Xi'an Jiaotong-Liverpool University

西交利物浦大學

Specification and Design Report

Title	Maze solving game for robot based on Raspberry Pi
Author	Zhiyong Liu
Module	CSE305
Supervisor	Dr. Jieming Ma
Date	25^{th} / Nov / 2018

Chapter 1

Specification

1.1 Project Description

Now more and more people are supposed to learn knowledge about programming. However, learning programming can sometimes become boring that causes learners give up halfway. This project develops a maze game for those people who wish to learn programming knowledge. Users can learn programming knowledge by playing games at the same time.

In this project, there is a robot which is based on Raspberry Pi. This robot is to be an explorer in the maze and needs to find a path from the start position to the end position. To run out of the maze, robot is supposed to have an algorithm to execute. This project aims to design a maze solving algorithm and achieves the algorithm on the robot. In addition to this, the code programmed in this project also allows the users to design their own algorithm and execute it on robot. Users modify the codes according to algorithm designed and test it in the real maze, which achieves the aim of programming education.

Many algorithms such as wall follower [1], Pledge algorithm [2], and Trémaux's algorithm [3], were invented specially to deal with the maze solving problem, and each of them have their own strengths and weaknesses. Besides, a maze can be viewed as a tree or graph, some algorithms used in graph theory also have the ability to solve the maze solving algorithm. One of them is Depth-first search algorithm [4], it is used to traverse the tree or graph data structure. Therefore, through the Depth-first algorithm, the maze can be traversed by the robot and the path from origin to destination eventually can be found.

This project are supposed to develop a maze solving algorithm based on Depth-first search algorithm and work accurately on the robot in the real maze.

1.2 Statement of Deliverables

The Deliverable upon completion of the project is a software. The software is written in python and has multiple functionalities. In the first place, it is responsible for guiding the robot out of the maze. The software is intended to be configured in the robot in advance. When the robot moves in the maze, software will give the next step command for robot to execute according to maze solving

algorithm. Furthermore, the software allows users to modify pre-configured algorithm and design their own maze solving algorithm. The pre-configured algorithm refers to the algorithm based on the Depth-first search algorithm designed in this project. After users have learned the pre-configured algorithm, they can improve the default algorithm and run the new algorithm in the robot.

To evaluate the project, the first thing to do is to test if the designed algorithm is able to work accurately in maze solving problem. More specifically, the robot pre-configured with the software will be tested in multiple different mazes. In all tests, the number of getting out of the maze will be recorded. By calculating the success probability, the project will be measured if it can solve the maze problem. Besides, since this is a game-based learning project, the feedbacks of users will be collected. The feedbacks contain multiple aspects:

1. Whether users think the game is interesting.
2. Whether they think they can learn programming effectively through this game.
3. Whether they are satisfied with the project.
4. What other aspects they think for this project to improve.

1.3 Conduct of Project and Plan

1.3.1 Background and Related Work

As computer and mobile technologies advance, a amount of games for educational purposes have been used among learners of different levels [5]. This form of learning is called game-based learning (GBL). It has become the best solution for soft skills learning when traditional learning are homiletic, expensive and difficult to implement. Game-based learning aims to balance subjects with gameplay and players' ability to retain and apply subjects to the real world [6]. Children prefers to cost much time on playing, and learns steps of digital games. In this way, they can play and learn at the same time.

Games have been used in education for a long time. In the middle ages, noblemen learned strategies of war by using ancient chess game. During the Civil War, volunteers from Rhode Island played American Kriegsspiel, which had originally been created in 1812 for training Prussian officers-of-war. In order to train for Prussian officers-of-war, American Kriegsspiel was invented in 1812 [7]. Until now, a wide range of game-based learning applications are used, such as exploring ancient history with video games, teaching empathy with video games [8].

The development of sophisticated digital game technologies in recent years has generated an $8.1B industry around for educational purposes [9]. A lot of research on how game-based learning can enhance teaching are carried out. For instance, in order to encourage pupils to read frequently and improve their reading skills effectively, the reasearchers contains a game-like exercise in a prototype using Multimedia Fusion Developer 2. The game-like exercise is that foam volcano

character spews bubbles includes letters and words [10]. Then, children must read them loudly to open up.

1.3.2 Implementation

This project contains three major components, which are maze robot, algorithm, and maze environment. The maze robot is built based on raspberry pi, which can be controlled by software. The maze robot is equipped with multiple types of sensors, such as optical sensors, range sensors and so on. With the sensors, maze robot has the ability to achieve wall detection. The robot manufacturer provides the API of robot action, which can be used to realize actions of robot, such as moving, making a turn. The algorithm refers to the maze solving algorithm, which has been mentioned in project description. This algorithm is based on Depth-first search and considers the details in actual maze. The algorithm can be modified and improved by the users. Moreover, users are able to design their own maze solving algorithm to replace the pre-configured algorithm, which achieves the aim of game-based learning. The third component are maze environment, it contains multiple types and cases. In this project, the mazes are made of actual planks. The robot configured with the maze solving algorithm will be tested in these maze environments.

1.3.3 Project Plan

The schedule of this project is based on the aim of project, which has been described in project description. The Gantt chat is shown as Fig. 1.1. In the whole process of this project, only one member participate in, which is the author of this report. The first milestone is to design an algorithm which can guide the robot to out of maze. The first stage takes five weeks and the designed maze solving algorithm is stated in chapter 2 in detail. The second milestone is to implement the basic actions of robot. The robot actions contain basic movement, making a turn, and wall detection. At this stage, the major work relates to programming with the API of robot, the actions programming modules will be output when the second milestone is achieved. The third milestone contains two tasks, algorithm implementation and configuration in robot. In this stage, the designed algorithm in milestone 1 tends to be implemented in python. Once it has been implemented, the related code will be configured in actual robot for later stage of tests. Then, to achieve the milestone of test, the robot configured with related algorithm code will be tested in actual maze environments. Tests are intended to carry out in different mazes for multiple times. The success rate of goal achieving will be counted. The last milestone are feedback collection and documentation. At this stage, multiple users try out the delivered product and feedbacks will be collected. In the end, the documentation of this project is supposed to be completed.

1.3.4 Risk Assessment

The major possible risks to meet and aversion measures are shown in Table 1.1.

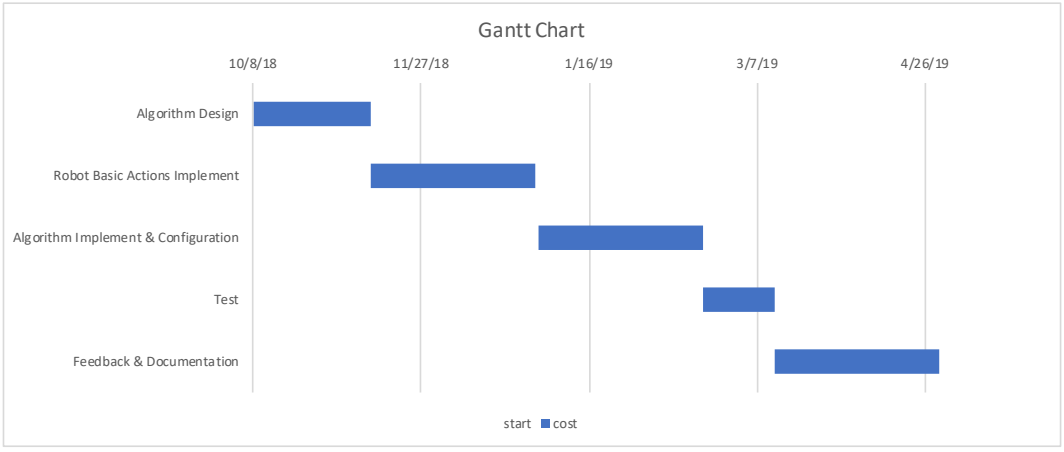

Fig. 1.1: Gantt chart of project

Table 1.1: Title Of Table

Risk	Description	Probability	Severity	Actions to Minimize Risk
Robot break down	The hardware of robot breaks down in the process of project development.	Moderate	Catastrophic	During the development of project, developer takes care of project equipment. Besides, developer can prepare a spare robot if development funds are adequate.
Schedule risk	Due to assignments in other courses, the project fails to run as schedule expected.	Likely	Major	Try to finish ahead of time, and balance well with other courses.
Technologies applied failure	The knowledge of controlling robot needs large amount of learning, developer fails to learn in time.	Likely	Major	Developer specially arranges time for learning new techniques.

Chapter 2

Design

2.1 Design methodology

This development of this project is about to adopt incremental model. The product will be designed, implemented and tested incrementally, until the project is completed. The product consists of multiple components, each of which is designed and built separately. The increments refer to a series of releases, and each increment will provides more functionalities to users [11]. After the first increment, a version which can be used by the users is delivered. Then, the plan for next increment is developed based on user feedback. This process will continues until the complete product is delivered. The general process of increment model is shown in Fig. 2.1.

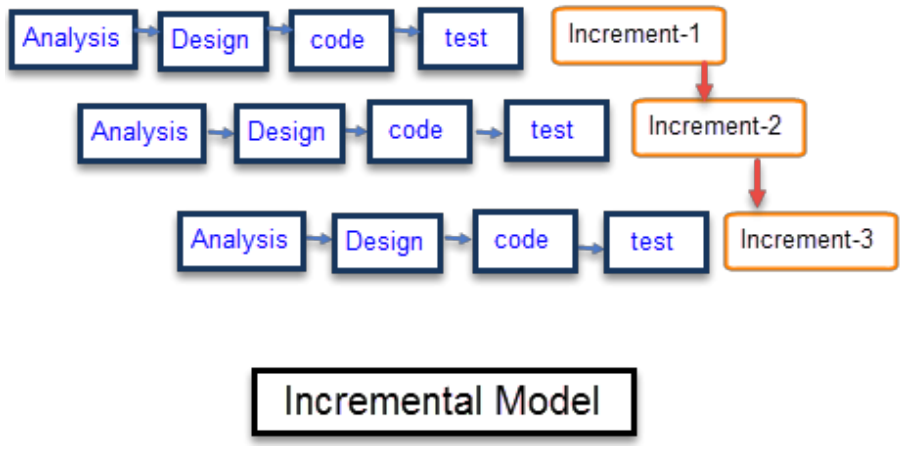

Fig. 2.1: Increment model

2.2 Data Structure

The MSA uses stack to store the current path of robot and record the positions of crosses. Stack is chosen because its property of First-In Last-Out (FILO) [12]. The first element put in the stack will be the last element to be pop. After the robot reaches the destination, the position list popped

by the stack is exactly the path from start position to end position. Besides, tree structure is also can used in this project for additional path drawing functionality. While robot is moving in the maze, the position of robot can be recorded in tree structure as nodes. Once the algorithm execute completely, all paths in the maze have been recorded in tree structure.

2.3 Algorithm Design

The designed maze solving algorithm (MSA) in this project use the recursive models [13]. In each recursive step, it will detect if the destination has been reached. If not, MSA will choose the next available position to move. The search for available direction is carried out in the order of left, positive, right and back. If it exists available exploration direction, the robot will move to the next point in that available direction and regard this position as new start position. In this process, MSA will guide robot explore as deep as it can until the current path of exploration is not available. MSA will explore another path from last crosswise. The whole process is repeated until the robot reaches the destination.

The pseudocode and flow chart of MSA is shown as Fig. 2.1 and Fig. 2.2 accordingly:

```
stack current_path;  //Save the current path
stack cross;       //Save the cross position

MSA(start, end)

Push the start position into the stack current_path
Get the top element n of the stack current_path

if n equal to the position of end
    save the stack current_path to all_path
    end
else
    if crossCheck(n)
        if n exists in stack cross
            pop the stack current_path to the last occurrence of n
        else
            push n into the stack cross

    i = the position of next point in available moving direction of n
    // The available moving direction is searched in this order: left hand, positive,
    // and right hand, opposite direction of robot.
    MSA(i, end)


crossCheck(position)
    if this position has two or more directions available to move
        // Only three directions are considered: left hand, positive,
        // and right hand direction of robot.
        return true
    else
        return false
```

Fig. 2.2: Pseudocode of MSA

There is one most important part in this algorithm. This algorithm uses a stack to store crosses

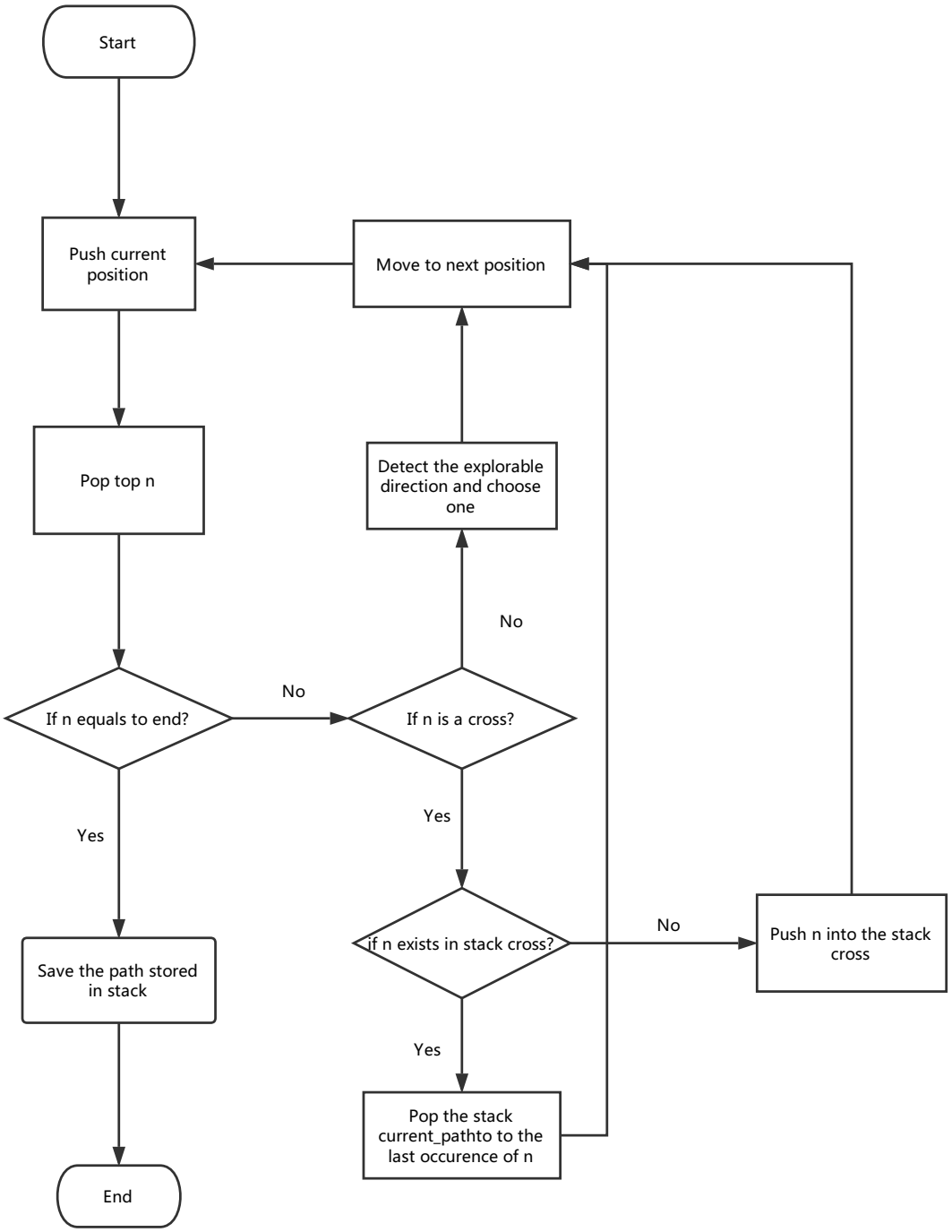

Fig. 2.3: Flow chart of MSA

while robot is traversing the maze. The cross refers to the position which has more than one direction available to move. When robot moves to a new position, the robot configured with this algorithm will first test whether this position is a cross. In this three directions: left hand, positive and right hand, if there is one more direction for robot to move, this position will be regarded as a cross. Then this position will be checked whether it has existed in the stack cross. If this position has not existed in the stack cross, this position will be regarded as a new cross and pushed into the stack cross. This recorded that the robot has reached position once. However, if this position has existed in the stack cross, it means that robot has reached this position once in the previous time. Then, the stack current_path will be popped to the last occurrence of this position. This is because the path from the last time robot got to this position to this time is invalid and it will not regarded as one part of the final path from start to end.

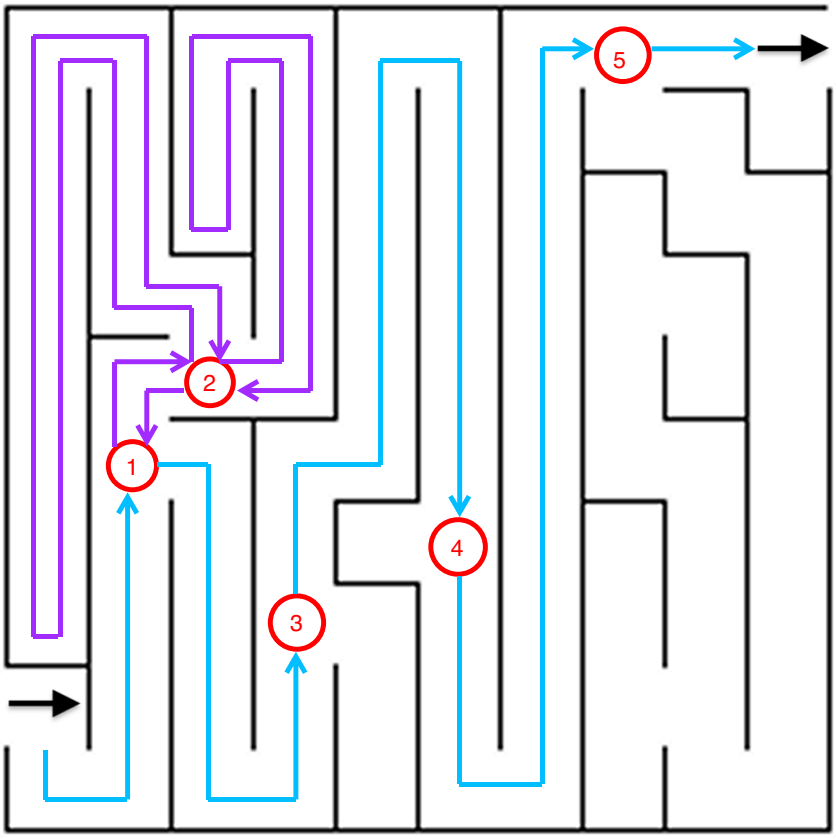

Fig. 2.4: Use MSA in a simple maze

In the Fig. 2.4, the MSA is used to obtain the path from start to end. The red circle with the number denotes the cross when robot is traversing the maze. Due to reach the same position for multiple times, the path denoted by purple line has been popped from the stack current_path. The purple part is not regard as the part of the final path. Therefore, the blue path denotes the final path from start to end.

2.4 Evaluation Design

2.4.1 Test

In the test stage, The primary task is to perform functional testing from low level to high level [14]. First, the each unit in this project is intended to be tested. Developer will first validate the correctness of each unit and check whether the codes are in a canonical format. Then, the functionality of each module will be test. For instance, the action module of robot will be test in terms of moving, making a turn. The third stage is integration testing [15]. The functionality of several modules that depend on each other will be tested. Finally, the function of the entire project is supposed to test. The robot configured with designed maze solving algorithm (MSA) should operate accurately in the actual maze in this stage. The tests will be repeated multiple times in different maze environment. In each type of maze, tester continues to test, and record the time and the solution path. In the end, the success rate of experiments will be recorded and used to evaluated.

In addition, according to project specification, this project allows users to modify the pre-configured algorithm and design their own algorithm. To test this requirement, some programming enthusiasts are intended to be invited to use this product. They modify the program code and try other algorithms, then the feedbacks which mentioned in project specification are collected.

To evaluate the project, the first step is to evaluate if this project satisfies the project specification. The aims proposed in the project specification:

1. Design a maze solving algorithm and work accurately in robot.
2. The delivered product allows users design their own algorithms and implement in robot.

2.4.2 Design a maze solving algorithm and work accurately in robot

In the previous test stage, tester carried out a lot of experiments and the results have been recorded. The evaluation of the designed algorithm MSA will be based on its success rate, which is calculated as shown in Fig. 2.4:

$$s = \frac{total\ success\ times\ in\ different\ mazes}{total\ test\ times\ in\ different\ mazes} \times 100\,\%$$

Fig. 2.5: Success rate equation

2.4.3 The delivered product allows users design their own algorithms and implement in robot

To validate this aim, the feedbacks from the users which have been collected in previous test stage will be considered:

1. is it user-friendly to modify the program code?

2. Whether they are satisfied with programmable functionality.

Chapter 3

Review against Plan

Generally, the process of this project follows the plan Fig.3.1. The work progresses to the point indicated by the red line in the Gantt chart. The first milestone Algorithm Design has been completed and status has been updated. Besides, the Implementation of Robot Basic Actions is under way. The remaining work shown in Gantt chart will be completed in following semester.

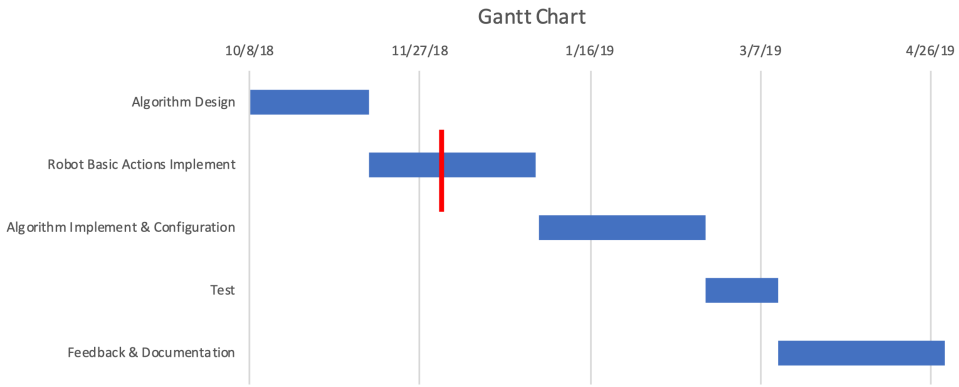

Fig. 3.1: Flow chart

Reference

[1] H. W. Bullen IV and P. Ranjan, "Chaotic Transitions in Wall Following Robots," *arXiv.org*, Aug. 2009.

[2] R. Klein and T. Kamphans, "Pledge's Algorithm - How to Escape from a Dark Maze." *Algorithms Unplugged*, no. Chapter 8, pp. 69–75, 2011.

[3] "Solving a Maze," in *Building Robots with LEGO Mindstorms NXT*. Syngress, Jan. 2007, pp. 327–348.

[4] A. Brændeland, "Depth-first search in split-by-edges trees," *arXiv.org*, May 2015.

[5] J.-N. Proulx, M. Romero, and S. Arnab, "Learning Mechanics and Game Mechanics Under the Perspective of Self-Determination Theory to Foster Motivation in Digital Game Based Learning," *arXiv.org*, no. 1, pp. 81–97, May 2018.

[6] D. Ifenthaler, D. Eseryel, and X. Ge, *Assessment in Game-Based Learning*, ser. Foundations, Innovations, and Perspectives. Springer Science & Business Media, Jun. 2012.

[7] M. Fenn, "Kriegsspiel mit Herz? Computer Games zum Ersten Weltkrieg," *Public History Weekly*, vol. 2014, no. 26, 2014.

[8] M. Prensky, *Digital Game-Based Learning*. Paragon House, Mar. 2007.

[9] B. An, I. Kim, E. Pakdamanian, and D. E. Brown, "Exploring Gaze Behavior to Assess Performance in Digital Game-Based Learning Systems," *arXiv.org*, Nov. 2018.

[10] M. Adrir' Scott, "Vocalnayno: Designing a Game-Based Intervention to Support Reading Development in Primary Schools," *arXiv.org*, Apr. 2013.

[11] M. Bell, *Incremental Software Architecture*, ser. A Method for Saving Failing IT Implementations. John Wiley & Sons, Jan. 2016.

[12] C. Roser and M. Nakano, "Guidelines for the Selection of FIFO Lanes and Supermarkets for Kanban-Based Pull Systems - When to Use a FIFO and When to Use a Supermarket." *APMS*, vol. 460, no. Chapter 33, pp. 282–289, 2015.

[13] B. Khoussainov, A. Nies, and R. A. Shore, "Recursive models of theories with few models," 1995.

[14] M. P. Reiman and R. C. Manske, *Functional Testing in Human Performance*. Human Kinetics, 2009.

[15] S. Kandl and M. Elshuber, "A Formal Approach to System Integration Testing," *arXiv.org*, Apr. 2014.