

What are the difficulties of implementing algorithms that procedurally generate game maps, in a highly coupled system?

COMP160- Software Engineering

1607804

March 20, 2017

1 Introduction

The implementation of new functionality to any existing system comes with complications. The one complication that the author wishes to focus on in this essay is coupling. In the following essay, the author aims to provide insight on the difficulties of adding new procedural content generation (PCG) algorithms for the generation of maps. A description of problems that may arise from high interdependence will allow the reader to notice if any of these symptoms are apparent in any work they have done or are doing. In the game industry, you may have to work on a game system designed by someone else. In these cases there may be little to no structure. By extrapolating research theory and case studies from the software engineering industry, the author will suggest strategies to keep the integrity of your data when implementing the PCG algorithms. In this essay we define 'game maps' as the physical environment the player has to transverse whilst playing the game. Level design can be a tedious cycle of designing and playtesting to

achieve the 'perfect' level. A level which is challenging enough for the player to enjoy but not too challenging it discourages the player [1]. This process is not only time-consuming but also expensive. One way to keep the players engaged but also to save resources is to automate this process through the use of PCG.

Write your introduction here. A brief introduction is recommended, which should outline key details of the chosen topic and the reviewed papers, motivate the work, and provide a roadmap of key points to the reader. The motivation is quite important here, as essays should have a contribution (i.e., what is the point of the essay, and what does the reader take away from the essay) and the link between the motivation (in the introduction) and the contribution (in the conclusion) should be made clear.

2 Discussion

In this essay, the content being procedurally generated is limited to world static objects. This includes but is not limited to: Environment layout, Spawning locations, static enemies (turrets for example). When attempting to generate a map one needs to be aware of any circular dependencies about the object being called. Calling an object in a way so that one of the objects cannot complete may result "in a deadlock or data loss" [2]. When implementing an algorithm that uses modules that have such dependencies special care must be taken to call them in a way that they will be successfully completed, or that if a deadlock does occur there are measures to solve them. (Needs case study or another such reference).

When choosing the objects you will spawn to generate your map consider carefully if any of them contain traces of content coupling. Content coupling "is considered the worst type of coupling" [3, p. 95]. This occurs when a private method in one class can be invoked from another class. This means that not only is the code difficult to understand but also the one cannot be changed independently of the other. When this is applied to map generation the results may be unpredictable. This high interdependence typically

only occurs in dynamic objects, objects that need to respond to others and alter their behaviour on the behaviour of other world objects. By limiting yourself to using only static objects in your generation algorithm you can avoid the uncertainty of

Using a set of 'building blocks' like sets of manually designed rooms to populate your map will result in a map that will look different with each generation but will fundamentally be the same.

3 Conclusion

Write your conclusion here. The conclusion should do more than summarise the essay, making clear the contribution of the work and highlighting key points, limitations, and outstanding questions. It should not introduce any new content or information.

References

- [1] F. Kayali and J. Schuh, "Retro evolved: Level design practice exemplified by the contemporary retro game," in *Proceedings of DiGRA 2011 Conference: Think Design Play*, 2011.
- [2] S. Nair and R. Jetley, "Solving circular dependencies in industrial automation programs," in *Industrial Informatics (INDIN), 2016 IEEE 14th International Conference on*. IEEE, 2016, pp. 397–404.
- [3] L. C. Briand, J. W. Daly, and J. K. Wust, "A unified framework for coupling measurement in object-oriented systems," *IEEE Transactions on software Engineering*, vol. 25, no. 1, pp. 91–121, 1999.