A Perspective on using Empirical Modelling for Concurrent Systems Modelling

Abstract

This paper explores the use of Empirical Modelling (EM) in modelling concurrent systems. This is often preformed with the use of formal mathematical techniques. The paper will concentrate on the use of EM techniques in allowing users to observe and interact with a concurrent system. In specific it will investigate the value of visualisation and the principles EM provides for developing concurrent systems with a comparison of the ability to view the system as an external observer and the role of visualisation with existing methods of modelling. To help identify the aspects looked at during the course of the paper, a model of a library system has been developed with reference to the key EM principles.

1 Introduction

Modelling concurrent systems is traditionally a difficult area in Computer Science. The ability to define the interaction between various agents at work in both a system and the world as a whole is commonly a complex process.

Empirical Modelling (EM) is an attempt to model systems in a less restrictive sense. The main thrust of the notion of EM is that systems can be modelled with reference to a much wider range of influences and not be constrained to a strictly defined environment. The subject has a place in modelling concurrent systems by providing a general framework that allows greater exploration by a user when developing such systems. The paper aims to explore the potential of the promises that EM with respect to the value placed in visualisation of the system as an external observer with the aid of agentification and visualisation of a particular situation.

2 Existing techniques and practices

The problem of modelling concurrent systems has been a focus of research for some time. Due to the complexity and scale of the many potential situations that can arise, methods have been developed to reduce the nature of this complexity. The two most common techniques are simulation and formal methods.

2.1 Simulation

Exploring a concurrent system maybe achieved through the use of simulation. This demands that some sort of model is formed about how a system works, executing that model and then analysing the output. This allows a more flexible approach to modelling, alleviating the need for strict definition of behaviour and focusing on how a system behaviours with respect to its inputs. A user may be then more reactive to problems and surprises that may not have been originally seen if the system was simply considered on a purely abstract level. An example of this technique can be seen in PDES[1] and tools such as simpack[2] .

2.2 Formal methods

Formals method aim to model the system in such way that its behaviour can be defined in some manner and proven. This is traditionally by a system of logic, such as CSP[3] and CCS[4] . This is more restrictive approach and demands a complete understanding of the system using the abstracted concepts by the models. For instance CSP was developed to facilitate synchronised communication between parallel processes.

[1] Parallel Discrete Event Simulation [1]

[2] Simpack. C Routines for creating simulation environment. More information available from Paul Fishwicks home page http://www.cis.ufl.edu/~fishwick/

[3] Communicating Sequential Processes [2]

[4] Calculus for communicating processes [3]

3 Empirical Modelling and Concurrent Systems

EM[5] expresses the notion of concurrency by referring to “commonsense concurrency”. This relates to the fact that the external observer plays a key role in developing a system. In order to express the concept of concurrency, EM has a number of principles and tools. The principles outline the need to identify agents and their interaction whilst the tools allow users to create an implementation of this interpretation. Once an implementation of this “view” is created then the external observer may interact with it to observe how the system behaves with respect to its environment. Concurrent systems then arise from this as the agents can be seen as acting independently at the same time but may still interact with other agents.

3.1 Principles

3.1.1 Agentification

EM generally looks to model systems by looking at interaction in a much more broader sense than conventional practices. Capture and design of requirements with respect to the environment is a strong focus. Deriving fundamental techniques for model construction promoting better development of concurrent systems are also key aims for the approach. EM focuses on using agents as a primary concept rather than restricting the approach to things such as processes. Developing systems using a more agent orientated approach begins to focus the idea of an external observer and the role they play in developing a construal of a system. This then broadens the consideration of factors without the restriction of more abstract mathematical techniques.

3.1.2 Consideration of observables and state

The idea of agency is a central concept and can be considered to be a source of state change. If these are capable of changing the state of an environment then it must be clear what an agent is capable of observing and what is capable of changing. As a developer of a concurrent system, one would assume the role of an external observer who is responsible for identifying the agents, observables, dependencies between them and what actions are possible within the constructed environment.

[5]More Information can be gained by visiting www.dcs.warwick.ac.uk/research/modelling. A good starting point is Radical Empiricism, Empirical Modelling and the nature of knowing byW.M.Beynon. In Proceedings of the WM 2003 Workshop on Knowledge Management and Philosophy, Luzern, April 3-4, 2003.

3.2 Tools

These are the tools that allow the implementation of EM models

3.2.1 Definitive Notation and EDEN

To express EM principles and create real world models, a definitive notation is used. This a simple formal language and allows a representation of state. The language allows a script to be built using this notation which describes the way state is changed. EDEN is an interpreter for evaluating definitive notations and allows programming of definitive scripts. Animated output is supplied by the DoNaLD and SCOUT interfaces.

3.2.2 LSD Notation

LSD notation was developed by Meurig Beynon[4], with further development from Mike Slade in ADM implementation [5]. It aims to specify the agents involved in a system. The specification attempts to formalise what an agent can observe and how it can affect the state through performing defined actions.

3.2.3 ADM

This is an attempt at generalising the definitive programming model. It is primarily used for animation of LSD scripts to represent concurrent action by parallel redefinition and using scripts to express context dependence of agents.

4 Comparison of EM to existing methods

It has been shown that applying the principles of EM and creating an implementation using the tools can give a visual representation of an observer's construal of a particular situation. The ability to interact with the environment and visually comprehend the significance between the agents is a powerful aid to developing a concurrent system. This system of development and interaction will be considered with respect to the two existing systems of simulation and formal methods.

4.1 Simulation and EM

Simulation is a powerful concept in areas where large, complex systems are investigated. Often the systems or so complex that formalisation of the behaviour is simply too time consuming to be viable. This has a similar outlook to EM in that the behaviour of a system in an unpredictable environment is only fully comprehended through investigating how the system reacts when placed in a particular situation and seeing the results.

A large downfall of this approach is that the system is effectively seen as a black box and the interaction with the system components is not clear. In a complex system where there are many sources of state change, understanding and visualising of the system becomes difficult. The main difference is that in simulation the state of the system can only be changed once by the user, at the beginning. EM on 1024the other hand, with the focus on agents ,will allow a user acting as an external observer to change the system state whilst the system is running.

The advantage of simulation is simply the wealth of states that can be applied though. The ability to input data, output and test is much faster than changing individual sections of a system and then observing a change. The advantage of EM however is that change can be seen as the system is running as opposed to testing the final result. It maybe that closer identification of key influential areas maybe better than simply entering huge amounts of random data.

4.2 Formal methods and EM

EM aims to have a much broader outlook to formal methods as a whole. Formal methods create an abstract, restricted model that will allow some way of defining the behaviour of a system that can be proven. The advantage of this is clear, a user will know their system is correct, acts as they wish in the environment described by the initial abstract model. For such system critical applications as medicine, aeroplane systems etc this is clearly the best choice.

However, the model does not comprehend the concept of external factors outside the system that may arise in the environment. The result of this is that it does not truly reflect the influences that may appear. It does not provide visualisation for the user either and interaction cannot be clearly seen. The system is clearly defined by formal methods whereas EM allows exploration and so becomes a powerful tool for visualising

4.3 Conclusion

Looking at the existing methods it can be seen that EM has some advantages owing to its ability to provide visualisation of a running system as well as providing a framework for extending the possible interaction to the world rather than restricted to a particular abstract model.

5 Library Model

To help express and highlight the usefulness of the EM modelling techniques, a model of a library will be developed.

5.2 Aims and assumptions

The library is modelled as a set of users, who may take out a set of books from a library. There are 5 members, 3 employees and a total of 8 books. The 5 members queue to be served and when an employee is free they are served. The different employees take different times to process of dealing with a member. Each user may want to take out a particular book and/or may wish to take a book out if available.

The model aims to allow a user to change :

The time it takes for an employee to process a members request

The books a user wishes to have

The ability to change the values as the model progresses will illustrate the power of EM to allow a user to change the values of a system and to view the changes to the system. This will show the user to instantly see the impact of their changes on the other agents.

5.2 Implementation

The agents in the model will be defined using LSD specification. The agents that are identified in this situation are the employees, members and the database. This will turned into definitive notation to create a definitive script for evaluation by the EDEN interpreter. The script will be animated using the SCOUT notation for the screen layout and window creation and the DoNaLD notation for describing and creating the objects on the screen. The stress is develop the system from the role as an external observer.

5.3 EM principles involved

The whole project was built using definitive notation and ran in using the EDEN evaluator. Visualisation feedback is given in three ways, the colour changing of the members to represent their state, the state of the database is given as well as the state of the employees. This reflects the external observation of the user. The ability to change the variables and view the results is an example of visualisation that allows user to observe the interaction occuring within a system. The LSD specification helped define the interaction that was expected.

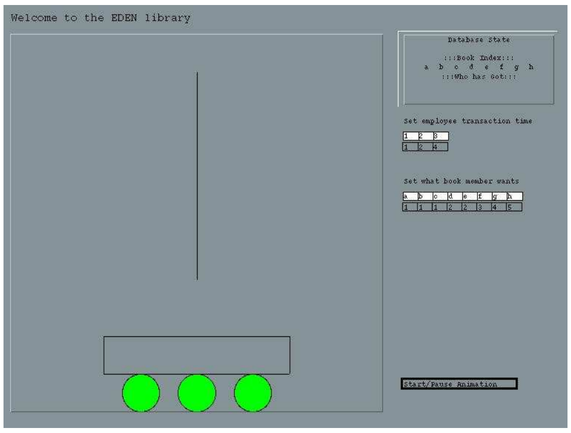

Figure 1: Example of Interface

5.4 Issues

Various issues and problems arose through the use of the EDEN environment. The first problem was successfully animating a project using the DoNaLD and SCOUT notations, the concept of time based activity is not clear when programming and required an unorthodox method. The lack of high level structure meant that a lot of code was repeated. To display similar objects and artefacts it meant building similar sections of code that often differed only by their parameter. The syntax proved to be a problem with the styles differering between the notations.

5.4 Outcome

The outcome was postive despite a minor problem with not being able to find a solution to controlling the frequency at which the animation refreshed. The ability to change the variables and watch the outcome was a postive step towards exploring the implications of the environment. Any changes are reflected in the database.

5.5 Conclusion

The common existing methods for modelling concurrent systems have been identified. The principles that EM holds for viewing systems has been outlined, together with the available tools. Finally, a basic comparison of EM and existing methods is given with reference to a model illustrating the concepts that give EM an advantage. The model is simply a small scale representation so the impact of EM on a large scale system is not identified. The model represents the role of user interaction and the value in visual feedback.

Acknowledgements

Meurig Beynon/Warwick Modelling group

References

[1] B.J. Overeinder; L.O. Hertzberger and P.M.A. Sloot, In W.J. Withagen, editor, *The Third Workshop Computersystems*, Faculty of Electrical Engineering, Eindhoven University, pp. 19-30. Eindhoven, The Netherlands, May 1991.

[2] Hoare, C.A.R. “Communicating Sequential Processes”, Communicat-ions of the ACM, 21 (8):666–677, (1978).

[3] Milner, R. *A Calculus for Communicating Systems*, Springer-Verlag,Lecture Notes in Computer Science, volume 92 (1980).

[4]W.M. Beynon, *The LSD Notation for Communicating Systems* (November 1, 1986).

[5] Mike Slade. Definitive Parallel Programming. MSc thesis, University of Warwick, April 1990.