The Simulator; Another, Elementary Problem Frame?

Ian K Bray
Bournemouth University (ibray@bournemouth.ac.uk)

Abstract

Problem framing is being recognised as an important new technique for software development.

In order to better model certain problem domains, an addition to the published set of elementary problem frames is proposed.

1. Introduction

Since being proposed by Michael Jackson [1], problem frames have been recognised as having great significance for the development of software systems. The approach is grounded upon fitting problem frames to the given problem domain; the closer the fit, the greater the advantage that may accrue.

Jackson originally identified five elementary problem frames; workpiece, control, information, connection and transformation. As has been illustrated in several works (for example, [2], [3]) more complex problems can, generally, be fitted to composite frames composed of two or more interacting, elementary frames.

Whilst Jackson explicitly made no claim to have identified all the elementary frames, as far as is known, to date, no others have been added. However, to better fit a particular class of problem, an additional frame is now proposed.

2. The simulator problem

This class of problem is characterized by the construction of an artifact (a realised domain) which exhibits behaviour that, to a required degree, mimics the inherent behaviour of some dynamic, real world original.

The simulation bears some resemblance to an external reality but this is established only through the requirements; there is no communication (no phenomena are shared) between the original and the simulation during operation. Even so, the original should not be ignored, it is central to the problem and investigation and description of the original would be a vital element of the requirements engineering.

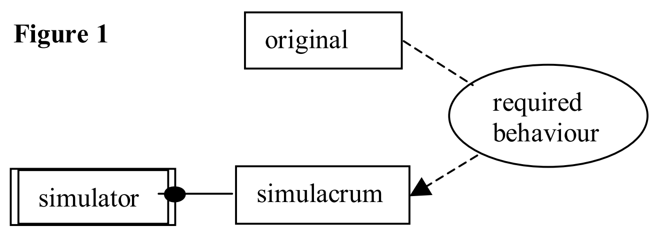
The problem frame may be pictured as in figure 1.

The original is a "real world" domain. It is suggested that the original must be dynamic (or it would have no behaviour to simulate) but that is the only constraint[1]. In particular, it may be controllable or autonomous.

The required behaviour defines the behaviour required of the simulacrum. Clearly, this will be based upon the inherent behaviour of the original but (as illustrated in the following examples) it is likely that there will be approximations or simplifications.

The simulacrum is the realized domain which behaves in accordance with the required behaviour. It is an embodiment of the selected event responses and exists only within the simulator, the configured machine that brings about the desired behaviour of the simulacrum.

Figure 1



The simulator frame is likely to be used in conjunction with other frames. In particular, although it could be trivial, an information system may well be required in order to view the state of the simulacrum. (This is akin to the simple workpiece frame requiring an information system to provide feedback [5], page 98.) It is also possible for the inherent behaviour of the simulacrum to be controlled by an associated control frame. However, these associated frames are optional; the simulator can stand alone and this prohibits its classification as a partial frame.

3. Examples

Consider a simulation of a planetary system (an abstract orrery, in the case of the solar system). The original has certain properties (mass and orbital period of bodies etc.) which can be detailed in a problem domain description. The requirements will state with what

[1] There may be some argument as to whether a model of a static domain (for example, a road in a traffic simulation) constitutes a simulation in its own right.

degree of fidelity the simulation should mimic the original. There will inevitably be some inexactitude and one might, for example, elect to treat the bodies as point masses, to constrain movement to a single plane, ignore the effects of small bodies and to change the timescale.

Once implemented, the simulation should comply with the requirements. However, during its operation, the model does not communicate (share phenomena) with the original and at any time, the state of the model may be very different from that of the original.

A second example concerns the simulation of ocean waves. The problem domain description would detail such matters as the propagation and decay of actual waves. The requirement would specify how closely the simulation must mimic these characteristics and the implementation should comply with those requirements.

The simulation could be used to make predictions about maximum wave heights, to provide test conditions for (simulations of) boats or simply as a piece of art. In none of these cases, however, does the simulation communicate (share phenomena) with the original or represent any actually occurring state of the original.

Thousands of examples of simulation problems appear in the literature, these include:

- simulation of chemical etching
- simulation of buildings (for earthquake research)
- simulation of vehicles and drivers (as part of a driver training environment)
- simulation of electronic circuits
- simulation of brewery fermentation (as test rig for a new controller)

4. Differences from other frames

The simulator shares certain characteristics with other elementary frames but has particular features which justify its separate classification. In part, the differences hinge upon the distinction between inherent behaviour and controlled behaviour. The inherent behaviour of a domain is a given. It is not open to negotiation but may be described more or less correctly. A complete description of the inherent behaviour of a domain encompasses all its possible behaviours. Bjorner et al. [4] define this in terms of a basis function which determines the range of its observable variables.

The simulator seeks to approximate the inherent behaviour of the original. If it is possible and desirable to impose a controlled behaviour over the inherent behaviour, this would be the subject of a separate (control) frame.

Jackson [5], page 339, offers a ranking of problem frames in terms of their "depth in the world outside the computer". This ranking runs from the transformation frame (the shallowest) to the control frame (the deepest) and may be viewed in terms of the degree of binding between the machine and the real world. Within this scheme it is suggested that the simulator problem fits between the workpiece and the information frame; the behaviour of the simulacrum is based upon some real world original (which, therefore, requires study) but during operation, there will be no communication between the simulator and the original.

5. Why isn't it a workpiece problem?

Both workpiece and simulator contain a realized domain (artifact). In the workpiece case, the realized domain is necessarily inert and it changes subject only to the whim of an editor. The simulacrum can be self modifying and is constrained by requirements that establish, to a greater or lesser degree, some mimicry of an external reality.

An inert domain may be simulated but, in the absence of any stimuli, will do nothing. It is hard to imagine the circumstances under which the simulation of an inert domain would, of itself, be worthwhile.

An inert domain simulation could, however, have a role as a partial frame. It would embody the event responses that would allow some other machine to exert control over the simulacrum. This approximates to the decomposition of the workpiece problem into two partial frames; a simulator and an "editor".

However, there remains the difference in that, with the simulator, there does exist an external original which, through the requirements, imposes constraints upon the behaviour.

Furthermore, the simulator has a far greater range; spontaneous and, even, autonomous domains may be simulated. Even in the absence of any external stimuli, such domains may "freewheel" and continuously exhibit behaviour. The bodies in the planetary simulation, for example, continue in their orbits without the need for any external stimulation and, in a brewery simulation, a vat of wort will continue to ferment if left alone.

6. Why isn't it an information problem?

An information system provides information about some reality. In order to do this, the information system will frequently contain a model of that reality (what Jackson refers to as the Dynamic Model, Partial Frame [6]). However, in that case, the correspondence between the reality and the model must be maintained (within acceptable limits) during the operation of the information system.

This is not the case with a simulator. At any given time, the state of the simulacrum need have no correspondence with the state of the original. This is why in the simulator frame diagram there is no connection shown between the original and the simulator; there is no communication, no shared phenomena. For example, a vehicle in a traffic simulation may crash but this certainly does not mean that any original has crashed.

In practical terms, correspondence between the simulacrum and the original is determined only by the requirements; there is no updating of correspondence at run-time.

That said, a simulation could form a partial frame in support of an information system *if* its purpose were to help derive reports about the actual state of the original *and* if there were some mechanism for ensuring an ongoing correspondence with the original. But this need not be the case.

7. Why isn't it a control problem?

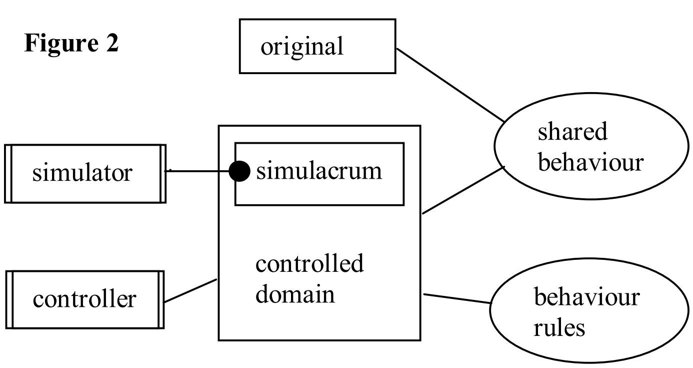
The control problem is characterised by the machine exerting control over some external reality.

In the case of the simulator, the external reality (the original) is not controlled (not even connected). There could, of course, be an associated control system for which the simulacrum *is the reality*. This could be depicted as below (figure 2), but the simulator and the controller are separate systems. This situation commonly occurs where a simulator is built in order to provide a test environment for a control system (without any risk of accident cause by defective control of the original).

However, a telling fact is that it is possible to simulate an autonomous domain (such as a weather system) which, by definition, cannot be controlled.

Even where control is possible, it is not *necessary* to have any planned control. Simulators for two or more systems could be interconnected such that each provides stimulation for the other and, without any requirements for controlled behaviour, they could be left to interact and their emergent behaviour can simply be observed. Certain simulations of evolving systems fit this pattern.

Figure 2



8. The fantasy problem frame

A further possibility is that, within a simulation, the original is imaginary; there is no real original to which reference can be made. This situation will occur with certain games, computer animations and the like.

An imaginary object, for example, a dragon, is endowed with certain behaviour and a machine is constructed that exhibits that behaviour. As before, it is quite possible that there will also be a connected system that provokes the simulation (with or without the aim of evoking a certain desired behaviour).

9. A final example: Traffic simulation

In order to provide test conditions for various traffic calming measures, it may be decided that a simulation of traffic is required. Since traffic consists of a number of independent vehicles, this equates to simulating vehicles or, to be precise, the vehicle/driver combination.

A major part of the development of the simulation is a study of the inherent behaviour of the original. It will be necessary to establish such matters as reaction time, acceleration/deceleration rates, look-ahead distance, follow-gap allowed at various speeds and so on. These will all exhibit a range across various vehicle/driver combinations and it will be desirable to establish probabilities across those ranges.

Clearly, to ignore the real world original would result in an impoverished, if not useless, model.

The next stage is to decide how closely the simulation should mimic the original. For example, it may be possible to establish a simple function that provides a sufficiently close approximation to the relationship between speed and follow-gap or speed and road width.

10. Conclusions and further work

Any development that allows closer fitting of problem frames to problem domains should prove advantageous and the proposed new frame appears to offer the possibility of achieving a better fit for certain types of problem.

Development of case studies will allow further investigation. In particular, the nature of the various domains can be more precisely determined and checklists (along the lines of those provided in [3], page 82 and in [7] page 103) for the investigation and documentation of the problem can be developed.

Ultimately, solution methods may be devised.

References

[1] M. A. Jackson, “Requirements & Specifications: A Lexicon of Software Practice, Principles and Prejudices”, Addison Wesley, 1995.

[2] D. Jackson and M. A. Jackson, “Problem Decomposition for Re-use”; *Software Engineering Journal*, Volume 11, Number 1 pages 19-30, January 1996.

[3] B. Kovitz, “Practical Software Requirements; A Manual of Content and Style”, Manning, 1999.

[4] D. Bjorner et al., “Michael Jackson’s Problem Frames: Towards Methodological Principles of Selecting and Applying Formal Software Development Techniques and Tools”; *ICFEM'97: Intl. Conf. on ``Formal Engineering Methods'', Hiroshima, Japan*, pages 263-271, Los Alamitos, CA, USA, 12-14 November 1997

[5] M. A. Jackson, “Problem Frames: Analyzing and Structuring Software Development Problems”, Addison Wesley, 2001.

[6] M. A. Jackson, “Problem Analysis Using Small Problem Frames”, *South African Computer Journal*, Volume 22, pages 47-60, March 1999

[7] I. K. Bray, “An Introduction to Requirements Engineering”, Addison Wesley, 2002.