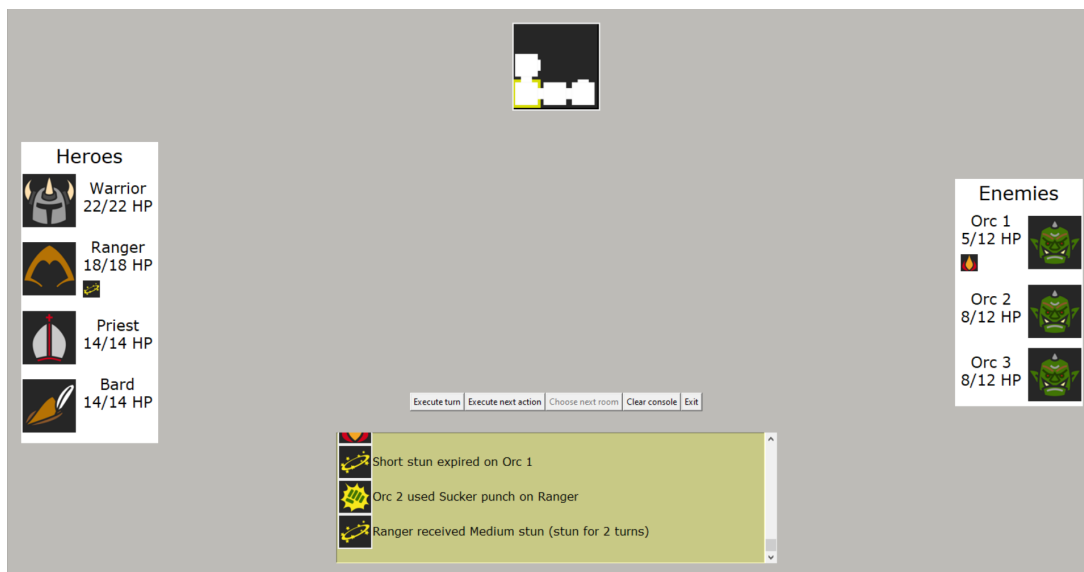
Unaprijed definirana tamnica ima 7 prostorija s jednim ulazom i jednim izlazom. U te prostorije su raspoređene različite grupe neprijatelja; pljačkaške skupine orkova, rojevi šišmiša i skupine goblina. Slika prikazuje u potpunosti istraženu mapu.



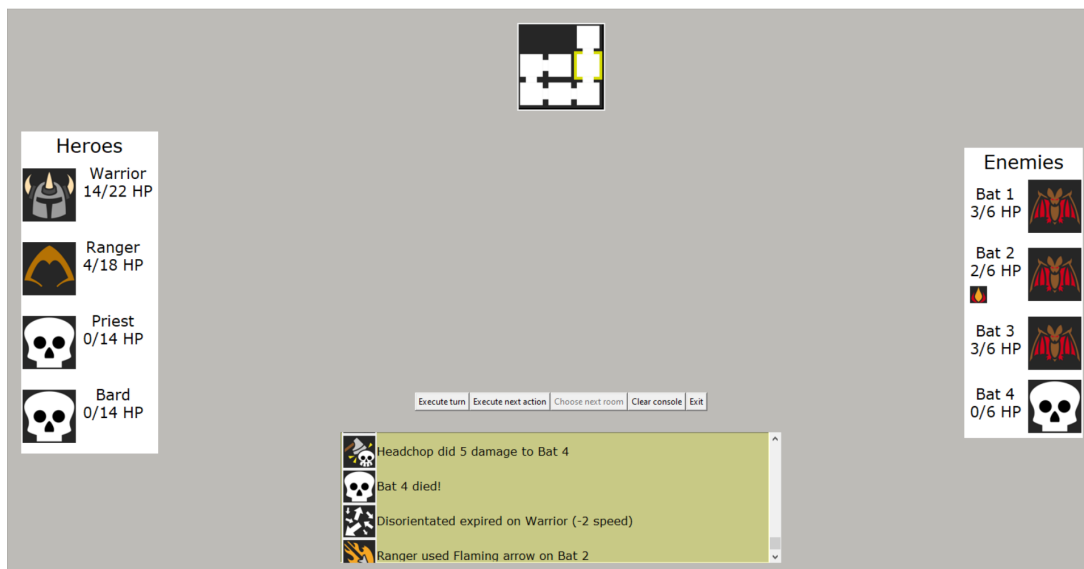
Slika 3: Istražena mapa

5.2. Izvođenje simulacije

Slike 4 i 5 prikazuju simulaciju u različitim stadijima izvođenja. Korisničko sučelje s lijeve strane prikazuje heroje, s desne neprijatelje. Na dnu je konzola koja ispisuje sve događaje, poput korištenih sposobnosti, statusnih efekata i rezultata glasanja. Iznad konzole su gumbi za kontroloiranje simulacije. Pri vrhu je karta tamnice. Soba u kojoj su heroji trenutno označena je žutom bojom. Karta nije u potpunosti vidljiva na početku simulacije nego se prostorije otkrivaju kako heroji prolaze kroz njih.

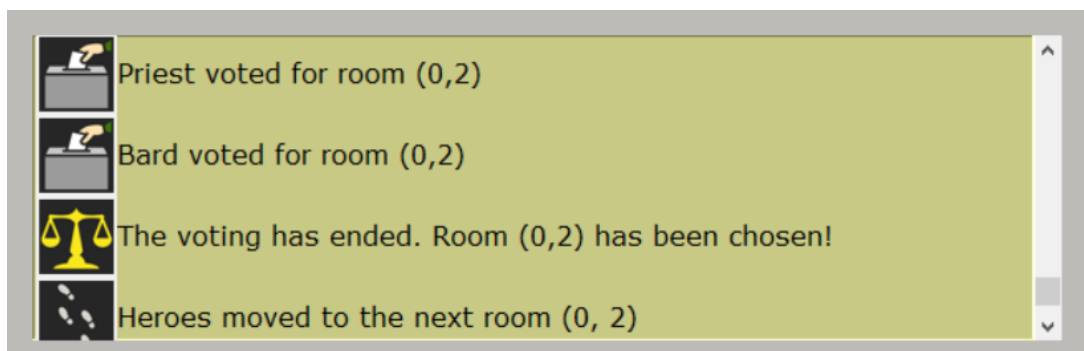


Slika 4: Simulacija u izvođenju 1



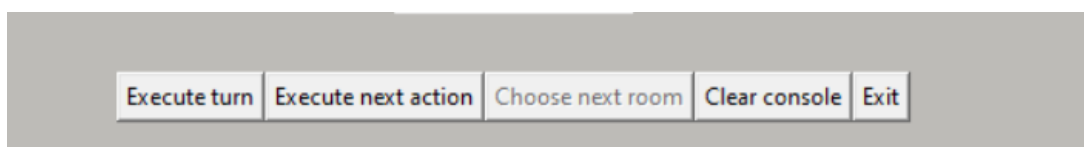
Slika 5: Simulacija u izvođenju 2

Slika 6 prikazuje ispise u konzoli tijekom glasanja. Glasanje se provodi nakon pobijedenih borbi ili nakon prijelaza u prostoriju u kojoj nema neprijatelja.



Slika 6: Konzola nakon glasanja

Slika 7 prikazuje gumbe za kontroliranje simulacije. Gumb *Execute turn* izvršava potez svih NPC-ova, dok gumb *Execute next action* izvršava potez pojedinačnog NPC-a koji je sljedeći na redu. Gumb *Choose next room* započinje glasanje. Gumb *Clear console* briše tekst iz konzole, dok gumb *Exit* izlazi iz simulacije.



Slika 7: Gumbi za kontrolu simulacije

Simulacija je izvršena veći broj puta koristeći podatke prikazane u prijašnjim poglavljima. Pri tome su dobiveni vrlo različiti rezultati; pobjeda heroja sa svima preživjelima, pobjeda heroja s nekoliko poginulih ili potpuni poraz heroja. Ovo pokazuje da postoji određena razina nasumičnosti u simulaciji i da ponašanja agenata variraju.

6. Kritički osvrt

Implementirana simulacija dungeon crawler igre temeljena na višeagentnom sustavu pokazuje da je odabrani pristup praktično izvediv te prikladan za modeliranje složenih interakcija između autonomnih agenata. Simulacija je na određenoj razini demonstrirala autonomnost, suradnju i odlučivanje agenata. S praktičnog aspekta, korištenje programskog jezika Python omogućilo je brzu implementaciju i jednostavno testiranje. Međutim, na temelju vlastitog iskustva, moglo bi se zaključiti da se kod izrade većih projekata u Pythonu dobar dizajn lakše zapostavi i napusti u odnosu na strogo tipizirane jezike. To je uočeno i pred kraj ovog projekta, ali to također može biti i rezultat manjka iskustva i vještine programiranja u Pythonu.

Jedna od potencijalnih primjena ove ili slične simulacije jest balansiranje videoigara. U razvoju videoigara iznimno je važno izbjeći stvaranje vještina koje su preslabe ili premoćne, budući da je cilj igraču pružiti zabavno iskustvo uz odgovarajuću razinu izazova. Zbog toga se mehanike igara često naknadno prilagođavaju kako bi se postigao što uravnoteženiji sustav. Proces balansiranja u pravilu zahtijeva puno testiranja, dok bi primjena ovakve simulacije omogućila automatizirano ispitivanje različitih scenarija. Ako je simulacija dovoljno fleksibilna i proširiva, kao što je ostvareno u ovom projektu, promjena vrijednosti koje se testiraju može se provoditi brzo i učinkovito.

7. Zaključak

U ovom radu prikazana je simulacija dungeon crawler igre temeljena na višeagentnom sustavu. Cilj rada bio je prikazati mogućnosti primjene višeagentnih sustava u kontekstu simulacije ponašanja likova u videoigrama. Kroz teorijski dio rada objašnjeni su osnovni koncepti agenata, njihove komunikacije i okruženja te procesa glasanja. U praktičnom je dijelu opisana implementacija simulacije i način funkcioniranja pojedinih komponenti sustava. Implementirano rješenje pokazalo je da je višeagentni sustav prikladan način za implementaciju simulacije dungeon crawler igre, posebno ako ta igra koristi borbu temeljenu na potezima te ima dobro definirana pravila i stanja agenata. Cjelokupni izvorni kod je dostupan na <https://github.com/NinoLoncar/dungeon-crawler-simulation>.

Popis literature

- [1] B. P. Gokulan i D. Srinivasan, *Innovations in Multi-Agent Systems and Application – 1*. Springer, Berlin, Heidelberg, 2010.
- [2] M. Huhns N. i L. Stephens M., „Multiagent Systems and Societies of Agents,” 1998.
- [3] J. Pitt, L. Kamara, M. Sergot i A. Artikis, „Voting in Multi-Agent Systems,” *The Computer Journal*, sv. 49, str. 156–170, 2006.
- [4] Lorc, Delapouite i drugi. „Games-icons.net,” pogledano 15. siječnja 2026. adresa: <https://game-icons.net/>

Popis slika

1.	Dijagram stanja heroj agenta	6
2.	Datoteke za učitavanje podataka	8
3.	Istražena mapa	13
4.	Simulacija u izvođenju 1	14
5.	Simulacija u izvođenju 2	14
6.	Konzola nakon glasanja	15
7.	Gumbi za kontrolu simulacije	15

Popis tablica

1.	Stilovi borbe	7
2.	Stilovi glasanja	8
3.	Heroji (ikone preuzete s [4])	11
4.	Sposobnosti heroja (ikone preuzete s [4])	12
5.	Neprijatelji (ikone preuzete s [4])	13
6.	Sposobnosti neprijatelja (ikone preuzete s [4])	13

Popis isječka koda

1.	Klasa <i>Npc</i>	5
2.	Klasa <i>Ability</i>	7
3.	Metoda za učitavanje heroja	9