CPSC 2720 FALL 2019

CastleScape

Team Navi

Jorge Aceytuno, Gavin Booth, Ben Hunt, Wes Waldern

October 20, 2019

TABLE OF CONTENTS

INTRODUCTION

Our group was tasked with developing a text-based adventure game. We chose to create a fantasy game complete with castles, dungeons, monsters, and damsels in distress. This project is an exercise to employ Object-Oriented Analysis and Design alongside good coding principles like SOLID and DRY. In this document, we discuss group organization, risk management and the development and design process. Each member is assigned the roles: team lead, quality lead, design lead and documentation lead. Our risk management outlines procedures we have in place should our design need re-evaluation, conflicts arise within the group, or we have problems with certain development tools. Our development process provides a framework for how we will communicate, resolve bug reports, and pull requests, and discusses our design for the text-based adventure game.

PROJECT MANAGEMENT

TEAM ORGANIZATION

As a team, we are committed to achieving common goals and objectives and produce the expected results.

The team organization to accomplish our goal will be formed with the following members with the described assignments to supervise, also the understanding that even though there will be a leader in each field, the job will be done as a team during the whole project.

Gavin Booth, as Team Leader, will oversee the whole project on track from the beginning to the end. His focus will be accomplished according to the adding paragraph about risk management. He will control the finalize review of the codes after discussion with the team members.

Ben Hunt, as Design Software Leader, will be the one ensuring that the project follows good object-oriented analysis Design. He will translate the ideas presented into UML and sequence diagrams that are needed during this phase.

Wesley Waldern, as a Development Process Leader, will facilitate coordination and communication tools, channels and meetings are according to described in the paragraph related below the description of this organization.

Jorge Aceytuno will write the form that the team will be organized and as a Quality Assurance Leader maintaining track that the requirement on the project is successfully met through analysis and testing control.

RISK MANAGEMENT

During development, various issues may arise hindering our completion. This section details plans for various scenarios separated into three different groups and how we will handle them.

REQUIREMENTS/DESIGN/ESTIMATION

As for most (if not all) of us, this project marks our first contributions to a larger scale system. With this lack of experience, errors may be made in regards to meeting requirements, proper design practices and under or overestimating any aspect of this project. In regards to meeting requirements, to ensure we design and build our project to the required specifications we will, as a group, try to keep each other on track with which aspects of the project contribute to or fulfill some aspect of the requirements and prioritize as such. For example, the team leader Gavin would like to try and compose some looping 8-bit music to play during gameplay which in no way is a part of the requirements. As such this will only be tended to when all other aspects that do fulfill requirements are completed. To ensure other such “unneeded complexities” are not introduced we will refer frequently back to our requirements. This should also help us to keep all of the requirements in mind so we can be sure we are meeting all of the requirements.

In regards to proper design practices, this will, again, be a group effort to keep each other on track. We have already been exemplifying this during the design phase of our project. Each data member and method is discussed in a group setting as to if it violates any of the design practices we have learned so far in class and, if it does, discussing how we can modify our design to maintain a SOLID design (pun intended) for our project.

In regards to the possibility of under or overestimation of project aspects, we plan to maintain a good degree of flexibility in our project through adherence to the Open/Closed principle. This should allow us to make changes from our original estimates to accommodate any surprises with sections taking less time than anticipated or sections taking more than anticipated. To ensure these probable changes to our system do not affect our ability to meet our requirements, the core classes of our system will be the first task during our implementation phase. We will aim to complete a working system with very little content in the way of rooms, items, NPCs and intractable elements as our flexible design should allow for easy content creation. Pushing for this goal first in our implementation should let us see very clearly where we stand progress wise in comparison to our original estimations allowing us to adjust earlier and more frequently to accommodate changes along the way. Should these kinds of issues be localized to a specific member of the project, for example, a member taking significantly longer on a section, we will assist in whatever way that team member requires.

PEOPLE

Assuming all members of our team are human, issues may arise relating to events involving team members. Such issues may be personal or may be interpersonal within the group. While some of these issues can be avoided through a mature and professional demeanour others may be unavoidable.

In the circumstance of an interpersonal issue, for example if there were to be a disagreement on a design or implementation proposition, provided we are able to maintain an open and professional discussion on the merits of each proposed solution we can simply work through the pros and cons of each solution until we are able to reach a consensus. If a member is not willing to behave professionally they will be asked to take whatever time or actions(within reason) they require to return to the discussion with an open and professional attitude. If such behaviour occurs frequently enough to create additional issues as far as deadlines are concerned we will consult Dr. Anvik as to potential changes in group membership. If a consensus cannot be reached the team leader will decide on which option the group will use. Team members may then either accept the decision made or request the group consult an

outside source like Nicole or Dr. Anvik as to grant further insight into potential repercussions that may affect the decision whether the information gleaned is of a suggestive nature or if one of the options is improper or incorrect in some way, shape or form.

Through the course of this project, we may encounter issues of a personal nature. We may have a team member leave. In this circumstance, assignments may be rearranged and sections cut due to reduced manpower. As our group currently has four members it seems rather unlikely we will have a new member, however, there is the possibility that more than one member will leave potentially creating a situation where we may take on an additional member. In this circumstance, we will meet as a group and review the current state of the project and discuss assignments for members as well as possible changes the "new" group might pursue due to such group restructuring. Such changes may be to the group structure or perhaps even to the design of the project. We may also have an individual who is unproductive, unable to meet deadlines or delivering unsatisfactory work for a variety of reasons. Our actions will depend on the specific reasoning for the missed deadline(s), unproductive behaviour or unsatisfactory work. Some reasons for these outcomes include but are not limited to a major life event, lack of familiarity with aspects of the project whether they be technical (e.g. C++ syntax, libraries) or a conceptual nature (e.g. design practices) or a lack of time/effort committed. In the event a member has a major life event that will affect their ability to contribute to the project they are expected to bring this information forward to the group as soon as they are able so the group can adjust as needed to lighten the load for the individual so they can continue to participate in the group. Depending on how significant the event is Dr. Anvik may be included so he may oversee our workload adjustment for the individual to ensure they are contributing enough to constitute membership in the group. If an individual lacks the knowledge required to complete their assigned section(s) we will try to accommodate this individual with assistance from another group member. Any group member (or multiple group members) should participate in this assistance when circumstances allow or other channels for assistance may be pursued, such as consulting an instructor or consulting The One Almighty Google. If after a reasonable amount of assisting or tutoring said individual does not improve their contributions in a meaningful way sections may be reassigned to an individual who's abilities are better suited to the assignment. If an individual's work is late or unsatisfactory due to a lack of time or effort dedicated the group will potentially adjust assignments if this circumstance came about from an individual having a significant amount of pressure or urgency from other classes. If this is not the case and the issue rather stems from poor time management the group will work to put us back on track for deadlines and the event will be documented so we can keep track of repeated events. If an individual's work suffers from poor time management frequently enough that it jeopardizes the project their membership will be discussed with Dr. Anvik.

LEARNING & TOOLS

Through the duration of this course and project, we will be exposed to and expected to learn a variety of new concepts and tools. As these tools and concepts are new to all of us, it is within reason to expect some of us may learn specific tools or concepts faster or slower than others. This introduces new situations and issues that may stem from an individual's ability to learn a particular tool or concept. As the participation in the course implies everyone in the course is expected to learn any individual tool or concept if an individual has an issue with a tool or concept we will first try and tutor or assist this individual on the tool or concept they are having issues with additional support channels through instructors or The One Almighty Google. If we are unable to improve this individual's grasp on the topic in a significant way in a reasonable amount of time assignments may be redistributed to avoid requiring said individual to use the tool or concept they have issues with. If all members lack proficiency with a tool or concept all members will participate in a study session where we will work together to learn what is

needed to complete the task requiring the specific tool or concept. All members are expected to participate so that we avoid a situation where only one individual is familiar with a tool or concept as if an unrelated issue arrises for this individual we have other members in the group that can assist or take over the task.

DEVELOPMENT PROCESS

Communication allows for our team to easily and effectively tell one another what parts of the assignment we are working on, which will increase our efficiency. Communication for our group is the ability to effectively and efficiently work on the project as a whole. During the development of the project, our team will need to communicate. Communication will allow us to handle the review process of the assignment. We have decided on communication methods to keep the team up to date on meetings, requests, and any possible conflicts so we can deal with them beforehand. Finally, we have decided how issues such as merge conflicts or bugs are dealt. Many other issues would arise if we didn't have these communication skills between our team members.

CODE REVIEW PROCESS

The code review process will consist of the group coming together in a meeting and reviewing the code. An example of reviewing the process is a pull request. The group will review the code that is written and discuss if it needs improvements. Once we are done discussing Gavin will be the one to pull to the master branch. This will result in more open communication between group members and it will reduce the number of conflicts and errors that may appear. During this phase we have created a design as a group so we can organize and plan our implementation. This singular plan gives us all a point to reference so as to avoid issues of members unfamiliar with our plans during implementation.

COMMUNICATION TOOLS/CHANNELS

Communication allows us to be more open, so everyone knows what's happening with the project. We will have multiple channels of communication with the group. Discord is our main form of communication. We picked discord due to it being freely accessible on multiple platforms, as well as, the ability to send files, images and group voice calls. This opens our communication by allowing each group member to post what they have and be able to explain issues easily. We have also created a shared google drive folder to share files for easy accessibility to any group member. This also allows as an added bonus to have our information across multiple platforms. Other communication methods that we will be using are email and a spreadsheet. Email will be reserved for one time messaging, such as sending a

meeting location and time. The spreadsheet will be used as a chart logging current and past development. Each member should be logging any work in the designated spreadsheet.

CHANGE MANAGEMENT

During the oncoming phases, we will need to discuss how we will deal with issues that arise with our code, whether that be bugs or conflicts. The management of these will be discussed with the group and our team lead Gavin will compile a list prioritizing this issues discussed. These priorities will keep the group on track on which bugs have been managed or not. The group will discuss these issues and come up with a solution. Bugs will be reported to the GitLab issue tracker with the possibility of confirming a bug isn't an intended feature through a discord message. When the bug report is issued everyone will review it. If the individual who wrote the code feels confident in their understanding of the solution discussed with the group they will have the responsibility of implementing the fix. If this is not the case another group member will be assigned. With issues such as conflicts of code, the group will look over the conflicts and discuss how they can be fixed. After discussing the issue if we can't decide which method to implement Gavin will decide which direction we will take. With each issue, it will be reported to the group, as to notify everyone of what needs to be fixed. However, Gavin will be the one who manages these delegations to each issue.

SOFTWARE DESIGN

DESIGN

The game relies on the principle that all things are objects. To that end, we created a superclass Object from which all subclasses will inherit. From there, we can define other objects: Person, Item, Room, and RoomObjects. Person is an abstract class from which all characters in-game have access. From Person, we have classes Hero (our player), Villager, and Enemy. We also realize that certain objects are used by other objects thus our Item class which, by way of enumeration, composes the items that can be used by the player (i.e. keys, weapons, and potions). Finally, we have rooms and objects in those rooms. Room relies on RoomObjects allowing the composition of rooms with chests, buttons and locks (each of which has a separate class).

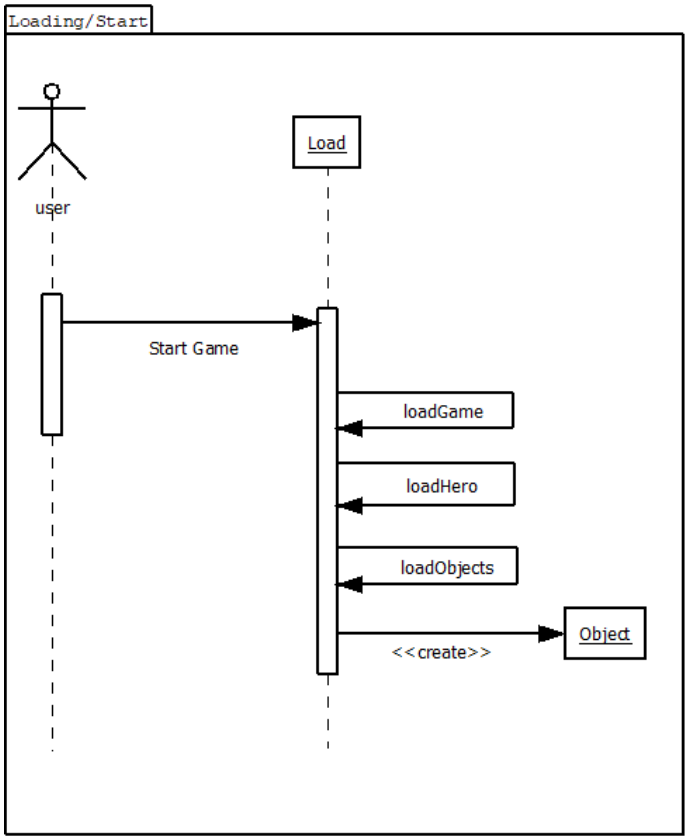
DESIGN RATIONALE

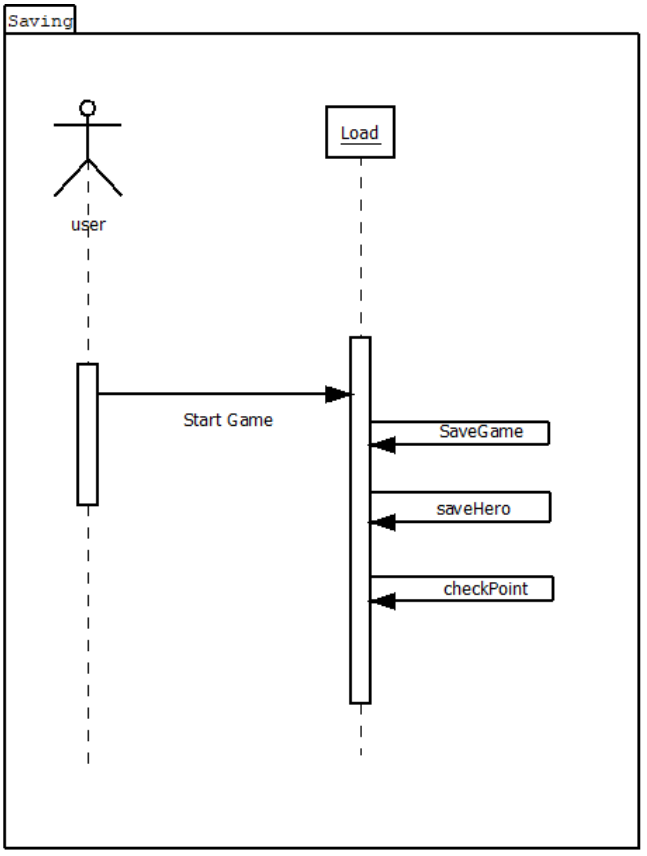
Being our first experience with designing something of this magnitude, we attempted to employ the principles and design patterns taught in class. Our hierarchy of objects is an attempt to employ

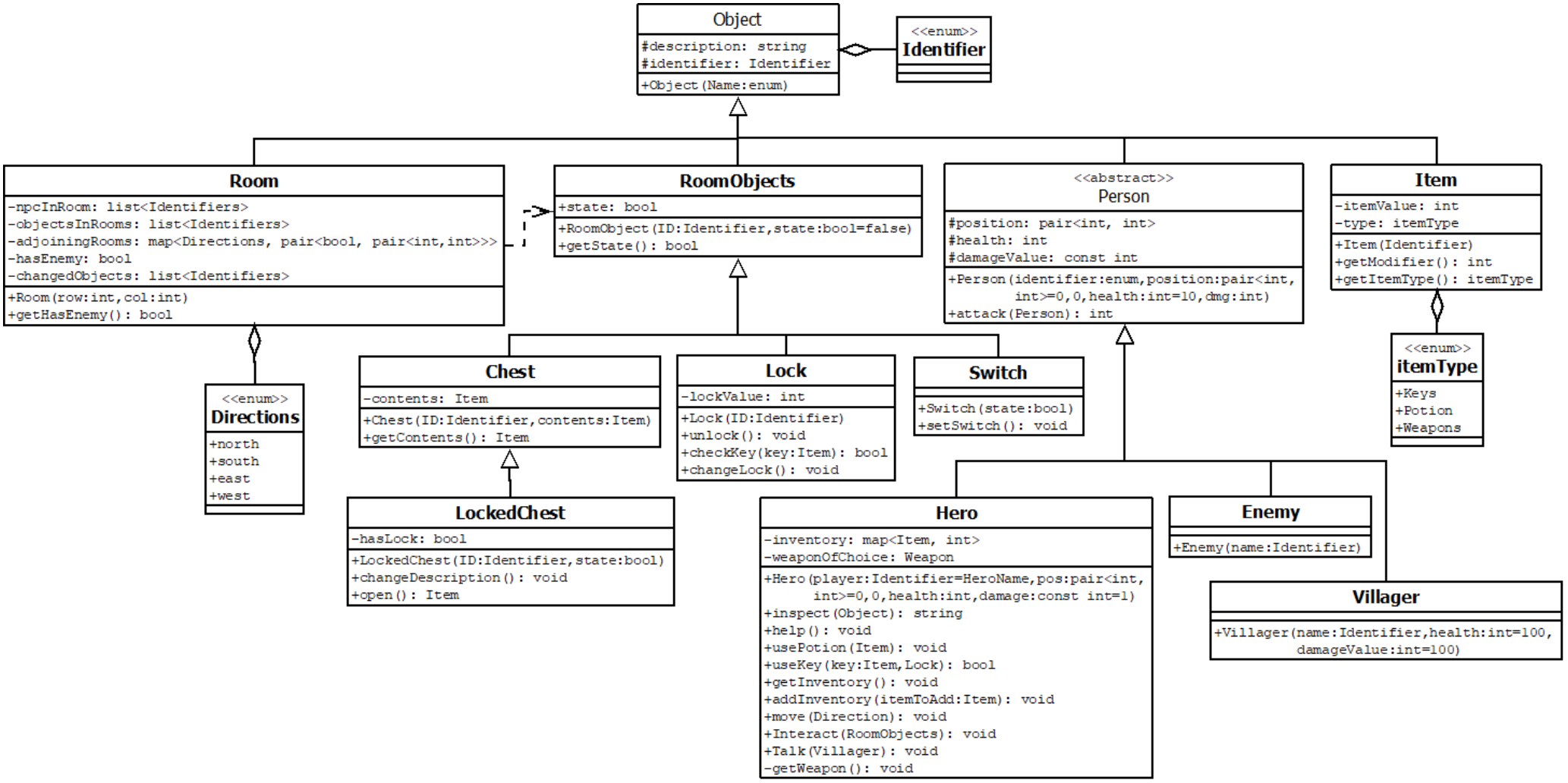
inheritance, SOLID and DRY. We rely heavily on inheritance in this current iteration of the design, as such we aim to reuse much of the code created, seeking to make an implementation that much better. We also wanted to stress Single Responsibility principle, again for ease of use during our implementation. We wanted to create a “loose” environment where tight coupling is not present. Using Objects means we can have classes interact with one another in a general sense, thus we attempt to create loose coupling, however, we are aware of areas where tight coupling may occur (this can be addressed in further phases). Our use of enumerations, specifically for class Item, allows for a single class instead of three. The game design as it stands is imperfect but will surely change and improve as we move forward. Currently, our design is the best way we could think of doing it for efficiency and ease of use.

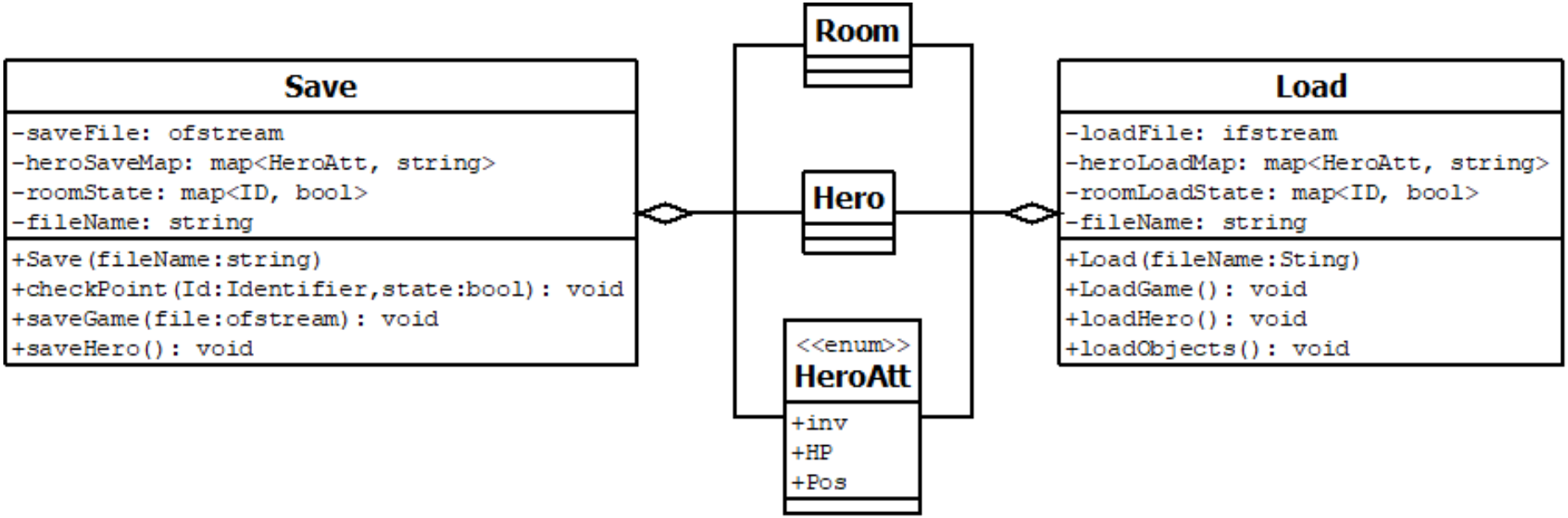
APPENDICES

APPENDIX A: FIGURES AND TABLES






Object
#description: string
#identifier: Identifier
+Object(Name:enum)
<<enum>>
Identifier
Room
-npcInRoom: list<Identifiers>
-objectsInRooms: list<Identifiers>
-adjoiningRooms: map<Directions, pair<bool, pair<int,int>>>
-hasEnemy: bool
-changedObjects: list<Identifiers>
+Room(row:int,col:int)
+getHasEnemy(): bool
RoomObjects
+state: bool
+RoomObject(ID:Identifier,state:bool=false)
+getState(): bool
<<abstract>>
Person
#position: pair<int, int>
#health: int
#damageValue: const int
+Person(identifier:enum,position:pair<int,
int>=0,0,health:int=10,dmg:int)
+attack(Person): int
Item
-itemValue: int
-type: itemType
+Item(Identifier)
+getModifier(): int
+getItemType(): itemType
<<enum>>
Directions
+north
+south
+east
+west
Chest
-contents: Item
+Chest(ID:Identifier,contents:Item)
+getContents(): Item
Lock
-lockValue: int
+Lock(ID:Identifier)
+unlock(): void
+checkKey(key:Item): bool
+changeLock(): void
Switch
+Switch(state:bool)
+setSwitch(): void
<<enum>>
itemType
+Keys
+Potion
+Weapons
LockedChest
-hasLock: bool
+LockedChest(ID:Identifier,state:bool)
+changeDescription(): void
+open(): Item
Hero
-inventory: map<Item, int>
-weaponOfChoice: Weapon
+Hero(player:Identifier=HeroName,pos:pair<int,
int>=0,0,health:int,damage:const int=1)
+inspect(Object): string
+help(): void
+usePotion(Item): void
+useKey(key:Item,Lock): bool
+getInventory(): void
+addInventory(itemToAdd:Item): void
+move(Direction): void
+Interact(RoomObjects): void
+Talk(Villager): void
-getWeapon(): void
Enemy
+Enemy(name:Identifier)
Villager
+Villager(name:Identifier,health:int=100,
damageValue:int=100)


Save
-saveFile: ofstream
-heroSaveMap: map<HeroAtt, string>
-roomState: map<ID, bool>
-fileName: string
+Save(fileName:string)
+checkPoint(Id:Identifier,state:bool): void
+saveGame(file:ofstream): void
+saveHero(): void
Room
Hero
<<enum>>
HeroAtt
+inv
+HP
+Pos
Load
-loadFile: ifstream
-heroLoadMap: map<HeroAtt, string>
-roomLoadState: map<ID, bool>
-fileName: string
+Load(fileName:Sting)
+LoadGame(): void
+loadHero(): void
+loadObjects(): void