Универзитет у Новм Саду
Факултет Техничких Наука
Нови Сад

Пројекат “Turn Snake”
Тим 18

Гаврило Продановић, ПР 158/2015
Коста Ђурчић, ПР 60/2015

Дистрибуирани рачунарски системи у ЕЕ
- Примењено софтверско инжењерство -

Нови Сад, 15.1.2019.

Садржај

Општи рад апликације

Ошти рад апликације се може сагледати са сегмента механике игре и са сегмента софтверске имплементације

Механика игре

Апликација „Turn Snake“ је стратешко акциона игра базирана на потезима која се игра од 2 до 4 играча. У елементима игре сваки играч посједује од једне до три змије којима може да се креће по мапи терена са циљем да би заробио змије својих противника. Свака змија од карактеристика посједује своју дужину, број корака и боју припадности.

Игра се састоји од двије фазе које се понављају док се не добије побједник:

- Фаза планирања – Играчи паралелно планирају путање својих змија које ће бити извршене у следећој фази. За сваку змију коју играч посједује мора да испланира путању довољно дугу да би покрио број корака који змија посједује. Ако играч испланира краћу путању она ће бити аутоматски продужена до преосталог броја корака користећи последњи правац у планирању. Играч може да види само своје путање у процесу планирања, путање змија других играча су непознате играчу који планира. Фаза планирања је временски ограничена.
- Фаза извршења путања – извршавају се путање за све змије корак по корак, помјерајући њихове позиције. Послије сваког корака провјерава се да ли је дошло да колизије између змија. Ако једна змија својом главом додирне тијело друге змије, змија која је „ударила“ својом главом губи свој живот и бива избачена из игре. Фаза извршења планирања траје док све змије не обаве своје кораке. Ако на крају ове фазе остану змије само једног играча, игра се сматра завршеном, а тај играч побједником.

Игра посједује и следеће секундарне елементе:

- Зид – непробојан, било која змија која удари у њега губи свој живот.
- Храна – храна се креће независно, праволинијски неколико корака. Змија може да поједе храну и тако да продужи своје тјело или да повећа број корака.
- „Deus ex machina“ – Специјална врста хране која када се покупи или одузме живот једној змији сваког противника или одузме живот змији која ју је појела.

Софтверска имплементација

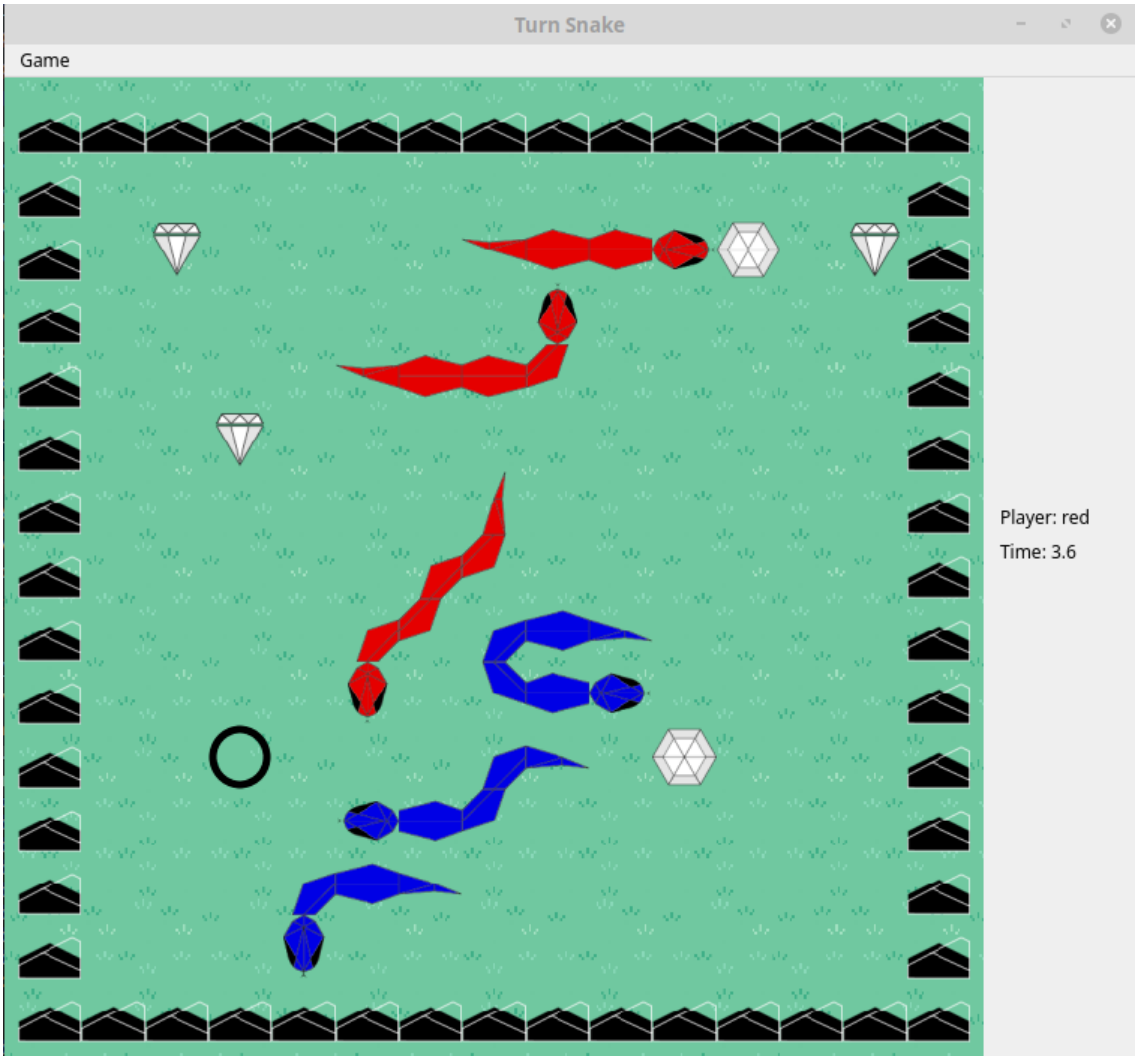
Општа архитектура софтверске имплементације је комбинација серверско-клијентске архитектуре и вишеслојне архитектуре

Са серверско-клијентског погледа на архитектуру примјећујемо следеће компоненте:

- Сервер – садржи стање игре, управља механиком игре, прима команде од клијената које извршава и обавјештава клијенте о новом стању унутар игре.
- Клијент – Приказује кориснику тренутно стање игре кроз графичку представу. Приме од клијента команде које прослеђује серверу. Од сервера прима обавјештења о новом стању игре које приказује кориснику.

Са вишеслојног архитектурног погледа примјећујемо следеће слојеве:

- Слој механике игре – Садржи ентитете који учествују игру и механику понашања тих ентитета који граде правила игре.
- Слој мрежне комуникације – Садржи дефиницију протокола, серверске и клиејнтске компоненте које омогућују да се механика игре дистрибуира на више рачунара.
- Слој презентације – Садржи графику која представља ентитете из механике игре. Садржи интерфејс којим се непосредно управља ентитетима игре.



Слика 1: Слика екрана једне партије игре

Предности и мане Пајтон језика

Пајтон (енгл. *Python*) је програмски језик високог нивоа опште намјене. Подржава императивни, функционални и објектно-оријентисани стил програмирања. Није строго типизиран и извршава га интерпретер. Ове основне особине пајтон језика истовремене су његове предности и мане у зависности од проблема који се њиме рјешава.

Предности Пајтон језика:

- Због интерпретерске природе програми писани у Пајтон језику су подразумијевано више платформски.
- Због интерпретерске природе веома га је лако уградити у комплексне програме да буде АПИ за писање плагинова.
- Постоје различите имплементације интерпретера базиране на већ постојећим виртуелним машинама, па је због тога нпр. постојећи Јава код лако интегрисати у Пајтон.
- Писање малих програма је веома брзо зато што није стриктно типски оријентисан и јер посједује велику стандарду библиотеку која долази уз интерпретер.
- Стил писања кода је саставни дио Пајтон синтаксе па су програми писани у пајтону увијек структурно уредни.

Мане Пајтон језика:

- Због интерпретерске природе код може спорије да се извршава у односу на еквивалентан код писан у језику који се компајлира.
- Због интерпретерске природе програмски код је тешко сакрити када се програмско рјешење испоручује у продукцију.
- Рад на великим пројектима и у великим групама може бити веома отежано зато што Пајтон није стриктно типски оријентисан.
- IDE алати теже могу да предвиде и понуде жељено аутоматско допуњавање кода зато што је тип промјењивих често у потпуности непознат.

Предности и мане PyQt5 оквира

Qt је општи вишеплатформски апликациони интерфејс, који се користи за развој графички оријентисаних програма, али и за развој неграфичких програма, попут конзолних сервера. Прва верзија се појавила 1995. године, а до дан данас *Qt* се активно развија. Писан у C++ језику.

PyQt5 Пајтон библиотека која врапује (енгл. *wrapping*) *Qt* интерфејс писан у C++ језику да би се могао лако користити у Пајтон програмима

Предности *PyQt5* оквира:

- Вишеплатформска библиотека која корисника библиотеке сакрива од детаља конкретне имплементације функција које су уско везане за оперативни систем када је у питању цртање графике
- Омогућује једноставно писање комплексних графичких програма
- Подржава специјализоване *threading* библиотеке које олакшавају паралелни рад у графичким апликацијама

Мане *PyQt5* оквира:

- Не постоји дизајнер који олакшава израду распореда графичких компоееннти
- Распоред компоненти је дефинисан и уско везан уз изворни код

Паралелизација рада

Пајтон програмски језиг у стандардној библиотеци нуди подршку за више-процесну и више-нитну паралелизацију рада. Такође нуди све потребне класе које омогућавају синхронизацију и комуникацију између нити и/или процеса.

Веома је битно при паралелизацији рада бити свјестан да више-нитна паралелизација у CPython имплементацији дјеле једну системску нит тако да се не може упослити више процеса уколико их систем посједује. За праву паралелизацију потребно је користити више-процесну паралелизацију

Закључак

Уколико је потребно написати мању апликацију или скрипту пајтон је одличан избор због позитивних особина које су наведене у овом документу, поготово због његове велике библиотеке и популарности коју посједује.

Конкретно на изради овог пројекта: величина пројекта је дошла приближно критичној тачки гдје непостојање строге типизираности је почело успоравати писање кода. Строга нетипизираност језика је представљала и проблем у комуникацији тима јер је често доводило до потребе да се додатно објашњавају параметри и повратне вриједности функција.

Битно је нагласити због уграђене подршке за синхронизацију објеката мрежно програмирање је било знатно олакшано него у случају других језика.

PyQt5 оквир је био потпуно погрешан избор за израду графичког дјела игре и веома је отежао процес графичког развоја и није могао дати добре перформансе јер није постојала могућност директног приступа графичком хардверу