Chapter 4

Simple System Dynamics and Control System Project Models

A. S. White
Middlesex University, UK

ABSTRACT

This chapter examines the established Systems Dynamics (SD) methods applied to software projects in order to simplify them. These methods are highly non-linear and contain large numbers of variables and built-in decisions. A SIMULINK version of an SD model is used here and conclusions are made with respect to the initial main controlling factors, compared to a NASA project. Control System methods are used to evaluate the critical features of the SD models. The eigenvalues of the linearised system indicate that the important factors are the hiring delay time, the assimilation time, and the employment time. This illustrates how the initial state of the system is at best neutrally stable with control only being achieved with complex non-linear decisions. The purpose is to compare the simplest SD and control models available required for “good” simulation of project behaviour with the Abdel-Hamid software project model. These models give clues to the decision structures that are necessary for good agreement with reality. The final simplified model, with five states, is a good match for the prime states of the Abdel-Hamid model, the NASA data, and compares favourably to the Ruiz model. The linear control system model has a much simpler structure, with the same limitations. Both the simple SD and control models are more suited to preliminary estimates of project performance.

INTRODUCTION

Software projects, as reported in the Chaos reports (Standish group 2009), still have a low success rate in terms of reliability, meeting due dates and working within assigned budgets (Smith 2002, Yeo 2002, Royal Academy of Engineering 2004), despite recent disputes (Glass 2005, Eveleens and Verhoef 2010) about the accuracy of the Chaos report descriptions. Factors, which determine successful project management (Abdel-Hamid and Madnick 1989), may be related to technical production processes, time scheduling in a dynamic environment and individual differences in project managers, members and team processes. Capers Jones (1996) has estimated that such projects only have a success rate of 65%. Projects may be considered as a system in which demands are
made (the requirements) and an internal project organisation, is controlled to produce the software goals, while being disturbed by the external environment.

The cost of such disasters such as the UK National Air Traffic System, Health Service computerisation and the London Ambulance Service computerisation is high in both money and human terms.

Despite these failures significant progress has been made in the use of System Dynamics (SD) methods to describe the operation of software projects (Rodrigues and Bowers 1996). Other workers such as Lin & Levary (1989) describe computer aided software design using System Dynamics, expert systems and a Knowledge based management system used in the design of a space station. More recently Häberlein (2004) has discussed the common structures involved in SD models. He contrasted the Abdel–Hamid model, which uses tasks completed with an approach based on requirements. Although we do not use this here the idea has considerable merit as the Abdel-Hamid approach has difficulty including requirements phases. His approach would be more suitable to use when modelling agile methods for example. This point was made by him in referring to the universality of the model. Rodrigues & Bowers loc. Cit. and Lyneis & Ford (2007) have established the role of System Dynamics in project management, pointing out the dominating effect of rework and the way factor such as quality or productivity affects project performance. Rahmandad and Weiss (2009) describe work investigating the dynamics of concurrent software development, illustrating the feedback loops dominating inter-project dynamics and tipping points. An excellent presentation by David Ford (2009) highlights in a very clear way the essential feedback processes arising in all projects, not just software development, illustrating how the projects run into problems as indicated by the SD models, leading to very clear solutions with regard to the tipping points on the project. Earlier in 2007 David Ford and James Lyneis have given an excellent state–of-the-art synopsis of where SD methods had placed project research, indicating how SD models had been used in litigation after project failures. They made a plea for better post project analysis and use of SD methods for estimation of costs and manpower planning. A significant milestone has been the use of an SD model in agile software development by Cao, Ramesh and Abdel-Hamid (2010). This will show the software community that thought SD project models were only applicable to Waterfall processes that they were incorrect. Trammell et.al. (2013) has used SD methods to analyze the effects of funding fluctuation on software development. A recent comprehensive analysis by Mingers and White (2009) of Systems Thinking and its’ contribution to Management Science reviews the contribution of SD to many fields of study but doesn’t stress the very important contribution that SD project analysis has made, possibly the greatest financial contribution of any Systems methodology. System Dynamics models are now widely used in construction (Zhang et. al. 2014) to assist with planning, risk management (Boateng et.al. 2012) and defense project control (Cantwell 2012).

Some of the earliest attempts at creating project models by Roberts (1964) led to the very successful SD models of operation of the software development process models of Abdel-Hamid & Madnick (TAH) (1991). These TAH models use equations relating levels such as the number of perceived errors, or the number of reworked errors and relate them to rates such as the error detection rate or the rework rate, significant features of these models included the decision processes. These models were validated against NASA project data for a medium size project and the agreement is strikingly good. The SD model structure is highly non-linear with a number of theoretical assumptions, for example about how the errors in the coding are propagated. These structural assumptions do not allow for System Dynamics models to enable any general rules to be developed by academics to...