The Design and Development of a Prototype Community Banking Game

Matthew Long, Bradley Steel, Mary Martin
Department of Computer Science and Information Technology, La Trobe University
Bendigo, Victoria, Australia

Phineas Istratoaie, Alex Duncan
Department of Urban, Rural and Environmental Planning
La Trobe University
Bendigo, Victoria, Australia

***Abstract*— A prototype game simulating the Bendigo Community Bank ® model was designed and developed for a client at a large regional Australian bank, Bendigo and Adelaide Bank. The purpose of the prototype was to demonstrate the value of using the game as a learning tool for the bank's staff, to assist them in understanding the concepts of community banking. If successful, the Bank intended to prepare the game for product development and distribution. Because the timeframe for the project was only three months, the decisions to adopt the Agile Project Management and Agile Programming methodologies, to design and develop the prototype game were critical. Also key to the project's success was the correct mix and selection of the team for the project that required programming, graphics and town planning expertise. The goal of a functioning prototype was achieved within the relatively short timeframe. The prototype met all of the client's project requirements and in some cases exceeded expectations.**

Keywords—game programming; game learning theory; agile project management; agile programming; project communication

I. Introduction

The 'Prosperous Village' community banking game was a collaborative project between La Trobe University and Bendigo and Adelaide Bank Pty Ltd (BABL). Originally proposed by the Head of Environment & Sustainability at BABL, who became the project's client, the project was to develop a prototype of an educational game for BABL's training division. The game would simulate the behaviour of BABL's Community Bank ® model [1], and hence assist BABL's staff who played the game to understand the concept of community banking. The goal for the project was a functioning prototype that the client could present to the stakeholders at BABL for their approval to prepare the game for product development and distribution.

To carry out the design and development of the prototype, a four-person project team from La Trobe University at Bendigo was recruited. Two members were Information Technology (IT) students. They were to undertake the design and programming development activities for the game's graphical interface and operation. The other two members were Town Planning students. They were to create the content (scenarios and events) for the game as well as develop the graphics for the game's interface.

The game was to simulate the impacts of a community bank within a growing town by demonstrating the effects of the decisions made by the player (whose game role was community bank manager). The overall aim of the game was to make the town thrive and hence become a 'Prosperous Village'. This was achieved if the bank as a business, was prosperous, in terms of both wealth and customer satisfaction, but the community had to become prosperous first. For the community to become prosperous all five forms of capital (built, natural, social, human and financial) needed to thrive together. The game focused on increasing these capitals through the player making decisions to increase the community's prosperity.

This paper describes the design and development of a functioning prototype game intended for distribution as a proof of concept. The methods and approaches adopted, along with detail about decisions made in the project are also described and discussed.

II. Background

The Bendigo and Adelaide Bank is a large regional bank with over 540 branches across Australia, a large portion of which follow the Community Bank ® model. The model was developed specifically to provide banking services in areas where large banks had closed, and assist those communities by giving back to the community through support and funding solutions for local projects and issues [1].

To understand the Community Bank ® model, the interactions of the five forms of capital and their effects on the town need to be appreciated. Using a game as a means to understand a relatively complex concept is thought to be a very effective way of learning [2–5]. This was the rationale used by project's client, and Head of Environment & Sustainability for BABL, Trudy Ellery when originally proposing the project. It was intended that the game would be released onto BABL's 'youLearn' system for the staff to use as a training module. This would give 6,500 BABL staff access to the game as part of their training.

The game was also to be developed so that it could be played on the staff office computer or as an app on a mobile device. After the initial rollout to BABL staff, the client hoped to take the game to the public in the Apple App Store [6] and release it on behalf of BABL.

Prior to La Trobe University being approached, the client had investigated support and resourcing for the idea of the game internally within BABL. However, while the idea gained support in principle, it proved difficult to assemble a team to develop it, partly because it required a mix of expertise (game programming, graphics, town planning) not commonly found together – especially in a bank! La Trobe University selected the La Trobe students because of their knowledge and skill sets in these areas, as well as their ability to adapt and learn quickly.

III. REQUIREMENTS

The client had four main requirements:

1. The game had to have a graphical interface that was visually appealing and which included an 'aerial' view of the town so decisions made would be seen to have an immediate effect. It would also display the community's five capitals graphically on a *capital wheel* at all times. The *capital wheel* would represent the level of each of the five forms of capital present, and these levels would change based on how the player handled events.

   The five main capitals, to be depicted on the capital wheel, were:

   - Built - The physical buildings and structures within the town and how maintained.
   - Natural - The state of the environment.
   - Social - The connections between people and their associations.
   - Human - The people within the town and their productiveness.
   - Financial - The economy of the town.

2. The prototype needed to be created in a way that would allow the user to play on either a desktop/laptop computer or a touch screen device (mouse and keyboard or a touch-only interface).
3. The game had to allow non-programmers to add new content (scenarios and events). This would assist in replayability and enjoyment of the game.
4. The game had to be either time based, or score based. This decision would be made later by BABL's Learning & Development division and their requirements for training modules.

   Considerations beyond the prototype stage were:

   - A 'kids-version' of the game for school children to play by using a more simple language and basic scenarios.
   - Multiple players - Have a multiplayer feature with different users controlling different types of town leaders that have their own separate roles.
   - Using real-life stories of situations, BABL had dealt with in the past.
   - Integrating a scoreboard within a social media system to compete with friends. This would greatly increase the level of replayability for the game due to competition amongst players.

Although these features were not in the scope of the project, they were kept in mind by the team throughout the project.

IV. METHODS

A. Project Methodology

The project methodology chosen was an agile [7, 8]or iterative project delivery model. This approach is recommended where a close working relationship with client and continuous feedback is required [9, 10]. The product or component of the product is demonstrated for client feedback at each iteration. Iterations (or delivery cycles) were short: one or two weeks. One of the benefits of this approach was that changes to the project could be made quickly and with minimum waste of time and resources [11].

Table 1 below shows the planned project schedule for the project. The methods or approaches used within activities are described in the following sections.

Consultations were conducted with the client on a weekly basis for the first four weeks, fortnightly for the following six weeks (to allow enough time for development of specific game features), then back to weekly meetings for the remainder of the project. In addition, as the client intended for all BABL employees to play the game within work hours, and it needed to be easily accessible for every employee, BABL stakeholders and experts, who could suggest options for development, were also present at meetings in the development phase of the project (week six to ten).

TABLE I. PROJECT SCHEDULE

Activity	Week
Client Consultation: Introduction to the project requirements Game Technology: Decide programming platform	1
Client Consultation: Continue requirements discussions Game Interface Design: Main menu layout	2
Client Consultation: Interface design feedback Game Interface Design: Town layout & capital wheel	3
Client Consultation: Interface design feedback, reverse brief	4
Game Interface Design: Capital wheel, email event system Client Consultation: Interface design feedback	5, 6
Game Design: Scoring & saving game Game Development: Map editor Client Consultation: Client feedback	7, 8
Game Development: Newsletters, data entry, instructions Client Consultation: Client feedback	9, 10
Game Testing: (Alpha - Developer) Client Consultation: Client feedback	11
Game Testing: (Beta – Client, refining game balance) Client Consultation: Client feedback	12
Game prototype submission, Project presentation	13

B. *Game Interface Design Approach*

The design of the graphical interface of gameplay incorporated into the agile development project schedule was planned to be completed in the first half of the project (signed off by week six) before starting development of the game operation. It was essential if the project was to be completed on time, that design decisions and client approvals for display of the main menu, city statistics, the *capital wheel* and the town view was given on a weekly basis during this time.

For the town view and layout, the client's original specification referred to games like SimCity 1994 [12] or FarmVille 2009 [13] so an 'aerial' view of the town was required. With this bird's-eye view, a system was also required where the player could receive and respond to events. It was decided that the player would be notified of events via an email-style inbox system within the game.

The *capital wheel* was to display the five capitals in the form of a wheel with five segments, like a pie graph. Each capital's value was to be displayed as a percentage and shown in numbers and colours. BABL's graphic designers provided the capital wheel symbols representing each capital.

Physical representation of the capitals in more than just numbers and colours within the *capital wheel* was also planned. The town had to react to varying levels of capital determined by the player so they were able to witness the expansion or diminishment of their town. Each individual capital was intended to visually change prosperity levels within the town in both graphical representation and overall town statistics. For example, the buildings within the town would deteriorate based on *built* capital, and the environment would flourish based on *natural* capital.

Another design decision to be resolved at this time involved the requirement that the game should be able to be played on iPhone device. Because the screen resolutions of iPhone 5 and iPhone 4 differ, this meant that two different resolutions were to be considered. To achieve this, the screen layout was based on the width of the screen while the game was played.

C. *Programming Platform*

The programming language platform in which the game would be built was a critical decision made in the first week of the project. What was required was a platform that could be used to develop a Web application and be easily adaptable for a mobile device. The Java, JavaScript and HTML5 languages were all valid options for programming a Web application but not ideal for a mobile app.

A professional app developer who was consulted recommended C# using XNA [14]. The XNA Framework provided a managed-code class library that contained features targeted specifically at game development. In addition, XNA Game Studio included tools for adding graphic and audio content to the game. The benefit of using C# and XNA was that it was intended for games development. Ultimately Microsoft Visual Studio 2012 was selected in conjunction with Windows Phone 7 to develop the prototype as this allowed the game to be directly transferred to a mobile device once development was complete. Also, this was a free solution in comparison to developing directly for the App Store [6]. Adopting this platform meant the prototype did not run as a Web application, however using C# meant it could easily be ported to other platforms (Android, iPhone, and Web).

D. *Programming Methodology*

The method of programming used by the two IT students was an adaption of an agile software development method known as 'pair' or peer programming [15]. The more difficult code sections were worked on collaboratively, while simpler programming sections were worked on separately. For example, the IT students collaborated to produce the overall structure of the game, but discrete components or functions of the game were worked on separately before the partner conducted testing.

As this was a prototype, programming was limited to the essential elements of the game. Even so, the list of additional functions that could be implemented in the game was always kept in mind so that these features could be more easily added afterwards if there was time. Some of the features that would be left out were real life scenarios, scoreboard, social media, cars driving along the streets and visible people inside the town. It was intended though, that these features would be implemented in the final product, if BABL decided to develop it.

E. *Graphics and Content Approach*

The Planning students had major responsibility for developing the graphics and content for the prototype. Like the IT students this meant that sometimes they worked collaboratively, while other times they allocated their work separately according to their particular skill or knowledge set. For example, they worked together to create the events, but then separated so that one focused on developing the scenario and event content for the game, while the other was the sole contributor for the graphics using free software (Google Sketchup [16]).

F. *Project Team Communication*

It was important for both pairs of developers (the IT students and the Planning students) to synchronise their work. With the help of free software namely Dropbox [17], OneNote [18], Skype [19], TeamViewer [20] and Facebook [21] the pairs managed to work almost simultaneously by constantly communicating and sharing project files.

In addition, the project team met weekly to give a progress report to the client and to ensure the project was staying within scope. This meeting was actually a two-stage process. The students met prior to meeting with the client to prepare a combined update of the week's activities so the client was able to give feedback at on a single report. This streamlining enabled the project to progress more quickly than separate reporting. Because the client was seeing a more complete picture, it also allowed for new ideas about the way the game was played. This was intended from the start, but as the project continued this became increasingly critical to the success of the project.

At various points in the project, the client also arranged for BABL design and programming experts to attend meetings, so new ideas and the issues around implementing them into the game could be discussed. These included whether or not the ideas fell within scope, if their content could be created and maintained, and if the features and functionality it required could be developed within the given time constraint. Even though many of the ideas fell out of scope, they could still be allowed for in the overall game design. The BABL experts also provided feedback and contributed suggestions and ideas that could benefit the prototype game being developed.

V. Results

The prototype met all of the client's project requirements and in some cases exceeded expectations.

A. *Game Playing Interface and Operation*

The first project requirement called for a graphical interface displaying an 'aerial' view of the town. This was implemented and refined with zoom in/out ability - an expected feature for any game with this type of view. Fig. 1 shows the aerial view of the town, along with a zoom view in the main interface of the prototype game.

Fig. 2 shows one view of the game when played. The player has been presented with a screen showing the email-style *inbox* overlaying the aerial view of the town on a grid layout, and a *capital wheel* along with town statistics. The *inbox* could slide on and off the screen to reveal the town. As the game is played, the interface is updated in real time. The *inbox* holds in-game emails informing the player of what is going on in the town and issues that need attention. After a series of emails on a certain issue, the player (whose role was the town's bank manager) is given an opportunity to choose whether they would like to invest in a solution to aid the town for that issue. Each action by the player affects the town's various forms of capital and this is demonstrated through the fluctuating levels in the *capital wheel*.

Another feature implemented was an automatic move capability. The town could be automatically moved until a set destination was reached. There were two options for doing this: *snap*, which placed the screen on the destination and *move*, which moved everything at a requested speed until the destination was reached in a gliding motion. Fig. 3 shows a view after a move to a set destination. Other game interfaces developed included entry and exit menus to allow the player to choose game levels of difficulty, on entry and save multiple games on exit.

B. *Game Playing Input Action*

The second project requirement was that the game should be developed as both a mobile device app (touch input) and web interface (which would need the game to be completely rewritten) on a desktop computer. For the sake of simplicity and to make the process of converting from one platform to another easier, the game was developed so that it could be played by click input, but adapted for touch. The game was played during the development stage with a mouse, which simulated a 'touch'. For example, the user would click and drag with the mouse much as if a user would touch the screen of a touch-device and drag their finger across the screen.

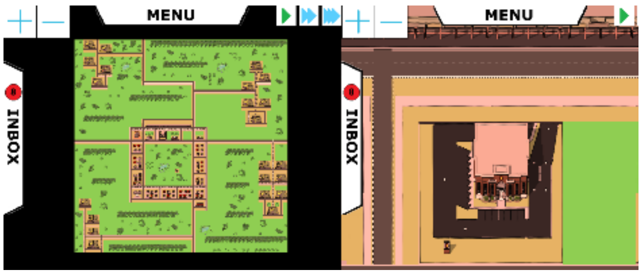
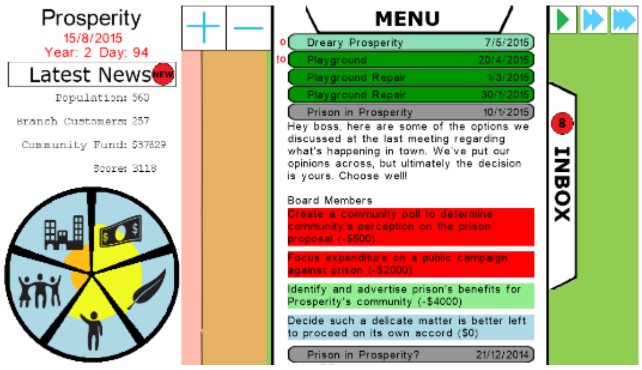


Fig. 1. Town in a grid format, along with zoom view on a building.



Dreary Prosperity 7/5/2015
Playground 20/4/2015
Playground Repair 1/3/2015
Playground Repair 30/1/2015
Prison in Prosperity 10/1/2015

Hey boss, here are some of the options we discussed at the last meeting regarding what's happening in town. We've put our opinions across, but ultimately the decision is yours. Choose well!

Board Members

Create a community poll to determine community's perception on the prison proposal (-$500)

Focus expenditure on a public campaign against prison (-$2000)

Identify and advertise prison's benefits for Prosperity's community (-$4000)

Decide such a delicate matter is better left to proceed on its own accord ($0)

Prison in Prosperity? 21/12/2014

Fig. 2. Screen with email-style inbox, town statistics, capital wheel.

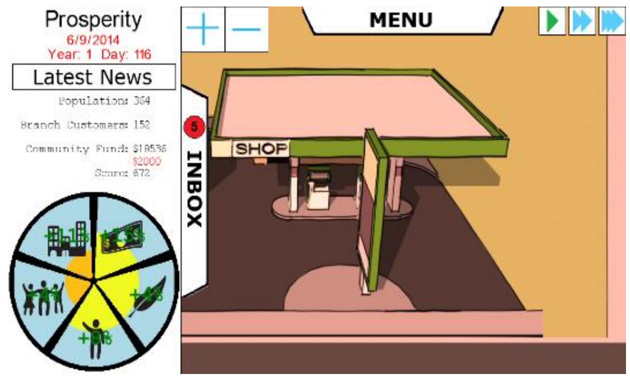


Fig. 3. Screen view after moving to a set destination.

C. *Data Entry Editor*

The third project requirement was that the game should have a strong and easily understood system for non-programmers to enter content. This was achieved by storing the content in csv files, i.e. a 'comma separated values' file, that would be read in by the game as an email. To produce the csv file, one option was to use a common spreadsheet application, Microsoft Excel to enter the data, however this would require data entry on a 77 x 81 grid, thus was not user friendly and error prone.

The option chosen was to develop a new utility program for entering the content within the game. It loaded the existing 'emails' file and allowed the content developers to change the way the emails were used inside the game via a graphical interface

Five main types of emails were developed: *Blank*, *Option*, *Built*, *Natural* and *Fund*. A *Blank* email was informative, an *Option* email was an event, *Built* and *Natural* generated a status of the town, and *Fund* generated a fund for the Market Development Fund. All email types had a unique ID, subject and chosen colour. The email could also refer to a placeholder variable PLAYER_NAME to allow the player's actual name to be substituted.

A *Blank* email (Fig. 4) could allocate cash and prompt for another email to follow in a certain number of days after it arrived. It could also change the capital.

The *Option* email (Fig. 5) was the most complex. An *Option* email could contain up to 10 options. Each option was valid for a pre-set number of days after which it could not be answered. When an option was answered, it sent out a specified email in reply, and the number of days before it would be activated.

As well as its content, each option had a cost associated with it. *Option* emails changed the capital inside the game based on the five figures shown along the bottom (positive, 0 or negative) of the screen.

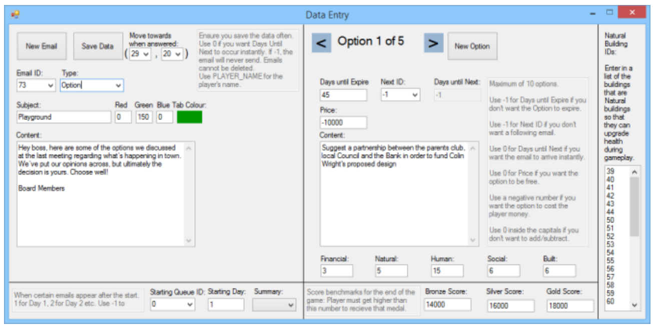
This data entry utility essentially allowed any user an easy-to-use method to continue the 'development' of the game in the form of constant addition of events.

D. *Other Game Playing Interface Features*

A player scoring system was implemented in the game to meet the fourth project requirement. At the bottom right of the screen, a 'Score Benchmark' was displayed. This allowed the player's score to be measured against the set 'medals' of the Score Benchmark. Benchmarks could be set within the data entry program.

Another feature was a dropdown list that appeared in the bottom left of the screen when the game was first loaded. It displayed all of the email IDs and formed a starting queue within the game according to the time set for their 'Starting Day' (the number of days after day one).

On the far right of the screen, the Building IDs of buildings that were to be seen by the game as *Natural*, otherwise buildings were classified as *Built*. In the game, *Natural* buildings were allowed a health upgrade when the *Natural* capital was high.


Data Entry

New Email | Save Data | Move towards when answered: (29 , 20)

Ensure you save the data often. Use 0 if you want Days Until Next to occur instantly. If -1, the email will never send. Emails cannot be deleted. Use PLAYER_NAME for the player's name.

Email ID: 73 | Type: Option

Subject: Playground | Red 0 | Green 150 | Blue 0 | Tab Colour: 🟩

Content:
Hey boss, here are some of the options we discussed at the last meeting regarding what's happening in town. We've put our opinions across, but ultimately the decision is yours. Choose well!

Board Members

< Option 1 of 5 > | New Option

Days until Expire: 45 | Next ID: -1 | Days until Next: -1

Price: -10000

Content:
Suggest a partnership between the parents club, local Council and the Bank in order to fund Colin Wright's proposed design

Maximum of 10 options.

Use -1 for Days until Expire if you don't want the Option to expire.

Use -1 for Next ID if you don't want a following email.

Use 0 for Days until Next if you want the email to arrive instantly.

Use 0 for Price if you want the option to be free.

Use a negative number if you want the option to cost the player money.

Use 0 inside the capitals if you don't want to add/subtract.

Financial: 3 | Natural: 5 | Human: 15 | Social: 6 | Built: 6

When certain emails appear after the start. 1 for Day 1, 2 for Day 2 etc. Use -1 to

Starting Queue ID: 0 | Starting Day: 1 | Summary:

Score benchmarks for the end of the game: Player must get higher than this number to recieve that medal.

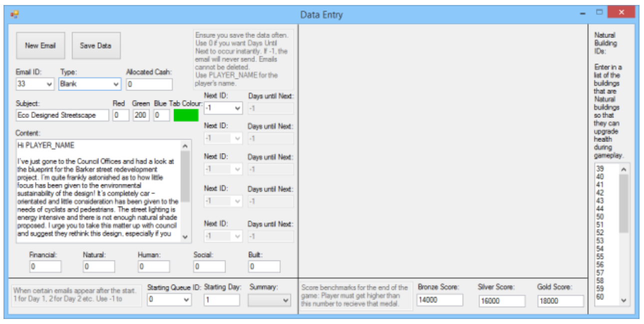
Bronze Score: 14000 | Silver Score: 16000 | Gold Score: 18000

Natural Building IDs:

Enter in a list of the buildings that are Natural buildings so that they can upgrade health during gameplay.

39 40 41 42 43 44 50 51 52 53 54 55 56 57 58 59 60

Fig. 4. Blank email.


Data Entry

New Email | Save Data

Ensure you save the data often. Use 0 if you want Days Until Next to occur instantly. If -1, the email will never send. Emails cannot be deleted. Use PLAYER_NAME for the player's name.

Email ID: 33 | Type: Blank | Allocated Cash: 0

Subject: Eco Designed Streetscape | Red 0 | Green 200 | Blue 0 | Tab Colour: 🟩

Next ID: -1 | Days until Next: -1
Next ID: -1 | Days until Next: -1
Next ID: -1 | Days until Next: -1
Next ID: -1 | Days until Next: -1
Next ID: -1 | Days until Next: -1

Content:
Hi PLAYER_NAME

I've just gone to the Council Offices and had a look at the blueprint for the Barker street redevelopment project. I'm quite frankly astonished as to how little focus has been given to the environmental sustainability of the design! It's completely car – orientated and little consideration has been given to the needs of cyclists and pedestrians. The street lighting is energy intensive and there is not enough natural shade proposed. I urge you to take this matter up with council and suggest they rethink this design, especially if you

Financial: 0 | Natural: 0 | Human: 0 | Social: 0 | Built: 0

When certain emails appear after the start. 1 for Day 1, 2 for Day 2 etc. Use -1 to

Starting Queue ID: 0 | Starting Day: 1 | Summary:

Score benchmarks for the end of the game: Player must get higher than this number to recieve that medal.

Bronze Score: 14000 | Silver Score: 16000 | Gold Score: 18000

Natural Building IDs:

Enter in a list of the buildings that are Natural buildings so that they can upgrade health during gameplay.

39 40 41 42 43 44 50 51 52 53 54 55 56 57 58 59 60

Fig. 5. Option email.

E. *Map Editor*

The Map Editor was a utility component of the game developed so a non-programmer, i.e. the graphic designer, could change the grid display of the town. Buildings were drawn on a grid to create 'blocks' (Fig. 6) which were stored in image files for the game to read in and place on the grid. Once these images were created, the Map Editor was used to 'draw' the towns.

The first size chosen when programming the grid dimensions (50 x 50 = 2,500 individual blocks) ultimately proved to be the ideal dimension. The graphic designer found the Map Editor an excellent way to manage all 2,500 buildings.

The Map Editor developed (see Fig. 7) for this prototype was modelled on an approach used in SimCity. The purpose was for the graphic designer to be able to view all imported graphics, click on a desired building block, and use it to 'paint' the town. Each block was assigned a unique ID. When a block within the grid was clicked, the ID displayed was changed to the currently selected building block's ID. If the mouse was held down, several buildings could be changed by dragging the mouse across a large area to set them all to trees, or blank grass ('painting'). The graphic designer could change the setting from 'Place Buildings' to 'Drag Screen', and then use the mouse to move around the town (and zoom out like normal). Right-clicking buildings increased their health until the maximum was reached (which was based on stored counter).

This utility simplified the drawing of the game considerably. The graphic designer was provided with three different major 'levels' of the town, and was able to increase the number of levels to nine simply by changing the health for each of the buildings to represent the town 'growth'.

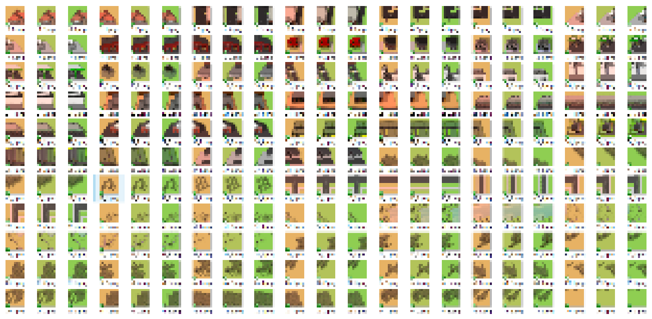


Fig. 6. The 'blocks' to create the town in Google Sketchup.

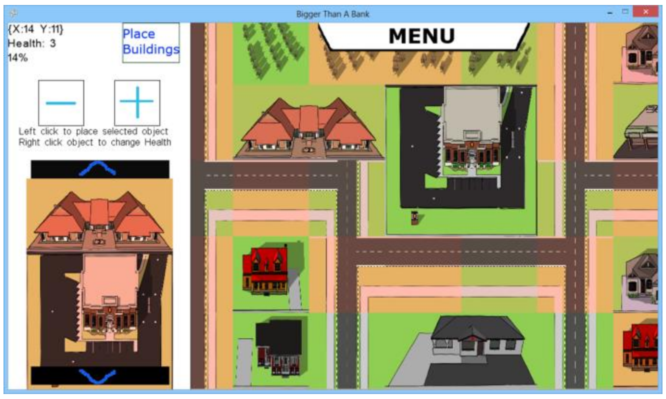

Bigger Than A Bank
{X:14 Y:11}
Health: 3
14%
Place Buildings
Left click to place selected object
Right click object to change Health
MENU

Fig. 7. The Map Editor showing information for a block screen.

VI. Discussion

The design and development of the prototype Community Banking game was to be accomplished in the very short time frame of three months (12 weeks). This meant that decisions made during the design and development phases were critical to the project's success. One misstep or poor decision could cause irretrievable delays. The following provides insight into the impact of the project and programming methodologies adopted, and how some of the game features came about.

A. *Project Methodology*

From the project outset, it was recognised that frequent client feedback at weekly or fortnightly intervals, along with close contact within and between the design and development team members was of paramount importance. This was necessary to meet the project goal in the required period. In contrast to the more traditional 'Waterfall' project methodology, rapid turnaround times and continuous contact between team members to implement features and obtain client approval are features of the agile methodology that was chosen [22]. Without this approach the obstacles that appeared in the design stage in the project would not have been resolved quickly enough. Fortunately, the client was available for timely consultation throughout the project.

The agile project methodology was also necessary because the client had not envisaged the way the game interface would operate. Therefore, it was vital that the client was able to approve each feature as it was implemented in the game. The email-style *inbox* system is a good example. When the idea of running the game content through an 'email' system was suggested it was initially dismissed by the client because the client was thought it would take away from the immersion of the game if the player had to keep opening their email client outside of the game. Once the client realised that the email system would be an in-game system that was ran entirely from within the program it was accepted.

B. *Programming Methodology*

The decision to adopt a 'pair' programming methodology was also a critical one. This approach maximised the IT students' available time for programming. Each nominated code sections they had the skill set for and mostly managed their actual coding individually. At the same time, they kept in close contact informing each other how each section worked. They consulted each other with various errors and troubleshooting throughout the programming stages. The major components were worked on together; including the save game component, and the system where each of the sections created individually would meet and communicate. When a major component had been implemented, both programmers collaboratively tested the function. The *inbox* was an example of this.

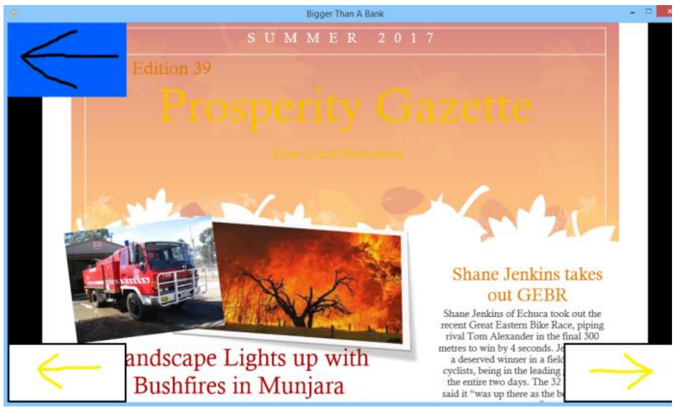
C. *Other Game Features*

Many of the game's features that were developed emerged from feedback meeting or testing activities.

One feature developed after one of the testing phases was the automated closing of the custom developed email-style *inbox* system. Testing the interface revealed that the majority of the gameplay was spent looking at the *inbox* and not on the large town that had been created. After much discussion, the team decided that the town view could be triggered when the player picked certain options within the emails. On a trigger option the *inbox* would close and the screen would pan over the town to the building or area that was affected by the decision of the player. Although this was not done for every email issue (as not all issues revolved around a location or building), it was made to occur as often as possible to show the player the town. An example is the 'Petrol station going out of business' scenario. The player would select an option from the email. The capital in the *capital wheel* would change accordingly, the *inbox* would close and the screen would pan over the city to the petrol station that the player either helped stay open or made another decision which forced them to close.

A feature from a feedback meeting proposal was the ability for the player to provide funding in the game. Originally, it was designed so the player could allocate whatever amount of money to an issue they deemed appropriate by using a slider bar interface. The purpose of allowing the player to influence an issue with a certain amount of money was an attempt to simulate just how a community bank helps the community. The slider bar would provide much more dynamic gameplay, the change in capital would be dependent on just how much money was allocated. For example if *Built* capital was very high and *Natural* capital was very low, the player could choose to allocate a large sum of money to an environmental issue, and a smaller sum of money to a building issue. Raising the *Natural* capital significantly while maintaining the *Built* capital without wasting money would be of much more value elsewhere. This feature was designed but not fully implemented in the prototype due to time constraints and its relatively low priority in the project.

Another dynamic feature proposed through feedback was the town newsletter (Fig. 8) that would be printed according to the decisions made on issues by players. This required a different article to be written for every option the player could choose for the issues written about in the half-yearly newsletter. Similarly, this feature was subject to the project's time constraints and low priority rating so was only partially implemented in the prototype to demonstrate the concept.



Shane Jenkins of Echuca took out the recent Great Eastern Bike Race, piping rival Tom Alexander in the final 300 metres to win by 4 seconds. Je a deserved winner in a fiel cyclists, being in the leading the entire two days. The 32 said it "was up there as the b

Fig. 8. Example newsletter.

VII. CONCLUSION

This paper described the design and development of a prototype game in the relatively short time of three months. It met all of the client's requirements and in some cases exceeded expectations. At the final presentation of the project and demonstration of the fully functional prototype, the client confirmed this view by declaring, "The outcome is fantastic, they achieved it completely."

The Data Entry Editor developed for non-programmers to use, allowed the game to be added to with ease to continue the growth of the content, which would assist in replayability and enjoyment.

Similarly, the development of the Map Editor utility allowed non-programmers (i.e. the graphic designers) to easily create the town building grids, health and growth levels.

The goal and overall success of the project though, was only achieved by adopting the agile methodologies for both project management and game programming. These methodologies ensured that the project did not fall behind schedule at any point. Obstacles encountered were able to be quickly resolved. Feedback and suggestions given were able to be incorporated into the redesign and further development.

Finally, the human factors contributing to the success of the project must be acknowledged. The correct mix of appropriately skilled and knowledgeable team members, and the team's ability to work closely with the client, who in turn had to be available on a frequent and regular basis throughout the project, to enable timely client approvals was crucial. The opportunity for team members to gain feedback and suggestions from professional and experts within BABL was also highly beneficial to the project.

ACKNOWLEDGMENTS

The authors would like to thank Trudy Ellery, Head of Environment & Sustainability at the Bendigo and Adelaide Bank for the project proposal, her enthusiasm and ongoing feedback as the client throughout the project. Thanks also to the BABL experts and professionals who gave their time and expertise at consultative meetings in the project. Thanks are due also to La Trobe University for its ongoing support of project collaboration with industry.

REFERENCES

[1] "About Community Bank®." [Online]. Available: https://www.bendigobank.com.au/community/community-banking/about-community-bank. [Accessed: 30-Nov-2015].

[2] H. Mahmoudi, M. Koushafar, J. A. Saribagloo, and G. Pashavi, "The effect of computer games on speed, attention and consistency of learning mathematics among students," Procedia - Soc. Behav. Sci., vol. 176, pp. 419–424, 2015.

[3] N. E. Cagiltay, E. Ozcelik, and N. S. Ozcelik, "The effect of competition on learning in games," Comput. Educ., vol. 87, pp. 35–41, 2015.

[4] M. Soflano, T. M. Connolly, and T. Hainey, "An application of adaptive games-based learning based on learning style to teach SQL," Comput. Educ., vol. 86, pp. 192–211, 2015.

[5] J. Hamari, D. J. Shernoff, E. Rowe, B. Coller, J. Asbell-Clarke, and T. Edwards, "Challenging games help students learn: An empirical study on engagement, flow and immersion in game-based learning," Comput. Human Behav., vol. 54, pp. 170–179, 2016.

[6] "App Store." [Online]. Available: https://en.wikipedia.org/wiki/App_Store_(iOS). [Accessed: 10-Nov-2015].

[7] "Agile Methodology." [Online]. Available: http://agilemethodology.org. [Accessed: 11-Oct-2015].

[8] D. Cohen, M. Lindvall, and P. Costa, "An introduction to agile methods," Adv. Comput., vol. 62, no. C, pp. 1–66, 2004.

[9] D. Rover, C. Ullerich, R. Scheel, J. Wegter, and C. Whipple, "Advantages of agile methodologies for software and product development in a capstone design project," in Proceedings of the Frontiers in Education Conference (FIE), Madrid, Spain, 22-25 October, 2014, pp. 1–9.

[10] M. Coram and S. Bohner, "The impact of agile methods on software project management," 12th IEEE Int. Conf. Work. Eng. Comput. Syst., pp. 363–370, 2005.

[11] T. Smith, K. Cooper, and C. Longstreet, "Software engineering senior design course: experiences with agile game development in a capstone project," Games Softw. Eng., pp. 9–12, 2011.

[12] "SimCity is Now Open." [Online]. Available: http://developers.slashdot.org/story/08/01/12/1846256/simcity-source-code-is-now-open. [Accessed: 11-Oct-2015].

[13] "FarmVille." [Online]. Available: https://en.wikipedia.org/wiki/FarmVille. [Accessed: 11-Oct-2015].

[14] "Getting Started with XNA Game Studio Development." [Online]. Available: https://msdn.microsoft.com/en-us/library/bb203894.aspx. [Accessed: 03-Mar-2013].

[15] L. Williams and R. R. Kessler, Pair Programming Illuminated. Reading, Massachusetts: Addison-Wesley, 2003.

[16] "SketchUp." [Online]. Available: https://en.wikipedia.org/wiki/SketchUp. [Accessed: 11-Oct-2015].

[17] "Dropbox." [Online]. Available: https://en.wikipedia.org/wiki/Dropbox_(service). [Accessed: 11-Oct-2015].

[18] "OneNote." [Online]. Available: https://en.wikipedia.org/wiki/Microsoft_OneNote. [Accessed: 11-Oct-2015].

[19] "Skype." [Online]. Available: https://en.wikipedia.org/wiki/Skype. [Accessed: 11-Oct-2015].

[20] "TeamViewer." [Online]. Available: https://en.wikipedia.org/wiki/TeamViewer. [Accessed: 11-Oct-2015].

[21] "Facebook." [Online]. Available: https://en.wikipedia.org/wiki/Facebook. [Accessed: 11-Oct-2015].

[22] D. J. Fernandez and J. D. Fernandez, "Agile project management-Agilism versus traditional approaches," J. Comput. Inf. Syst., vol. 49, no. 2, pp. 10–17, 2008.