Requirements

The first step in the requirements engineering process was to elicit our requirements. We needed to identify how to gather these requirements using different methods such as stakeholder negotiation and user scenarios. It was also important for us to record the elicitation phase in detail so that we can form good rationale for each requirement [1, p.87]. Below are the processes we went through during the elicitation phase.

Elicitation Phase

Assessing the Brief

We began by using our first team meeting to address ambiguities in the brief. From there we formed a list of questions that addressed these uncertainties, which were asked in our initial client meeting. Using the answers provided by the client we formed a list of initial requirements, which we used to form the basis for the rest of the elicitation process.

User Survey

To be thorough with the quality of the requirements gathered we designed a survey
containing questions based on these requirements. This was then distributed among one of our stakeholders, our cohort. We received 25 responses out of the 49 people in our cohort which is an acceptable response rate for an internal survey of this kind, the responses can be found in reference [2]. The answers to the survey raised some conflicts between what the client wanted and what the users wanted from the game. These conflicts were noted [2] so that they could be presented to the client during validation.

Validation

We prepared a list of conflicts gathered from our user survey and first client interview. Finding suitable compromises to these conflicts through negotiation with the client was critical to ensure that all stakeholders in the project were satisfied with the final set of requirements. The second meeting with the client gave us the opportunity to validate all of our requirements and resolve conflicts so that we could produce a final set of agreed upon, unambiguous, requirements. Additionally, we presented the client with a prototype of the map design and game mechanics; so that we could verify that we were in line with what he wanted.

Requirements Specification and Presentation

Our requirements table's design and standard was based on the IEEE specification [3], however we modified it to include reference numbers to make it easier to refer to individual requirements throughout the project. Research was done into the different types of requirements that could arise in a project like this [3, pp.16-17]. Following this research, we decided to include functional, non-functional, constraint and performance requirements into our table to cover all the types' of requirements we had gathered during our elicitation process.

Requirements were mainly derived using three methods: primarily interviewing the client [4], user survey [2] and finally from using use case scenarios [5]. The use case scenarios allowed us to model how the user would interact with the system. This identified features that the system should include which led to the derivation of requirement F15, something that could have otherwise been missed out if we had not formed Scenarios.

Table Key

C = Constraint Requirement	F = Functional Requirement
NF = Non-Functional Requirement	P = Performance Requirement

Software Requirements Specification Table		
ID Number	**Requirement**	**Rationale, Assumptions, Associated risks and Alternatives**
C1	System must be able to run on a computer in the Computer Science department.	The game should at least be able to run on an operating system that the computers in the department have, (Windows or Linux). Additionally, we should take into account devices with lower specs being able to run the game as well.
C2	The system should appeal to our SEPR cohort and prospective university students.	Based on the University of York Communication Officer's statement the game will be used at open days to advertise to prospective students. Hence, we need to target the game at an age range of 17 – 20 year olds. A risk here is that we may alienate other demographics such as older users, (i.e. parents).
C3	The game shall be able to take input from both the mouse and the keyboard.	Team names will need to be entered using the keyboard and the mouse should be used to navigate and control the game. There is a potential risk to make the controls too complex here which would make the game difficult for new players.
F1	Game time should be between 10 and 30 minutes.	Our user questionnaire [2] suggested a game time of approximately 1 hour, however our client requires that the game should typically last 10 to 30 minutes [4]. It will also reduce the risk of a player becoming bored part way through a game, as it is shorter.
F2	The system should have a turn timer that can be toggled on/off in the game setup/options menu.	Based on our user questionnaire [2] and validation in the client meeting [4] the player should have the option at the start of a game to enable or disable a turn timer.
F3	The turn timer, if enabled, should pause while the mini-game is being played.	If the Vice-Chancellor appears then the turn timer should be paused for the duration of the game as the current player would be unfairly disadvantaged if they missed out on part of their turn time.
F4	Turn time limit should range between 30 and 120 seconds.	Based on our user survey [2] the median group of users felt a turn should last between these times.
F5	A mechanism is required to resolve conflicts, i.e. Team A is attacking a sector held by Team B.	Relating to requirement F6 this mechanism will contain an element of RNG but will have an element of skill involved as well. The amount of skill required should be based on the relative strength of armies in combat. Meaning that if the attacker is significantly more powerful than the defender then it will not take much skill to defeat the opponent.
F6	The battle mechanic requires an element of skill.	User feedback suggested that the combat mechanic should contain an element of skill [2]. We must consider

		play testing a mechanic like this to make sure it keeps the game balanced and fun.
F7	When a player conquers a sector there is the possibility that the Vice-Chancellor may appear, triggering a mini-game.	The Vice-Chancellor mini-game is triggered upon conquering a sector he is hidden on. If it was too frequent, we risk the mini-game becoming tedious and annoying so a possible spawn rate for the Chancellor should be added.
F8	The Vice-Chancellor mini-game should last approximately 30 seconds.	Our client specified in an interview [4] that the mini-game should last approximately 30 seconds. If the game lasted much longer it would risk becoming boring and other players waiting for their turn could become bored.
F9	No bonus should be awarded to the player if they fail the Pro-Vice Chancellor mini-game.	Due to ambiguity in the brief we validated this requirement with the client [4].
F10	A player limit of 2-4 players and in games of 3 or 4 players there may also be a neutral player.	This was a decision we came to based on the data gathered from the user survey and client interview [2, 4]. A risk here is that map size is fixed. Meaning if 4-players decide to have a game then it could get crowded if the map size is too small.
F11	A third neutral AI player must be present in a 2-player game.	This AI can only defend and will never receive reinforcements or move. As validated by the client [4], the purpose of this AI is to also create a layer of unpredictability in the game.
F12	The system should include the ability to save at least one game and be able to reload it at a later time.	Mentioned in the brief this is essential due to how long a game could possibly go on for.
F13	The system should contain a GUI based on the university campus map, subdivided into sectors.	The map should be recognisable as the University of York campus; however we have the liberty to modify it to a degree in order to improve gameplay. There is a risk that if we try to make the map too much like reality we will not be able to make it balanced.
F14	A bonus mechanism should be included for holding sectors at the end of the player's turn.	For each sector that the player captures, at the end of their turn, they will be given some bonus. This bonus could be troops or in-game currency. If the bonuses are poorly balanced then we risk it being impossible for a losing player to make a comeback as one player could become unfairly powerful.
F15	Before the start of every game the user should be prompted with an intermediate setup/options menu.	This ensures the user can alter the system environment in a controlled way before games start. This eliminates the frustration of having to alter game settings mid game. We also found that through user scenarios [5] this is a mechanism the game needs for the user to transition smoothly into a game.
F16	A mechanism for calculating how many new gang members each gang receives in	Each sector has a value for the amount of reinforcement troops it provides at the end of a player's turn if they own it. The player can then allocate these troops to any sector they are currently holding.

	each turn; new gang members should be allocated to held sectors.	
F17	At the start of the game, all sectors are unclaimed. Each sector should be allocated (by some random mechanism) to a gang.	The allocation mechanism needs to be tested so the gameplay is balanced. i.e. Players don't receive too many high value sectors. The risk here is that a completely random mechanism could create disadvantages for some players and advantages for others.
NF1	The game must be easy for new players to pick up.	This must be considered as the game will be used at University open days and UCAS days. A complicated game will just frustrate users who may only have a couple of minutes to try it out before moving on. But, there is a risk of making it too simple. Turn based strategy games are known for being complicated and we don't want to drive away experienced players of the genre.
NF2	The game should be stylised using a hybrid between realistic graphics and cartoons.	This style was chosen based on user feedback from the survey [2]. A risk here is that producing all of the graphical resources could consume a lot of time.
NF3	The game should be suitable to use in advertising situations.	The product must be suitable for the University to use in advertisements, such as at open days and UCAS days. Regular meetings with the client should ensure that the product we are developing is suitable for the University to use in this way.
NF4	The game should have a soundtrack including background music and sound effects.	Relating to requirement C2 and NF3 the game needs to engage the player and capture the users interest respectively. By using a soundtrack we can immerse the user more. There is a risk that the sound production process may take more time and resources than we can allocate.
NF5	The game should have accessibility features for disabled users	A risk here is that implementing such features may consume a lot of development time.
P1	The game must run smoothly. i.e., Should not crash or lag.	If an issue is found with performance on lower spec devices we can optimise the game.

References

[1] I.S. Sommerville and P.S. Saywer. Requirements Engineering, A Good Practice Guide. https://www.scribd.com/document/337626753/Sommerville-Ian-Sawyer-Pete-Requirements-Engineering-A-Good-Practice-Guide [Accessed: Oct. 31, 2017].

[2] SEPR “Survey Results Analysis” Risky Developments [Online]. Available: http://riskydevelopments.co.uk/documents/UserSurveyResults.pdf [Accessed: Nov. 3 2017].

[3] "IEEE Xplore Document. IEEE Recommended Practice for Software Requirements Specifications". [Online]. Available: http://ieeexplore.ieee.org/document/392555/ [Accessed: Oct. 31, 2017].

[4] SEPR “Client Interview Records” Risky Developments [Online]. Available: http://www.riskydevelopments.co.uk/documents/ClientInteviewRecords.pdf [Accessed: Nov. 3 2017].

[5] SEPR “Use Cases” Risky Developments [Online]. Available: http://riskydevelopments.co.uk/documents/UserScenarios.pdf [Accessed: Nov. 3 2017].