Towards Simple and Effective Formal Methods for Intelligent Environments

Martin Henson*, James Dooley†*, Abdullah Al Malaise Al Ghamdi†* and Luke Whittington†*
*Department of Computer Science, University of Essex, UK. E-mail: hensm@essex.ac.uk
†Faculty of Computing and Information Technology, King Abdulaziz University, KSA

Abstract—In this paper we motivate and illustrate the use of bigraphs as a formal framework and methodology for the description, design and analysis of intelligent environment systems. Through a series of examples, we provide an overview of bigraphs, their composition, their evolution under reaction rules, and their refinement. We argue that bigraphs offer several advantages: first, they are intuitive and lie close to the topic of investigation; second, they are relatively simple to understand and deploy (in contrast to the systems they may analyse); third, they offer a means to tame complexity through multiple description at different levels of abstraction; fourth, and finally, the system itself can be usefully used without having to engage with its mathematical foundations.

I. INTRODUCTION

The backdrop to this paper is uncontroversial: intelligent environments are increasingly complex asynchronous, concurrent, dynamic, distributed systems – and they are not about to become less so. The need for a precise framework, a formal method, within which to describe them, design them, refine them, verify them, analyse them, transform them, and study them – in a way that is sufficiently robust for precise shared understanding – is evident. For example, in [1] we captured something of the complexity of supporting a world populated by a dynamic collection of distributed ubiquitous computing spaces – supported on a layering of abstract spaces upon levels of peers, entities, and underlying devices. In [2] a much more comprehensive attempt to capture aspects of an information-centric architecture for describing ubiquitous e-artefacts was undertaken. What is interesting in much of this work, is the intuitive appeal of the diagrams outlining the informal arrangements within, say, a world of spaces, and the somewhat cumbersome set/logico-theoretic mathematical machinery that does service to capture the same arrangements more formally. This conceptual gap is the first point we want to highlight (and, indeed, bridge).

One does not buy and learn to fly a helicopter in order to go out to purchase a newspaper from the corner shop. On the other hand one does not attempt to mow the lawn with a pair of scissors: one employs the right tool for the right task. Really good tools are those that are themselves simple, but are capable of delivering results of great complexity – imagine for example the contrast between a chisel and an ornate wood carving. That is what we should aim for as a contrast between a formal method and the applications it analyses or describes; that is, a method which is, in itself, approachable and graspable, but capable of describing and variously analysing systems of enormous size and complexity. This notion of simple but effective methods is the second point we want to highlight.

It is not necessary to understand aerodynamics in order to fly an aeroplane and one does not need to understand the thermodynamics of combustion to drive a car. We adopt an appropriate perspective sufficient for understanding (or operating) the system in question. There are two points we wish to make here. Our third point is that large complex intelligent environments can and should – whether informally or formally – be understood at a variety of levels and from various perspectives. If we do want to provide a formal understanding, the methods we use should respect that: it should be possible to hide or reveal details of a given system. This is important for general comprehensibility, but it is also important for handling large and complex systems, as it offers us the ability to zone in at the right level of detail. Our fourth, and final point, concerns perspectives on the method itself. Think now of the method itself as an aeroplane or car – it should be possible to drive the method, to use it usefully in practical contexts, without having to know every last detail of its underlying mathematical foundation. In this paper, the approach we take using bigraphs has a mathematical model in category theory – a branch of abstract algebra. In fact, very little category theory is needed, but that is not really the point. If we are to succeed in proselytising the use of formal methods as a useful and creative adjunct in a major applications-focused research area (such as intelligent environments), we will not get very far if the prerequisites involve seriously advanced mathematics.

The aim of this paper is to argue that a particular methodology (bigraphs) is approachable and appropriate for applications in intelligent environments; to encourage and raise confidence; and to begin to build a community of interest to explore the topic in detail. Our contribution here is to show how scenarios from intelligent environments can be rendered in the framework, and how recent advances in the area (in particular, refinement) can in principle be used to tame the potentially overwhelming complexity inherent in intelligent environments.

There are, of course, other approaches and associated tools, several of which have been deployed in interesting ways. For example, UPPAAL and TCTL [3] have been used in the analysis of AmI applications [4] and SPIN [5] in an AAL application [6]. Comparison with these is beyond the scope of this paper – but such an analysis will be important topic for future research.
II. PRELIMINARIES

The structures we are proposing to illustrate are called bigraphs. The definitive account appears in [7], though the ideas have been around for much longer. As our fourth aim describes, there is a perfectly approachable, indeed in this case diagrammatic, representation that is surprisingly close to the applications’ area itself. Let’s start with the simplest possible example. Imagine we have some abstract notion of space – within which perhaps user and computational activity resides (and is contained). But at this stage we do not even want to specify such details. This is Figure 1. The ellipse is a node and the dashed surround a region. We can use regions to determine and control locality. Nodes may be physical structures (maybe a building or a classroom), or entities (such as computers, keys, teapots), or a virtual entity (possible a password, or a software agent), or a process (perhaps a command or enquiry) or many more potential uses. An important point is that the diagram translates seamlessly into more and more formal mathematical descriptions that underpin the bigraph framework. The seamless nature of this translation implies that the diagram itself is as close as we may usually need to a perfectly precise formal description. We will come back to this, and indeed dip our toe just a little into other ways in which such diagrams might be understood.

Before we move on we should explain the label (S) that appears in the node. We shall typically want to lay out a set of types of entities that appear in our system description – and we can give each type a name (a label). We call that set a signature – it comprises just a little more than the set of labels that specify the entities we want to deal with – but we will come back to that in a moment.

Let’s look at a marginally more complicated description. In Figure 2 we have a world of spaces. There are a number of spaces and they are contained within a world. We might (and will) think of each space as accommodating a local-area network and the world as supplying the Internet – or at least a wide-area network. None of that is implied, though, by the diagram.

Figure 3 shows a space again, but this time there are members, shown by the square nodes. All reside within the region, which means that they hang around together – they are all local to one another (we say they are proximal), but two are inside the space and one is outside.

In Figure 4, the member that is outside the space is also in a different region – the member does not travel, as it were, alongside the space, and may be physically separated from the space (we say they are distal).

The framework allows these diagrams to reconfigure, according to reaction rules. Figure 5 shows a reaction rule that allows a proximal (that is, within the same region) member that is outside a space, to enter.

Such a reaction rule would allow the system shown in Figure 3 to evolve into one in which all three members were inside the space. On the other hand it does not apply to the system state shown in figure 4.

Now something new has appeared in the reaction rule: the dashed area that lies within the space. We call this a site – it is somewhat like an internal region. Its presence in the reaction rule is crucial, since without it the rule would only apply when
the space was otherwise unoccupied. With the site in place
the two members that are already present in the space can be
matched against the site.

What we have illustrated so far is spatial organisation and,
indeed, motion. Through nesting and occupancy, entities can
be contained; they can be juxtaposed (proximally) within a
region; and they can be separated (distally) between regions.
Motion of entities can be captured through evolution of
system description states via reaction rules – and the spatial
relationships adjusted as a result.

We should now move onto connection and communication,
because at present all our entities are entirely isolated from one
another. Consider Figure 6. Our space now contains devices
and members – and there are connections shown between some
of the entities.

Let us return to the idea of a signature for our entities.
For each label (we have three of them), we assign an arity
– this is the number of connecting ports that entities of that
label possess. In our case, spaces have arity 1, members arity
1 (though we will adjust this later), and devices have arity
2. There are a few notational matters that are worth listing
at this point. First, there is no significance to the shapes of
the nodes. We use different shapes like labels. Indeed, if we
use the shapes consistently we could dispense with the labels
and use those shapes as a surrogate). Second, when ports are
not connected we may either omit them entirely, or show
them explicitly as disconnected. In Figure 6 the right-most
member is not attached to a device, and we are showing the
port with a hanging disconnected link. Typically ports will
have various meanings. In our example, one port is a user
interface (connects to a member) and the other is a network
connection. This connects to the single port of the space, which
is supposed to be its local-area network. We can distinguish
multiple ports in various ways. In our cases, the user-interface
port is on the top left side of the device and the network
port is at the very bottom of the device. Notice too that
the network port of the space has two links, to the two connected
devices. A port may have zero or more links. What do the links
mean? Again, the answer is a varied as for the nodes – they
may be real connections, or referants, or indicate pathways,
and many more. We will illustrate the variety as we proceed
– but even here we have some probably variation: the links
between devices and space may be ethernet connections, or
they may be wireless connections. On the other hand the
connection between the members of the space and their device
may indicate a login relationship, or interaction via a keyboard,
and so on.

We promised to look a little underneath the mathematical
hood of the diagrams – the next level of underlying description
is to understand the diagrams as a pair of graphs. Figure 7
shows the system description of Figure 6 in this form. One,
in this case, is a tree (we will see a little more complexity
later) representing the spatial arrangements – containment in
the diagram is captured by the parent-child relationship in the
tree. This is called the place graph. The other is a hyper-
graph: all the nodes appear at the same level and the links
are indicated by hyper-edges. A hyper-graph is distinguished
from a graph by the multi-edges (as in this example). This is
called the link graph.

Let us return to the idea of a user (member) entering the
space. We might feel our description was too permissive.
A member can simply enter the space, it appears, without
let or hindrance. Here is a subtlety: there is a difference
between an implemented system and a specification. Certainly,
an implementation of our system would indeed permit free
access to the space. But this is because, once a system is
realised, the real world supplies all other possible contexts,
entities, protocols, and so on – and these are, because they
are not included in the implementation, specifically ruled
out. For example, such an implementation rules out a login
or key access protocol, even though such things exist in
the real world. In the abstract world of the specification,
however, nothing exists other than what is stipulated. Thus, the
specification does not permit free access, because there is no
conception (in this stripped down specification) between free
and managed access. In fact, the specification simply states
that users may enter the room – and this is something we do
want to happen in an implementation (in some shape or form –
probably conditioned) or that implementation would not have
the liveness properties we would insist upon. If we do want to talk about protocols for access in our specification, then we need to include those explicitly. Figure 8 shows access to the space that requires (some notion of) a key. What we have here is another reaction rule for entry.

Since the port of the space is a network connection, the key is probably electronic (we could of course add additional ports to the space to represent a physical key if we desired). Indeed, the reaction rule shows the key disappearing once the user has entered the space. Perhaps it is a use-once pass-code.

Before a user can enter the room, then, they need to get hold of a key. Figure 9 shows a reaction rule that shows a user (who is not connected to a key) getting hold of a key (which is not held by anyone else).

Note the link from the key that hangs upwards. This is the network connection to a space. When matching the left-side of the rule against a particular system state, that link may (or may not) be connected to a particular space in that system. It is a convention to place those links which are to be received by a context at the top of the diagram. Such links are analogous to regions. Regions (like the one surrounding each side of the rule) are also received by the context into which they are placed.

Of course, we may want to adopt an entirely different approach. For example, imagine a multi-use key that persists on use. Figure 10 shows a slightly different reaction rule to cover such a case.

Here we need to take into account the possibility that others are already sharing this key (maybe they all know the passcode). Given this we can provide a reaction rule (Figure 11) for entry which leaves the key in place so that others’ use of it remains possible.

We have now looked at regions and the upward free links that are received by a context into which the diagram is embedded. We have also seen sites – the complement of regions – that are contexts that receive diagrams. The one missing concept are the complements to the upward links – which we unsurprisingly draw downwards. These are links that are analogous to sites; they are the link contexts that receive diagrams. To illustrate this, consider Figure 12. Here we see a reaction rule that connects a device to its space’s intranet. If this is the only reaction rule that establishes such a connection, then only devices that a user has adopted can connect to the network.

Note the downward facing network link from the space.
We have to allow this rule to match a system description state when there are other users and devices in the space – and in particular, we must allow those devices to be connected to the space’s intranet. Note that this link will connect to corresponding links in the diagram the fits into the site. That certainly handles the intranet. However, there may be a problem here. The entire diagram (the region in which it sits) may itself be received by a larger context (maybe a site in a world of spaces like that shown in Figure 2). The spaces may be connected together by an internet. At the moment our reaction rule could not apply to a space that is so connected. The solution is not so problematic: all we need do is add an upward facing link which will connect as the entire space (region in which the space sits) is embedded within a larger system (possibly a world of spaces). Within that the upward link will connect to a suitable downward link establishing the internet connection. This is essentially illustrated later in Figures 15 and 16 (look at the links with name $y$).

III. COMPOSING SYSTEMS

In the course of the previous section we have talked, informally, about how reaction rules match against particular sub-diagrams and evolve the system state description by replacement. In order to provide a comprehensive description we would have to delve several levels deeper into the underlying mathematical foundations, which is neither our aim here, nor is it necessary – since [7] provides all the details one could need in this regard. We will content ourselves with a look at how diagrams come apart at the level of the diagrams themselves – and we do this by exploring the converse notion: how diagrams compose. This will clarify any difficulties over the roles of regions, outer links (those facing upwards), sites, and inner links (those facing downwards).

Consider Figure 13. Here we have a space with a single member and a site where other entities might be embedded. There is an inner link, named $x$, that may be connected into whatever is received by the site. We also have two users both of which are connected to devices. These four entities are proximal – that is, they exist in the same region and will remain together in that sense. One of the devices is definitely disconnected from the network; the other is potentially connected. The diagram show an outer link, named $x$, that will connect with any similarly named link should be diagram be received into a site. To complete our terminology, we call $x$ on the outer link an outer name and when on an inner link an inner name. Note that the region and site are also named – by numbers.

The idea is that we can sensibly place the lower diagram’s region into the site within the upper diagram. Obviously not every diagram would sensibly fit inside another in this way: the interfaces somehow agree (as these do). We will come back to that later. When we compose these two diagrams, the result is captured by Figure 14.

Note, in particular, how the inner link of the upper diagram of Figure 13 has connected to the outer link of the lower diagram.

Figure 15 is a second example of composition, showing a more complex arrangement with regions and sites. In this case we have two spaces within different regions. Consequently, even though there are connection between the spaces (internet) they are not necessarily proximal and may be a continent apart. The upper diagram has two sites and an inner link that will (presumably) provide the live network to support communication inter-spaces. Once again, this composition makes sense. The matching conditions concern both places and links – the two regions match up, as do the inner and outer names. The result of plugging the two space regions into the world of spaces is given in Figure 16.

Before we move on, let us look at the underlying place and link graphs of the lower diagram of Figure 16. This is given in Figure 17.

As we saw before, the link graph is a hyper-graph with multi-edges. In this case, however, we see that the place graph
is actually a forest (a collection of trees). Finally, we can be a bit more precise about the plugging conditions – that is, the appropriate conditions for composing diagrams. It may seem a little perverse, but it is useful to consider each diagram like a transformer – with the sites plus inner names (the inner face) as an input, and the regions plus outer names (the outer face) as the output. Then, treating every number \( n \) as the set of numbers less than \( n \) – that is, for example, \( 3 \) as \( \{0, 1, 2\} \) – we can express the type of a face (inner or outer) of a diagram as a pair comprising a number (the regions or sites) and a set of names (the inner or outer names). Consider Figure 15. The lower diagram has no sites and no inner links but 2 (0 and 1) regions and the outer link \( y \). Thus its type is written \( (0, \{\}\) \( \rightarrow (2, \{y\}) \). On the other hand the upper diagram has two sites and the inner link \( y \), and a single region (labelled 0) but no outer links. Its type us thus \( (2, \{y\}) \rightarrow (1, \{\}) \). Note that the result type of the lower diagram and in input type of the upper diagram are the same. This is why the composition works. Not surprisingly, the type of the resulting diagram (Figure 16) is \( (0, \{\}) \rightarrow (1, \{\}) \). This idea, of treating diagrams a bit like functions, is in fact essentially the fundamental mathematical interpretation of bigraphs in category theory.

IV. NON-DETERMINISM, TRACES AND REFINEMENT

It will have been clear that reaction rules may match a given configuration in several ways. Thus, given an initial system description, there will be different ways in which it might evolve. Consider Figure 18. There are three ways in which the reaction rule of Figure 12 may apply to this.

We can capture the possible evolution of this system under the reaction rule of Figure 12 as a tree. Please note that this is very different from the trees so far which are alternative means for describing the place structure of a diagram. In Figure 19 the nodes of the tree are entire diagrams and paths through the tree are successive applications of the reaction rule. The labels in the nodes indicate which of the three devices has just connected to the intranet.

From the tree we can collect together what are called the traces of the evolution. Essentially these are all paths and partial paths through the tree (including the empty path). There are therefore, in this tree, 17 such traces. We use this set of traces as the mathematical model of the dynamic history of the system.

At the outset of the paper we highlighted the importance of being able to provide multiple perspectives on a system at different levels of abstraction. We mentioned this again in the discussion following Figure 7. We should be able, one supposes, to say something precise about two system descriptions that purport to express something about a system at two levels of detail. Only a little research has been undertaken in this area to date. The recent work [8] is a welcome start in this regard – but much more is needed. The wealth of ideas emerging from applications’ areas such as intelligent environments will surely provide material to inspire further technical development.

We can give a flavour of this too without having to delve at all deeply into the mathematical underpinnings. Let us begin with Figure 20. We see two users outside a space but proximal (they are in the same region).

Imagine that this system description is subject to the reaction rule of Figure 5. The trace tree is small and very easy to construct (we will see it below in Figure 22). However, let us
next consider Figure 21 which, again, considers entry into the space, but this time through the use of the key. At the moment the key is linked to the space but is linked to neither user.

We imagine that this system description is subject to the reaction rule given in Figure 8. Since the key is single use, the key disappears after the first application of the reaction rule. The trace tree here is even smaller.

Returning to the discussion after Figure 7. Try not to think of this as two implementations but two specifications at different levels of description. Try not to think of the first version as more permissive than the second, but simply as a version in which we are completely silent about how a user enters the room. From the perspective of the traces, try to think of them as a superposition of possibilities rather than as a system in which both users are within the space. This is what the non-determinism is capturing here. As we mentioned above, that is not to say we couldn’t implement the first version. We could, of course, but then the non-determinism cashes out in a different way – and it is indeed a more permissive implementation than that sanctioned by the second version.

There are six traces in the model for the first version, and there are four traces in the model for the second version. Note that, if we remove all reference to the key from the system states occurring in the second version (just think about the labels in the trace trees of Figure 22 – this has essentially done the job of suppressing the key) then every trace of the second version is a trace of the first. To a first approximation (that is, very roughly), and with all technicalities set aside, this is the what appears as safe vertical refinement in [8]. It gives us an answer to the question we raised above: when are two descriptions perspectives of the same system but at different levels of abstraction.

The problem with safe refinement, noted in [8], is that the empty diagram – exhibiting no behaviour at all – is a safe refinement of our original system (indeed of any system description): recall that we take the empty trace to be part of the model of every system. Nothing will go wrong with a safely refined system – but we cannot guarantee that anything specific will go right. What we need is a liveness criterion – something that ensures that some specified properties must hold. In our case it should be the requirement (captured in some sense) that a user can get into the room. This is explored in [8] to some extent – but much work needs to be undertaken in this area.

V. Final Example

Imagine that the spaces are iClassrooms and that the members of the iClassrooms are variously teachers and pupils. Perhaps a teacher is able to disconnect a pupil from their device, should they misbehave. Figure 23 is a specification of the situation.

As we mentioned much earlier, we have changed the arity of the member nodes – we now have a port that links to the intentions of the member. In this case it is the intention to disconnect a pupil in a remote iClassroom from their device. The second link from the intention node (the cloud) indicates the locus of the teacher’s wrath. This highlights the variety of roles that nodes can play in this approach to modelling. Here the node represents a command that the teacher intends to implement. How that command is to be effected is not yet clear. We can formalise that lack of clarity by introducing the reaction rule (not shown) that leads immediately to Figure 24.
By magic, the disconnection is effected and appears to be the direct consequence of the teacher’s intention. But it is not really magic at all. Presumably the disconnection is forced through the network between the teacher’s and pupil’s devices. It’s simply that, at this high level of description, we cannot see the details. We could provide (some of) those details. For example a reaction rule that turns the intention into a particular instruction on the network (Figure 25).

This message on the network creates the disconnect instruction at the relevant point (Figure 26).

After which the situation shown in Figure 24 pertains. The idea is, of course that the more complex version is a refinement of the simpler. In this case no reduction in non-determinism occurs, simply that details of how the system operates at a lower level can be hidden in a more abstract description. Such hiding of details seems to be of paramount importance, if we are to create comprehensible models, designs and analyses for extremely large intelligent environments.

VI. CONCLUSION AND FUTURE WORK

Let us return to the four points we highlighted in the introduction. First, the diagrams we have provided are certainly intuitive and descriptive. But they are more than that: they are also formal (mathematical) objects. There is essentially no difference between the informal and formal account. Second, though there are a number of new concepts related to these simple diagrams (terms such as regions, sites, inner (and outer) names, and notions such as composition, reaction, and refinement, etc.), these are fairly rapidly understood and their purpose and effects are readily comprehended. Third, we have illustrated by means of examples, possibilities for specifying behaviours at more than one level of abstraction – and, through the notion of refinement, provided a precise way to make good the claim that two perspectives are, indeed, two ways of looking at the same system. Finally, we hope that the reader could already take an elementary scenario and use the framework to describe it. If that is the case, then our fourth point is well made: since we have provided almost no indication of the mathematics that takes place “under the hood”. In this paper we have provided only the interpretation of diagrams as bigraphs, by providing some examples of the underlying place and link graphs. That these can be further understood in algebraic terms, terms that we have not described, has not been an impediment to understanding.

Within our own work on intelligent environments there is much to do. The approach we have taken here will be deployed to document, examine, effectively communicate and analyse both existing and newly planned systems – especially in the area of middleware. Beyond that, we hope to build a community of interest within the Intelligent Environments community, and to inspire further investigations across a broad range of applications’ areas.

ACKNOWLEDGEMENTS

This work has been undertaken as part of the ScaleUp project, which is funded by King Abdulaziz University, KSA.

REFERENCES


