DEPARTMENT OF COMPUTER SCIENCE

THE UNIVERSITY OF YORK

SEPR - Assessment 2

Updated Requirements Document

Evil Geese

Contents

1. Requirements Justification

1.1 Requirements Elicitation Process (Updated - entirely new)

We have continued to use an agile methodology, which informed our method for updating and generating requirements for this assessment, using frequent iterations and adaptations of our elicitation process. We continue to use the Agile Negotiation Framework, by employing inquiry, acknowledgement and advocacy as tools to shape our meetings and ensure a favourable outcome [1]. The specification that we made for assessment 1 followed the IEEE [2] and VOLERE [3] hybrid requirements template which used Requirement IDs that have proven extremely useful, as they allow clear traceability and allow easy discussion with the stakeholder.

We began the elicitation process for assessment 2 by analysing the existing scenario as a group and applying the current requirements to this to see if their fit criterion matched the current outlined model of the game and then brainstorming the necessary parts of the game due for this assessment. Inspired by our assessment 1 feedback, we have devised a new set of elicitation strategies that were found through consulting appropriate literature and through the development of the game. These strategies included prototyping; writing definitions of terms to clarify group and stakeholder understanding; modelling rationale and assumptions; discovering new scenarios/use cases through implementation and discussion; and analysing priorities, such as time pressures - this arose as a particular risk as mentioned in the risk update section - this lead to certain requirements needing to be removed.

Prototyping was initially done using pen and paper, sketching out components of our game and what would be needed in software to implement them. This served the dual purpose of setting out a basis for the software architecture and eliciting requirements. As the architecture began to develop we were able to prototype more with software for example character movement was modelled with simple objects that were later edited into characters. This process lead to redundancies in our statement of requirements and so we started to update them by refining and editing our previous requirements, regularly working through each item of the table as a group. For each requirement we decided whether it needed to be implemented according to the specification given by the stakeholder in writing, as well as information gained from interviews with them. We were also careful to monitor the effect of changing risks on requirements. These editing and elicitation stages were iterated until an System Requirements Specification (SRS) was created with requirements that were achievable in the given time frame for this task and prioritises the minimum specification of the assessment as well as the stakeholder's aspirations of the game as determined by discussion with them. We then checked the final, reformed SRS with the stakeholder to ensure that everything was correctly specified.

1.2 Requirements Specification and Presentation (Largely Unchanged)

We aimed to make an SRS that had the following characteristics: correct; unambiguous; complete; consistent; ranked for importance; verifiable; modifiable and traceable [2].
We have used an IEEE [2] and VOLERE [3] hybrid requirements template. Our specification for functional requirements is appropriated from the former and the specification for non-functional requirements from the latter. The simplistic specification for functional requirements allows us to limit the size of documentation, a goal of agile development [4]. We believe the explicit categorisation of non-functional requirements will aid us in risk management, more clearly distinguishing the types of requirements with which we are working. Our requirements are split into the following categories:
Functional (F); Usability and Humanity (UH); Performance (P); Operational and Environmental (EO); Maintainability and Support (MS); Security (S); Cultural and Political (CP); Legal (L).

We have included a fit criterion for each requirement which can be used itself as both a measure of a requirement being met and as a test case. We are aware of the inherent ambiguity of natural language and as

such believe it is vital to use consistent language within our SRS, ensuring that both we and the stakeholder are aware of the significance of the terminology used. Standard terminology is used to ensure the SRS is internally consistent [2]. We have included an ID alongside each requirement to ensure they are forward traceable [2]. Traceability is key to being able to document changes made to requirements, complementing our agile methodology.

2. Software Requirements Specification (Updated)

The requirement IDs have been colour coded to indicate changes/edits - at the bottom of the table there is a list of the requirements that have been deleted and a quick justification of the reason why is has been removed - this is for traceability as they may need to be put back in for the next assessment.
Justification for these edits are found in the update document.
Requirement ID Key:
Blue = new requirement; Yellow = edited requirement; Blank/clear = unchanged requirement.
Status for Assessment 2 Key:
Red = not yet implemented; Green = has been implemented; Orange = partially implemented.
Table 1: Software Requirements Specification

Requirement ID	**Status**	**Description**	**Fit Criterion**	**Environmental Assumptions, Risks or Alternatives**
F1		Game has 6 complementary characters.	Player selects 3 characters to play as during combat mode.	Assume characters are distinctive enough to warrant 6 of them. Risk that players will not find their differences distinctive enough and only stick to the same 3. Alternately, we could mandate the player plays as specific characters for specific parts of the game.
F2		Game world has 10 seperate locations.	10 distinct locations are included in the game, reflecting real locations on the University of York campuses.	Assume player understands relevance of locations being based off of real locations.
F3		Game must have multiple save slots.	Players should be able to create up to 6 save slots, each of which can be loaded or deleted.	Assume player requires no more than 6 slots. Risk that more than 6 are required on a single machine. Alternative is to allow unlimited save slots but this could negatively impact performance.
F4		Multiple assets can be collected in town mode.	Multiple assets can be found randomly through the town mode and are randomly dropped by enemies when defeated.	Assume assets are worth collecting. Risk that asset rewards are unbalanced and can be taken advantage of.
F5		Player can save progress at any appropriate time.	Players can save their current progress and location at anytime in the game except during combat or when the game is loading or during dialogue.	Assume players remember to save when they want to, risk that they forget. Alternatively, progress could be automatically saved after reaching certain milestones.
F6		Characters improve after critical story	Characters improve by gaining new abilities, more health, and having disadvantages removed	Assume new abilities are useful for combat mode. Risk that characters become too overpowered for combat and

		points.	after critical story points e.g. major combat events.	make gameplay unrewarding. Alternative, switch to experience point system instead.
F7		A player's' progress can be viewed from a top-down world map.	The player's progress through the game can be viewed on a world map which displays all game locations.	Assume each player has same definition of 'progress'. Risk that all forms of progress may not be able to be visualised simultaneously.
F8		Enemies can be battled in combat mode.	Game characters can battle and defeat enemy monsters in combat mode.	Risk that defeating enemies isn't a strong enough incentive for player to warrant their attention.
F9		Assets can be spent in town mode.	Assets collected by player can be spent in town mode in trade for ends valuable to the player.	Risk that trades on offer do not offer value to the player.
F10		Have a well-defined game end.	Once the Vice-Chancellor is defeated, after a brief cutscene to finish the plot, the credits will roll, the game will exit and the player will be brought to the main menu.	Assume game end is reachable and player wants to reach it. Risk that player doesn't see the game through to its conclusion.
F11		Game characters must have their own backstory.	When selecting a new character, an under 1 minute long cutscene will play, introducing the characters backstory and relevance to the game world.	Assume that player wants to play as an existing character.
F12		A demo mode can be started from the main menu.	A five minute long playable demo which allows a player to view all three play modes; the world map, combat mode and town mode. It can be replayed as often as desired by the player.	Assume that all three modes can be shown in restricted time. Risk that demo will not be representative of full game.
F13		A mini-game can be entered within the main game.	A mini-game of a different style, yet still relevant to the main game, can be entered from the world map. In the first instance, it will be entered as part of the story, appearing as an option to be entered via the world map from then onwards.	Assume that mini-game will add to game enjoyment. Risk that it breaks up core gameplay, resulting in a more disjoint gameplay experience.
F14		Game is displayed in orthogonal 2D	The game primarily uses a top down orthogonal view of a 2D game world.	Assume that other game systems will work well from an orthogonal 2D viewpoint. Isometric 2D was an alternative which was discussed and dismissed.
F15		Game begins with 3 characters.	The game begins with 3 characters to make combat possible, when new characters are encountered during the game they become playable in combat	Assume player wants to enter combat with 3 characters. Risk that introducing 3 characters at once is unengaging/not distinctive enough to make player want to use

			mode.	them. Alternative, stagger the introduction of characters and delay initial combat.
F16		Only one character is visible in town mode.	Game uses one of the 6 character sprites to indicate movement around town mode.	Assume that all sprites interact with town mode in the same way and do not offer an advantages or disadvantages over other choice of sprite. Risk that player may be confused about potential playable characters in combat mode if only one is visible in town mode. Alternative, show all 6, or represent party with alternative icon.
UH1		Game is accessible for those with limited mobility.	Game uses a fully rebindable control scheme.	Assume that player knows which control scheme works best for them. Risk that too much control choice could confuse player. Alternative is to have several default control schemes defined by us.
UH2		Game is accessible for those with impaired vision.	Ability to change game zoom and change to red/green colour-blind friendly mode.	Assume visually impaired players can navigate to where game can be visually changed. Could configure settings in walkthrough at start of game.
P1		The game must run on specified minimum hardware.	Game must run on the desktop PCs in the Computer Science departments' software labs at 30 FPS 99% of the time.	Assume that game will run consistently on the same hardware and that hardware will remain static. Risk that future hardware change will reduce performance of game.
OE1		Game is appropriate to be demonstrated on open days to both prospective students and their guardians.	The game contains no sexual content or inappropriate language or imagery, it should be PG-13. The game must portray the University in a generally positive light.	Assume that we can determine what would be considered inappropriate. Risk that we are too sensitive and impact on enjoyment of game. Discussion with potential end players may be helpful during development. Discussion with stakeholders will be required to ensure the portrayal of the University is sensitive to their wishes.
MS1		Game shall run on Windows 10.	The game can be run on Windows 10 machines inside and outside of the software laboratory.	Assume that Windows 10 will continue to run game after project delivery. Risk that updated Windows OS can no longer run game.
S1		Game shall not require a server.	The game stores all data locally, not requiring internet connectivity.	Assume that data files stored will not be accessed by player from outside the game. Risk that player corrupts local data.
CP1		Game shall use British English language.	The game uses British English language for all audio, visuals and text.	Assume that player will understand British English. Risk that game is used by audience who do not understand British English.
L1		Game can be resold, with all	All assets and game plug-ins used within the game are able to	Assume that online publisher of artwork is owner. Risk that this is not the case

		its components being commercially resellable.	be used commercially and are modifiable if required.	and there is ambiguity as to who is the owner. Alternative is creating all artwork in-house to avoid risk.

References

[1] E. E. Adam Weisbart, “Agile Negotiation: The Agilist’s Guide to Persuasion and Influence,” August 2013. [Online]. Available: https://www.agilealliance.org/resources/sessions/agile-negotiation-the-agilists-guide-to-persuasion-and-influence/ [Accessed: 25 Oct.2017].

[2] I. C. S. S. E. S. Committee and I.-S. S. Board, “IEEE Recommended Practice for Software Requirements Specifications,” Institute of Electrical and Electronics Engineers, 1998.

[3] S. Robertson and J. Robertson, Mastering the Requirements Process: Getting Requirements Right. Addison-wesley, 2006.

[4] I. Sommerville, Software Engineering. Pearson, tenth ed., 2016.