Rule Migration: Exploring a design framework for modelling emergence in CA-like systems

Heather R. Turner and Susan Stepney

Department of Computer Science, University of York,
Heslington, York, YO10 5DD, UK.
turner|susan@cs.york.ac.uk

Abstract. We propose a framework for *engineering* emergent behaviour that allows system specification and design at the level of the emergence. We discuss how rule migration can be used to translate this high-level multi-layer design (*mobile process* model) into a simple *cellular automaton* model with only local rules, from which the behaviour can be described as emergent. In this paper, we use a case study in 2D to illustrate the process involved in deriving a migration rule for a simple random walk. To conclude, we discuss how the mobile process architecture may be extended to model systems with multiple levels of emergence.

1 Introduction

The design of emergent systems is a contentious topic among many researchers. If the behaviour is *designed* into a system, then by some definitions of emergence, it is not really emergent. Those of us who hope to be able to *engineer* emergence at some point in the future tend to employ definitions that focus on the properties of the emergent system, rather than the means by which it was designed.

We are interested in developing an architecture for modelling emergence in cellular-automaton-like systems. In [6] we discuss, from a refinement perspective, the difficulties faced when trying to engineer an emergent system. By definition, there is no simple translation between macro-level desired emergent behaviour, and the micro-level rules. We conclude that it should be possible, by choosing an appropriate representation architecture for the micro-level, to find ways of bridging the gap.

In [7], we consider a case study of blood platelets moving and aggregating in a 1D system. We introduce an architecture in which mobile processes (MPs) are used to *tag* emergent properties on a cellular grid, thus identifying them at an upper layer. Some of the operational logic can be transferred to the MPs, providing downward causation to the lower layer grid, thus making the lower layer simpler than one implemented as a pure cellular automaton (CA). We also introduce the idea of *rule migration*, as a means of translating the simpler MP models into pure CA models.

With such a framework, we can design a complicated model using the layers of the MP architecture. Emergent properties such as motion, and aggregation,

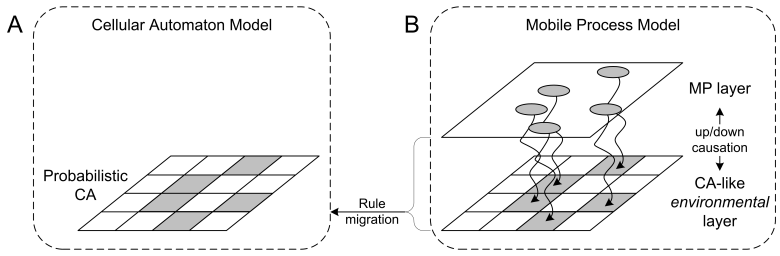

Fig. 1. An illustration of the relationship between (A) the CA model, and (B) the MP model.

can be described and observed at the higher level. The rules from the MP model are then translated by *rule migration* into low level CA rules, as illustrated in figure 1. These CA rules are unlikely to encode explicitly the higher level properties, and as such the properties can be considered *emergent properties* of the CA model.

In this paper, we extend the work on rule migration into 2D. We use a similar platelet case study system. It is important to ensure that our work generalises into higher dimensions, particularly because most potential real world applications exist in 2D and 3D environments.

We constrain our model to ensure that objects within the model do not need to know their absolute position in their environment. Relative location is important, such as proximity to other objects, but no knowledge of absolute location, a global property, should be necessary. The layers of the MP architecture are particularly suited to the enforcement of such constraints. By placing restrictions on the communication protocols, we can ensure that no global position information is passed upwards from the lower layer. Intermediate layers can be used to abstract away many types of detail, and help us to build a structure where the information at each layer is restricted to only that which is necessary, and local at its own scale.

For the purposes of this study, we take an abstract view of blood platelets as objects that can move through the blood vessel and form clots to repair damaged tissue. In the following section, we present our designs for two different platelet models of a simple case study system behaviour. One simulation is implemented as a low level CA, and the other is designed in terms of the high level processes. We explore the relationship between the two models, adjusting each of them, as necessary, until they are functionally equivalent. Once the two models are equivalent, we have a direct mapping (*migration rule*) between the implementations at the two levels, and thus a means of translating high level designs involving this behaviour into a low-level implementation. In Future Work, we discuss our plans to explore the mapping between other *general* behaviours implemented at the different levels. We hope eventually to build a dictionary of these migration rules for converting high level designs into low level architectures.

2 Case Study

There are two major considerations in the design of the platelet model: how to simulate movement through the channel, and how to generate platelet aggregation. Each of these can be reduced to simpler scenarios by capturing a more general case. As an example, in the more sophisticated model, platelets are expected to travel primarily in the direction of flow in the channel, by following a biased random walk. In the simpler case, we allow motion in any of the four compass directions on the square grid, each chosen with equal probability. The situation is slightly complicated by the presence of multiple platelets; these act as obstacles to movement, and can compete for environmental real-estate.

Breaking the model into smaller, simpler sub-properties has obvious positive implications when it comes to the design of a system. We can explore each one independently, choosing a level of abstraction that simplifies the design process. As the model develops, we can add more detail, until all aspects of each sub-property are adequately captured. The rules for the completed sub-properties are combined to provide the rules for the model as a whole.

In the subsections that follow, we discuss two different, but equivalent, implementations of the simple random walk by a platelet on a 2D square grid.

2.1 Cellular Automaton Model

The probabilistic nature of the random walk cannot easily be implemented in a simple, deterministic CA. The CA model we present (see figure 1A) adheres to many of the properties that we consider to be the most important, defining characteristics of a CA. Space is modelled by a regular, square grid of sites that update in synchrony at discrete time intervals. Each site has an associated state, which in this case is composed of two variables that contribute, in total, nine possible unique states. The update rules, which define how the sites change state based on their own state, and those of their neighbours, are defined for the von Neumann neighbourhood of size 5. The automaton is homogeneous, that is, all the sites in the grid use the same rules. In contrast to the usual definition of a CA, the rules are probabilistic. They are also quite complicated. It is unnecessary to produce a look-up table for the 60,000 or so neighbourhood configurations; in the following sections, the rule is described as a compressed algorithm that could be used to generate the lookup table.

Returning to the design itself, we have a 2D grid with *occupied* and *empty* CA sites. The occupied sites, representing the platelets, have the ability to choose their direction of motion, based on the availability of empty sites in their neighbourhood. CA sites can update only their own state, and so have no direct means of indicating to the chosen destination site that it has been selected. We use a series of *passes* in the CA rule to allow the sites to indicate their preference, verify selection, and perform the necessary state updates.

We now describe in detail how this strategy can be employed with our platelet model. Figure 2 shows a possible initial configuration for the CA grid. Each site's state has two parts, a **type**, and a **direction**. The **type** can be *black*, *grey*, or

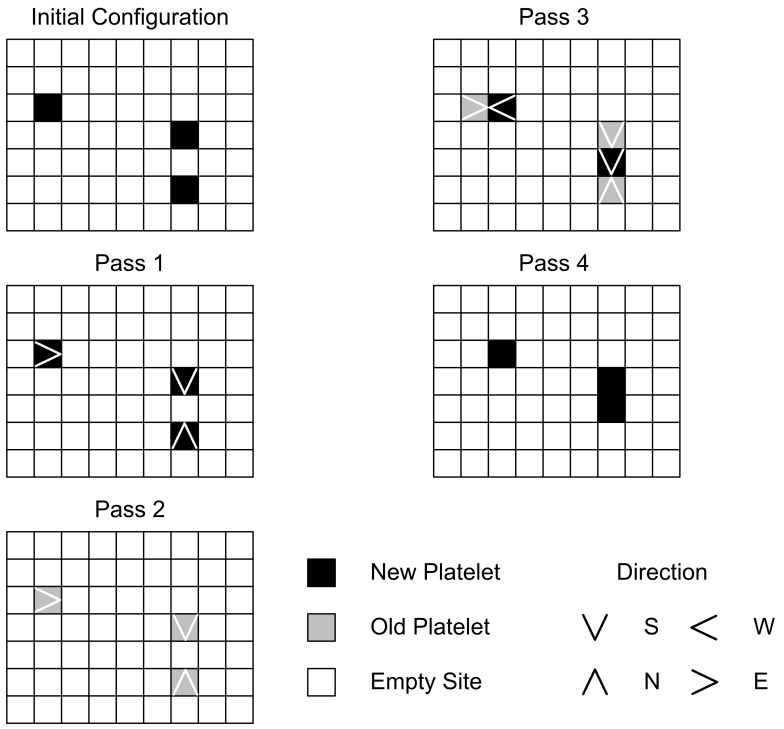

Fig. 2. This diagram shows one time step in the evolution of the CA. Four different passes update the states of the grid sites, comprising **type**, indicated by the colour of the grid square, and **direction**, identified by the arrows. By applying the simple local rules, we simulate observable random motion.

white, to indicate new platelet, old platelet, or empty site. The **direction** can take the values N, S, E, W, and corresponds to neighbouring sites in each of the compass positions. The value of **direction** is relevant only when **type** is indicating the presence of a platelet. The different snapshots, or passes, identify different synchronisation phases of the CA rule.

At the beginning of each time step, a platelet is given the opportunity to try to move. It first evaluates the states of the neighbouring sites, to establish a *current* view of its surroundings. The platelet may move only to a cell that is empty during this evaluation. Having determined the availability of empty sites in the four neighbouring directions, the platelet selects one, with uniform probability, if any are available. A platelet indicates its chosen destination by setting its direction state appropriately. This is the situation shown in figure 2 as Pass 1.

The platelet cannot just *move* in the chosen direction, as that would involve updating the state of its neighbouring site. In Pass 2, the platelets change their **type** to *grey* to indicate that they are ready to move.

In Pass 3, the *white* (empty) sites evaluate their neighbourhood, and determine whether any platelets want to move into them. They evaluate both the **type**, and **direction** of their neighbouring sites, counting any prospective new occupants. They select with uniform probability any neighbouring platelet wanting to move, and set their **type** to *black*, and **direction** to point at the selected platelet. Otherwise the site is unchanged.

In Pass 4 the *grey* sites are updated to indicate whether the platelet successfully moved, or whether it failed at this particular time step. Each grey site checks that it has one *black* neighbour with its **direction** pointing back at the *grey* site. If so, the *grey* site changes its **type** to *white*, thus completing the move. If the neighbouring states indicate that the move is not accepted, the *grey* site changes back to *black*; it can attempt to move again in the next time step. The direction is no longer relevant.

The four pass structure of this algorithm is used for clarity; it is equivalent to one single rule that includes an extra state that identifies which pass the system is in.

The "movement" of the platelet is an emergent property of this CA. Nothing has actually moved: sites merely change state. The rules have to be carefully designed to correspond to movement and conservation of platelets.

2.2 Mobile Process Model

The mobile process (MP) model for this platelet example (see figure 1B) is a 2D extension of the 1D model presented in [7]. A cellular array forms the lower layer, and certain sites are "tagged" with mobile processes, residing at an upper layer, that represent the presence of a platelet. Some of the logic resides in the lower cellular layer, which we call the environmental layer after [2], and some is captured by the MPs, making the movement easier to model.

An important aspect of the two layer architecture is its ability to abstract away detail. For example, at the mobile upper layer, processes have no notion of *absolute* location. Communication channels act like sensors into the environment, providing a means by which the MP can sense and interact with its neighbourhood, without actually knowing *where* it is. The environment contains state information indicating whether connections to upper layer processes exist at each site. The topology of this architecture is illustrated in figure 3. In this model, each MP is connected to precisely one site at the environmental layer, so there is a direct equivalence between sites representing platelets and processes representing platelets. At the environmental layer, there is an absolute frame of reference in the coordinates of the grid sites. Each site is directly connected to its four immediate neighbours by two communication channels, and it has two further channels by which it can connect to any MPs at the mobile layer. Upper layer MPs send information requests to their respective cellular site. The

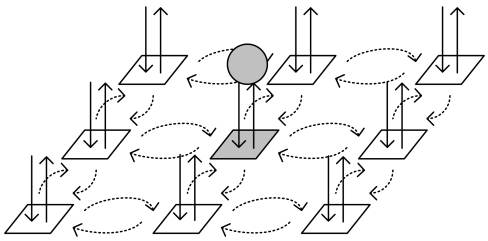

Fig. 3. A schematic illustration of the MP architecture used in the platelet model. The square grid represents the lower environmental layer, and arrows indicate the communication channels. The grey circle is a MP at the upper layer, and the grey square is a reflection of the state information that records its presence at the lower layer.

communication protocol permits the transmission of state information associated with neighbouring sites. In addition, the channels facilitate movement of the MP when it transfers from one site to a neighbouring site. For an example of how this movement might be implemented, see Appendix A.

In the platelet algorithm, platelets are represented by MPs at the upper layer. At each iteration of the model, the platelet tries to move. It submits a request to its environmental site to provide information regarding its neighbourhood. For this particular implementation, we choose to hide all location and direction information from the MP, but allow it to implement the probabilistic selection. To do this, the server counts the available neighbouring sites and reports back to the MP. The MP then chooses one with uniform probability, and sends the 'move' request to its environmental site. By a mechanism similar to that described in Appendix A, the MP attempts to move to the chosen neighbouring environmental site. If the movement is unsuccessful, because, for example, another process is chosen to move to the site, the MP's site re-evaluates the neighbourhood, and permits the MP to choose another direction. At this point, information about platelet locations in the two layers is no longer equivalent, the environmental layer must update. Sites that have lost connections become *empty*, whilst those that find their channels newly connected become *occupied*.

3 Rule Migration

To achieve our goal of providing a general mechanism for translation between the more simply definable MP architecture, and a pure CA-like system, we first consider some specific examples. As we have shown in the previous sections, we have independently designed an MP model, and a CA model, for our simple platelet example.

The next step is to make the two models functionally equivalent, that is produce the same result when given the same initial configuration. This process can be seen as an alignment operation, as we illustrate schematically in figure

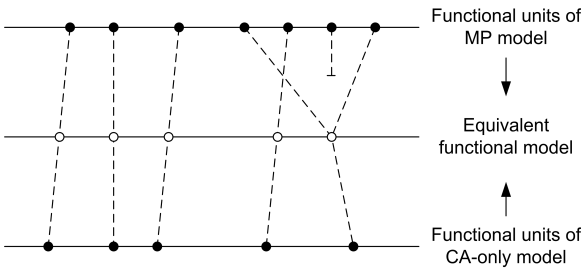

Fig. 4. This diagram is a schematic for the alignment process.

4. We work from the top (MP model) downwards, and from the bottom (CA model) upwards, and make necessary modifications and adjustments until the two models can 'meet in the middle', and be equivalent.

In the MP model, the platelet appears to have more identity, and with this, it seems logical, and simpler, to offer it more freedom and choice. When two platelets attempt to move to the same location, one is prevented from moving. In the MP model, it is reasonable to allow it to "have a another go", and try another direction, at least until it runs out of options. The simplicity is achieved because the MP and its environmental site can freely exchange information without requiring the complicated state changes that would be required in the CA model. In the MP architecture, when the CA site fails to fulfil a move request, it can send a message to the MP, and ask it to pick a different direction. The MP can reply with a new choice, and another move is attempted.

The pure CA model does not have this richer functionality, as it does not fit so naturally with the synchronous update model. In the process of aligning these two models, it is simpler to scale down the functionality of this more complicated MP model. The *second-chance* for the MP is removed from the model, and the alignment is complete. Figure 5 is the result of this process, and shows how the equivalent functionality is achieved by different means in each case. The details are discussed below. To further clarify what we consider to be equivalence, the configuration of the environmental layer of the MP model, and of the CA model, should be equivalent at the beginning and end of each iteration. In the intermediate processing, the CA model makes use of additional state information, and the MP model sends messages via its communication channels.

In figure 5, we show the finished product of the alignment process. The two models are functionally equivalent, and we show where the different sub-units are implemented in each case. In the diagram, numbered arrows link the parts of the algorithm that perform equivalent operations. These are described in more detail below:

1. Platelet environment is evaluated.
2. Platelet selects a direction in which to move.

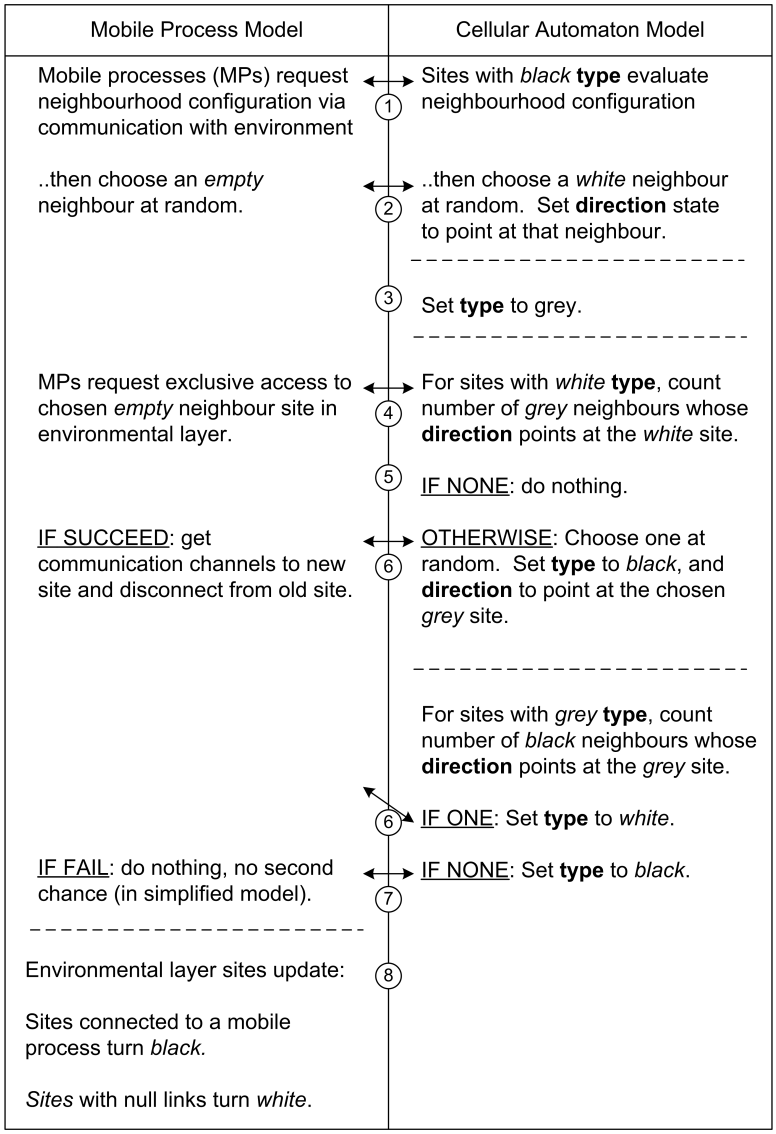

Fig. 5. The finished product of the alignment process. Numbered arrows show where the program sub-units are equivalent.

3. In the CA model the platelets that are trying to move change state so that they can be distinguished in later passes from platelets that have successfully moved.
4. Platelet ascertains whether it was successful in its request to move.
5. In the CA, we evaluate all empty cells in the grid to determine whether any platelets want to move into them. Many empty cells will not be near platelets and so should

not change state. In the MP model there is no equivalent as computation is carried out only in the locality of the MPs.

6. Much of the CA algorithm is covered in one step of the MP model. This step is the *actual* movement, and the communication channels are rearranged at this point. In the CA, we have the simple case where there is no competition for the destination site and the movement effectively occurs (although the source cell must be updated later, as identified by the second arrow marked with a 6). In the more complicated case where there is a conflict, the destination site can detect that more than one platelet is trying to move into it, and it has to choose one at random (this is not dissimilar from the random element to the process scheduling that determines which MP moves first). The successful platelet *moves* at this point.
7. The platelet did not get to its destination before it was occupied by another. The platelet does not move. In the CA case, the platelet location from the first pass is reactivated.
8. The environmental layer of the MP model is updated to reflect the new locations of the MPs.

Although this correspondence needs to be formalised, we hypothesise that the functional equivalences described above can be generalised. By considering more examples, and through further experimentation, we propose to find patterns of operation in the MP model that correspond consistently to other patterns of behaviour in an equivalent CA model. If we can build a dictionary of such correspondences (migration rules), we will have accomplished a major step towards an architecture for exploring emergence in CA-like systems.

4 Future Work

If we can describe two levels in an emergence hierarchy in terms of our MP architecture, then by applying our migration rules, we can build a low level CA that displays emergent behaviour. The MP model can include moving, aggregating objects, and these can be programmed explicitly. After migration, the rules for the CA model will be unlikely to have such transparency. It will be difficult to discern the system behaviour from the CA rules, and the resulting behaviour will be difficult to explain in terms of the low level CA rules; it will require description in a higher-level language. We would argue that although there is much controversy surrounding the definition of *emergence*, the behaviour just described would be classified as emergent.

Extending this idea, we plan to develop migration rules between MP layers. Ultimately we want to be able to model systems with multiple levels of emergence. By describing these systems in terms of MPs at the highest level, we can migrate the rules to produce an MP model at a lower level. The rules from this model can then be migrated downwards, until eventually the rules are migrated to a CA model. The MP to CA rule migration is the special case, while the more general MP to MP migration rules will permit us to model, at least in principle, an arbitrary hierarchy of emergence.

Our next step is to explore more examples, with wider breadth. We will generalise the results here, and construct the migration rules. In addition, we

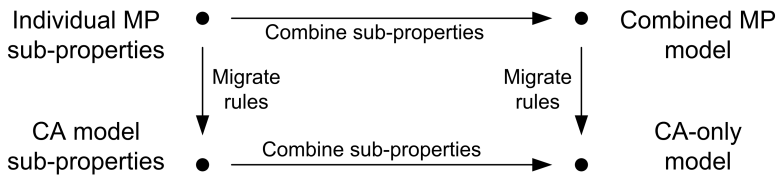

Fig. 6. Diagram illustrating the hypothesised commutative nature of the migration and combination procedures.

will explore the combination of simple sub-properties such as movement and aggregation, into a more complete model.

We expect it to be possible during the design process to either combine the MP sub-properties and then migrate the rules to a CA-only model, or to perform the rule migration on the individual sub-properties, before combining them. We hope that properly designed rules will have a useful degree of commutativity (see figure 6).

Once we have satisfactory results with migration between the MP model and the CA model, we will consider the addition of upper layers. In our case study system, this is likely to be the movement of aggregations of platelets. We will design rules at the level of the aggregation, and at the level of individual platelet MPs, and then formulate the appropriate migration rules.

5 Related Work

The hierarchical structure of emergence has been identified and discussed by many researchers. A recent literature survey by Fernando [3] provides a useful introduction to both the theoretical and experimental research in this area. He focusses on *hierarchical complexity*, and a study of the modelling constraints that are necessary to evolve this structure.

There appears to be a limited amount of material available concerning higher levels of emergence. Mayer and Rasmussen [5] also use a variation on a CA in their model. They call the architecture a *Lattice Molecular Automaton*. The state and dynamics of this model are highly complicated, and the rules are derived from the laws of physics. The authors claim that this detail is necessary to achieve the higher order structures. We intend to explore this similar work in more detail to discover the means by which the system is designed. It would be interesting to discover if we can derive a similar model from the top down, using our MP architecture, and rule migration.

Another researcher who uses a multi-layer architecture is Capcarrere [2]. His model, Phuon, is constructed from two layers, the active cellular layer, and a passive environmental layer. These layers seem to correspond quite closely to the layers in our MP architecture, and we have chosen to name our lower layer in the same way for consistency. That model particularly considers the

simulation of growth and development, but in addition incorporates motion at the cellular layer. We will explore growth and development in the context of our own architecture and the migration rules.

One of the initial problems we encountered in our case study was how to model movement using just CA rules. Other authors have considered this problem, although the explicit detail of how the movement is achieved in terms of the local rules is not always clear. Goel and Thompson [4] use movable finite automata in a specific biological case study. Wolfram [8] considers motion in a CA at the macro-level in terms of its ability to model fluid dynamics. That paper concentrates heavily on the mathematics of the macro-level behaviour, and the underlying rules for movement are not entirely clear.

6 Acknowledgements

This research is funded by EPSRC and BAE Systems as part of a CASE Studentship.

References

1. Barnes, F.R.M., Welch, P.H.: Communicating mobile processes. In East, I.R., Duce, D., Green, M., Martin, J.M.R., Welch, P.H. (eds), Communicating Process Architectures. Amsterdam, IOS Press, 2004.
2. Capcarrere, M.S.: An evolving ontogenetic cellular system for better adaptiveness. BioSystems **76** 177–189, 2004.
3. Fernando, C.: On the Evolution of Hierarchical Levels of Organisation. Unpublished DPhil literature survey, University of Sussex. Available from: http://www.informatics.susx.ac.uk/users/ctf20/end_of_year_report_sc.ps, accessed May 2005.
4. Goel, N.S., Thompson, R.L.: Movable Finite Automata. In Langton, C. (ed), Artificial Life, SFI Studies in the Sciences of Complexity. Addison-Wesley, 1988.
5. Mayer, B., Rasmussen, S.: Self-Reproduction of Dynamical Hierarchies in Chemical Systems. In Adami, C., Belew, R.K., Kitano, H., Taylor, C. (eds), Artificial Life VI: proceedings of the Sixth International Conference on Artificial Life. MIT Press, 1998.
6. Polack, F.A.C., Stepney, S.: Emergent properties do not refine. In REFINE 2005, BCS-FACS Refinement Workshop, ENTCS. April 2005.
7. Polack, F.A.C., Stepney, S., Turner, H.R., Welch, P.H., Barnes, F.R.M.: An Architecture for modelling emergence in CA-like systems. In ECAL 2005, LNCS. Springer, 2005.
8. Wolfram, S.: Cellular Automaton Fluids: Basic Theory. In Wolfram, S., Cellular Automata and Complexity: collected papers. Addison-Wesley, 1996.

A Implementing Movement with Mobile Processes

Figure 7 illustrates how we might implement the movement of mobile processes around the grid whilst hiding explicit location information from the MP itself.

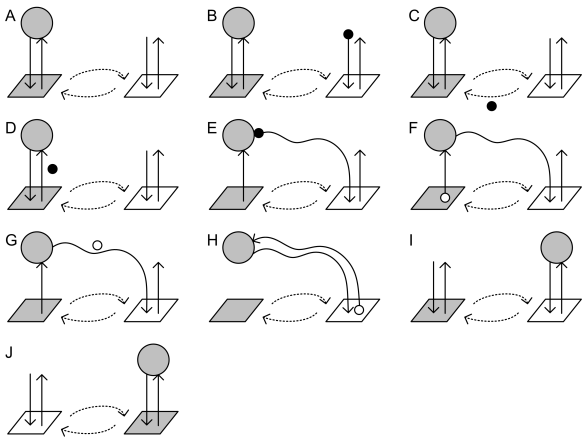

Fig. 7. An illustration of the series of steps (A-J) that implant the movement of a MP within the proposed architecture. Each step shows the two neighbouring environmental layer sites (squares) and the single MP (circle). Communication channels are represented by arrows, and small circles represent the mobile *channel ends*.

Each channel pair consists of a *send* channel and a *receive* channel (A). When an MP decides to move, it evaluates the local neighbourhood to determine the available space; this proceeds via an interaction with the lower environmental layer. The MP itself might know only that it has, for example, three options, and can choose one at random, without actually knowing which direction it corresponds to. The movement is all handled by the lower layer. When an appropriate destination site is chosen, the MP requests the new communication channels, and relinquishes the old ones. This procedure can be achieved by the MP's environmental site requesting that its appropriate neighbouring site should send the *source* end of its *receive* channel to the MP (B,C). When the MP receives this, it connects to the channel, thus releasing one connection to the site below (D-F). This new channel is then used to send the *source* end of the MP's *receive* channel to the new site (G). The old site is now fully disconnected, while the MP now has a two-way communication channel to the new server site (H). At this point, the state information at the environmental layer no longer accurately represents the configuration of MPs at the upper layer (I). The inter-layer connections for each site are examined. Where channels are not connected, the states are set to *empty*, but where active communication channels exist, the state is set to *occupied* (J).

The multi-layer model is designed to be implemented in occam-π, as this allows for an efficient parallel implementation, and incorporates mobile processes, and mobile channels [1]. As a result of the mobile processes executing in parallel, there is a possibility that multiple MPs will choose the same destination site. Only one MP succeeds in its movement, while other attempts to move to that site fail.