Requirements
Eng1 Group 29

Adam Hewlett
Ani Thomas
Dan Kirkpatrick
Dominik Hagowski
Matthew Crompton
Niko Chen

Introduction

Prior to the development and implementation of code, our team set out project requirements as this would:

-aid in the development process.
-help to keep the project on track with stakeholders' goals
-prevent the team from working on any unnecessary features

This was done at the start of the project to avoid common mistakes that had been outlined through research, such as eliciting unbounded and vague requirements, and the common tendency to make assumptions. [1]

The requirements for this project are broken down into 3 categories based on the research our team did: User, Functional, and Non-Functional requirements. [2]

Throughout the process of identifying and formatting requirements, the definition of a well-defined requirement was focused on: A well-defined requirement is a statement of system functionality that must have proof of its validation and must be met to achieve a customer's objective for a system. [3]

How were requirements elicited and negotiated

The requirements for the project were elicited and negotiated through analysing the product brief and a meeting with the client, where we asked for clarifications for certain points that would be needed before outlining requirements. After this, a brainstorming session took place using the notes from the interview as well as the project brief. We also looked at the requirements of similar projects and games such as 'Overcooked' in order to get a better understanding of the requirements to create a successful game. [4]

With this information, a Single Statement of Need (SSON) was created alongside a final list of User Requirements for the project. Our SSON is as follows:

SSON

The system should enable the consumer to play a single-player game that requires managing the staff around a kitchen, who will be preparing various dishes requested by customers coming into the Piazza Restaurant, that the user will have to prepare in a certain time limit, otherwise they will lose reputation.

The SSON acted as a guide for maintaining focus on the system requirements that would be used in the Architecture plan

Why requirements are presented as they are

To format the elicited set of requirements, the 'IEEE Guide for Developing System Requirements Specification' provided insight into a sensible format and the necessary level of detail for the requirements.
To distinguish types of requirements, colours were used to highlight the differences, and make the reflection process easier. The standard tabular approach was decided upon as it is the easiest to read and reread.

User Requirements

Requirement ID	Description	Risks and Assumptions	Priority Level
UR_PLAYABLE	The game should be playable and enjoyable for the target audience	May result in the game being considered boring in a effort to please everyone	Shall
UR_DISHES	The game should contain a variety of dishes to prepare	May lead to certain dishes being harder to prepare then others, so some order harder	Shall
UR_COOK	Cook should be controllable by the user to kitchen	May not know which cook they are controlling	Shall
UR_MIN_COOK	There should be at least 2 cooks in any scenario	Scenarios may be to hard with only one cook	Shall
UR_CONTROL	Controls should follow standard conventions or be explained and easy to use	User may not check the controls and blindly follow standard convention	May
UR_DIFFUCULTY	Game should have different difficulty scenarios	Game may end up being to difficult, leading to user quitting	Shall
UR_TIME	Each scenario should not take to long	Game may become boring and repetitive if scenarios take too long	Shall
UR_CONVENTIONS	The game should follow standard conventions	The conventions could be followed where it doesn't make sense in the game	May
UR_SUCCEED	The scenario is passed if all customers are served	The user may not know the requirements to pass the scenario	Shall

Systems Requirements - Functional

Requirement ID	Description	Risks and Assumptions	Design Requirement
FR_Prepartion_Steps	Each should have a number of steps to prepare it E.g cutting, cooking ect.	Some recipes may be more complex than others making it harder to complete in the same timeframe	UR_DISHES
FR_Ingredient_Stations	Ingredients should be able to be gotten by the cook from ingredient stations e.g salad basket.	Stations must be easily distinguishable so that they do not get mixed up.	UR_DISHES UR_COOK

Requirement ID	Description	Risks and Assumptions	Design Requirement
FR_PREP_STATION	Stations for food preparation should be available for the cook to interact with	User may not know what each station is for	UR_DISHES UR_COOK
FR_Cook_Move	Player should be able to move cook around	User must know the control scheme in order to do this	UR_COOK UR_CONTROL
FR_Cook_Interact	Player should be able to make the cook interact with objects and tasks	User may not know what to interact with	UR_COOK
FR_UI	There should be a UI displaying the recipes that need to be completed, and time left	UI may clutter the screen if too large and obstruct the game.	UR_PLAYABLITY UR_CONVENTIONS
FR_CUSTOMER	Customers should arrive asking for different dishes	Orders from the customers may not be clear for the user	UR_DIFFCULTY
FR_RANDOM	Dishes that the customer ask for should be random	Dishes may be all the same by chance making the game boring	UR_DIFFUCLTY
FR_TITLE_SCEEN	The game should be able to launch into a title screen	The player may not know where to navigate to from the title screen	UR_CONVENTIONS
FR_AREA_BOUNDARIES	The user should only be able to move the cook certain places and collide with counter objects	Make sure the boundaries for the game are clear	UR_CONVETNTIONS
FR_ANIMATIONS	There should be animations in the game, such as cutting up lettuce or walking	Assume that the hardware is capable of playing the game along with the animations	UR_PLAYABILITY
FR_MUSIC	The game should have background music to make it more enjoyable	Background music may distract the player	UR_CONVENTIONS
FR_SERVE	The dishes should be able to be severed to the customers by placing them on a counter	User may not understand how to serve the food	UR_SUCCEED

System Requirements - Non-Functional

Requirement ID	Description	User Requirement	Fit Criteria	Risks and Assumptions
NFR_RESPONSIVE	The system should respond quickly to user	UR_PLAYABLE	The response should be no	Assumes the user gives a

Requirement ID	Description	User Requirement	Fit Criteria	Risks and Assumptions
	input from the keyboard		slower then 0.5s after the button press	valid input
NFR_INFORMATION_TIME	The user should be informed of any changes quickly	UR_PLAYABLE	UI changes should take no longer then 0.5s	Player may not notice the change
NFR_ANIMATION	Animations should run smoothly	UR_PLAYABLE	Game should run at min 30FPS	Assumes hardware can run at this speed
NFR_GRAHICS	Graphics should look like the objects that they represent	UR_PLAYABLE	Player should be able to identify all assets	Player may not recognise the object

Constraints

Requirement ID	Description
CON_APPROPRIATE	The game should not contain explicit or graphic content, such as blood or gore, and should be suitable for any user in the target audience.
CON_LANGUAGE	The game must be programmed in Java.
CON_RUN	The game should be able to run on any University of York Computer Science Department computer
CON_ACCESSIBLE	The game must use colours with high contrast to ensure all users can play it

References

[1]'IEEE Guide for Developing System Requirements Specifications', IEEE Std 1233, 1998 Edition, pp. 14, Dec. 1998.

[2] D.Thakur, (2013, November.23), What is Software Requirement? Types of Requirement.

[3]'IEEE Guide for Developing System Requirements Specifications', IEEE Std 1233, 1998 Edition, pp. 11, Dec. 1998.

[4]'IEEE Guide for Developing System Requirements Specifications', IEEE Std 1233, 1998 Edition, pp. 16, Dec. 1998.