Workshop Proceedings of the 8th International Conference on Intelligent Environments
J.A. Botía et al. (Eds.)
IOS Press, 2012
© 2012 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-61499-080-2-434

FollowMe: A Bigraphical Approach

Martin HENSON [a,1], James DOOLEY [a], Luke WHITTINGTON [a] and
Abdullah AL MALAISE AL GHAMDI [b]

[a] *School of Computer Science and Electronic Engineering, University of Essex*
[b] *Faculty of Computing and Information Technology, King Abdulaziz University*

Abstract. In this paper we illustrate the use of modelling techniques using bigraphs to specify and refine elementary aspects of the FollowMe framework. This framework provides the seamless migration of bi-directional user interfaces for users as they navigate between zones within an intelligent environment.

Keywords. bigraph, bigraphical reactive system, refinement, intelligent environment, FollowMe.

Introduction

This paper explores formal methods in the context of intelligent environments. There are many roles such methods might play - for example in providing some evidence of reliability through a verification process. For example, UPPAAL and TCTL [1] have been used in the analysis of AmI applications [2] and SPIN [3] in an AAL application [4]. Here, however, we look at another important possibility: the use of such methods to express and communicate precise designs, and to relate designs at different levels of abstraction, providing perspectives on complex system that abstract away detail.

The framework we use here to examine these ideas is *bigraphs*. The definitive account appears in [5], though the ideas have been around much longer. We begin an examination, to be elaborated in future work, using this framework of *FollowMe*: a proposal for a persistent GUI, that allows for user interfaces to follow a user through the physical world, jumping between local display devices [6]. Bigraphs can be elegantly diagrammed, and we provide a few examples. For reasons of space as much as anything, it is impossible to provide such diagrams throughout this presentation and an algebraic notation is used (see Section 1.8 below). The reader may like to refer to [7,8] for a very gentle introduction and further motivation.

The organisation of the paper is as follows. The the next section we provide a brief overview of bigraph framework, at least at a level of presentation that is adequate for the present work. In the next section we outline the issue that the FollowMe proposal addresses. After that we look at some ways to express some very elementary aspects of this in bigraphical form. Our purpose here is to focus more on the principles and possibilities inherent in deploying the formal method in the context of an application in pervasive computing - so the analysis is highly simplified: we intend to provide a

[1]Corresponding Author: University of Essex, Wivenhoe Park, Colchester, Essex, UK.
E-mail: hensm@essex.ac.uk

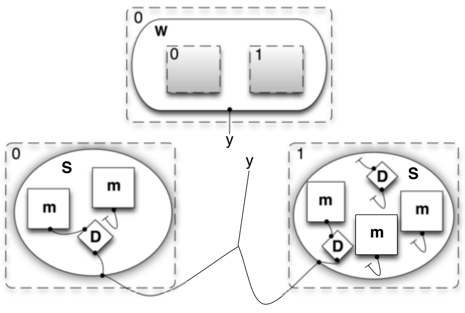

Figure 1. Two Bigraphs

comprehensive analysis in future work. In addition to abstract specifications we look at a the refinement of a specification using a technique developed in [9]. We end some conclusions and related work - especially as that is likely to be crucial to lines for our future work.

1. Bigraphical Reactive Systems

1.1. Bigraphs

A *bigraphical reactive system* (BRS) is a formal model of an interactive system. Underlying this is the notion of a bigraph - comprising two graphs: a *place graph* and a *link graph*. The former captures the spatial organisation of the system, while the latter captures connectivity within the system. Then there is the notion of *reaction rules* that can, in general non-deterministically, rewrite parts of a bigraph and express the potential dynamics of the system. Figure 1 shows how two example bigraphs (one above the other) can be diagrammed. The outer dotted rectangles are called *regions*. The shapes labelled W, S, m and D within these regions are called *nodes*. The labels themselves are called *controls*. The black dots on various nodes are *ports* and the lines connecting ports are *edges*. The inner dotted rectangles (shaded) are *sites*. Finally, the label y on the unconnected ends of two edges are *names*: the name on the upward-facing edge is an *outer name*, while that on the downward-facing edge is an *inner name*.

These diagrammatic representations of bigraphs are very suggestive, and combine the place and link graphs very helpfully. It is important, though, to understand how those two graphs may be untangled from the diagrams. What follows is a relatively informal account; consult [5] for precise details.

1.2. Signatures

The nodes in a bigraph correspond to various places, operations, entities, *etc.* and are classified as such by assigning *controls*. In Figure 1, the controls are W, S, m, and D. Controls are assigned an *arity*, that indicate the number of ports. In Figure 1 the arity of the W, S and m nodes is 1, and for D nodes is 2. A *signature*, κ, gathers all this information together. For our example bigraphs, κ is $\{W : 1, S : 1, m : 1, D : 2\}$.

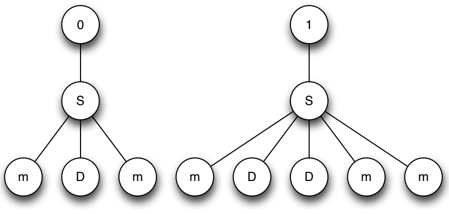

Figure 2. A Place Graph

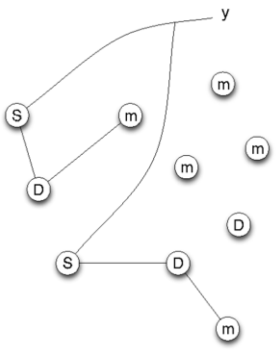

Figure 3. A Link Graph

1.3. Place and Link Graphs

The place graph represents the *nesting* of the regions, nodes and sites in a diagram. For example, the place graph of the lower bigraph of Figure 1 is given in Figure 2. In general, as is the case here, the place graph will be a *forest* of trees - when the bigraph comprises more than one distinct region, each region will be the root of a separate tree.

The link graph represents the connectivity of the nodes in a diagram. For example, the link graph of the lower diagram of Figure 1 is given in Figure 3 As is the case here, a connection may link more than two nodes. So the place graph is, in general, a *hypergraph*.

1.4. Interfaces

A bigraph has two *faces*: an *outer* and an *inner* interface. Each is a pair comprising a face for the place graph and a face for the link graph. A face for a place graph is a natural number. In this framework we treat a natural number as the set of numbers that are strictly smaller. That is, $n = \{0,1,\cdots,n-1\}$. The inner face of the place graph for the upper bigraph is 2. That is, $\{0,1\}$ - the bigraph containing two sites. The inner face of the place graph for the lower bigraph is 0. That is, $\{\}$ - the bigraph contains no sites. The outer face of the upper bigraph is 1 and of the lower bigraph is 2, because they comprise one region and two regions, respectively. A face for a link graph is a set of labels - those attached to the unconnected edges. In this framework we write, for example, the set of labels $\{x,y,z\}$ as $\{xyz\}$. The inner face of the link graph for the upper bigraph is $\{y\}$ -

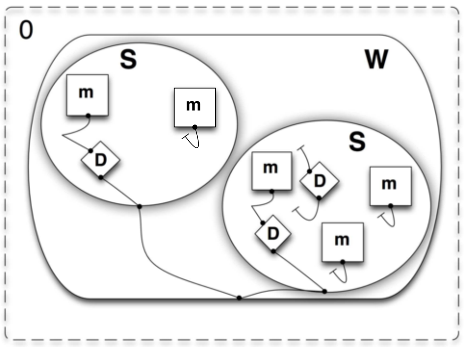

Figure 4. Result of Composition

one inner name. Its outer face is $\{\}$ - no outer names. The inner face of the link graph for the lower bigraph is $\{\}$ - no inner name, while the outer face is $\{y\}$ - one outer name.

The *place graph interface* for the upper bigraph is written $2 \to 1$ - from inner face to outer face. Thus, the place graph interface for the lower bigraph is $0 \to 2$. The *link graph interface* for the upper bigraph is written $\{y\} \to \{\}$ - again from inner face to outer face. Thus, the link graph interface for the lower diagram is $\{\} \to \{y\}$. Given all this, the interface for the upper diagram is $\langle 2, \{y\}\rangle \to \langle 1, \{\}\rangle$ and the interface for the lower diagram is $\langle 0, \{\}\rangle \to \langle 2, \{y\}\rangle$.

1.5. Composition

Note that the outer face of the lower diagram matches the inner face of the upper diagram. This means that the two diagrams may be *composed*. Think of sites receiving regions and inner names receiving outer names. When we compose these two diagrams in this way we obtain Figure 4. The interface of this resulting bigraph is $\langle 0, \{\}\rangle \to \langle 1, \{\}\rangle$. Generally, we will write $\langle k, X\rangle$ (*etc.*) for an interface.

1.6. Reaction Rules

Bigraphs evolve according to *reaction rules*. Figure 5 shows such a rule. The rule involves nodes from Figure 1. We showed various ports in Figure 1 as explicitly disconnected. In Figure 5 we have simply not shown the disconnected ports at all. Here an external member m of a space S (which may contain other entities - the site) enters the space. The member and the space are contained in the same region, and are therefore proximal before the reaction takes place. Generally there will be a number of reaction rules comprising a bigraphical reactive system. In any given system state (bigraph), several rules may apply, and indeed a given rule may apply in many ways. Thus, evolution of the system under reaction will generally be non-deterministic. A reaction rule, generally, has the form (R, R', η) where R, the *redex* is transformed into R', the *reactum*. η can often be omitted - but is sometimes needed in order to determine how sites in the reactum are related to sites in the redex. We often write reaction rules as $R \to R'$.

Let κ be a signature and B be a set of reaction rules over κ. Then (κ, B) is a *Bigraphical Reactive System* (BRS).

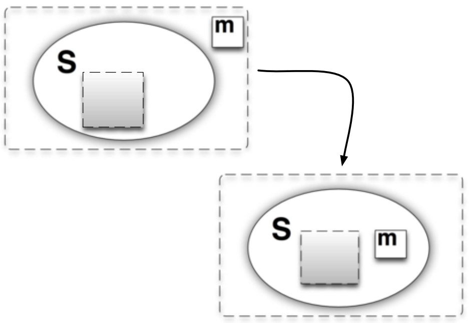

Figure 5. A Reaction Rule

1.7. Bigraphs Formally

We will look at refinement in Section 4, so we need a mathematically precise description of what we have introduced so far.

Definition 1.1 *(See [5] for further details) A bigraph is a quintuple*

$$(V, E, cntrl, prnt, link) : \langle k, X\rangle \to \langle m, Y\rangle$$

where V is a finite set of nodes, E is a finite set of edges, cntrl is a map from V to the signature κ that assigns controls to nodes, prnt is a (partial) map that takes each site and node (where it has one) to its parent node or region, and link is a map that takes each inner name and port to the edge or outer name to which it is connected.

1.8. Bigraphical Terms

Bigraphical diagrams are extremely helpful and suggestive. However, it is also helpful to have a more concise means to express them. A comprehensive axiomatisation is provided in [10], the language of which is often used, variously extended, for such presentation. For bigraphs A and B, we write $A \parallel B$ for juxtaposition of their regions (distal combination), and $A \mid B$ for juxtaposition within a single region (proximal combination). $U(V)$ represents the containment of V within U. $/x.U$ connects all occurrences of the outer name x in U by an edge (we will generalise this to allow a set of outer names *e.g.* $/X.U$ - writing the set X without brackets and commas). We write $-_i$ for the site labelled i and x/Y for the connection from the inner names in the set Y (which may be empty) to the outer name x (again writing the set Y without brackets and commas). Finally, for a node K with n ports named $\vec{x}$, we write $K_{\vec{x}}(U)$ for *ion* of K (containing U) - that is, K connected to outer names $\vec{x}$.

Given all this, we can write the two bigraphs of Figure 1 as

$$/z.W_z(-_0 \mid -_1) \mid z/y$$

and:

$$/x.S_y(m_x \mid D_{xy} \mid /w.m_w) \parallel /x.S_y(m_x \mid D_{xy} \mid /w.m_w \mid /w.m_w \mid /wz.D_{wz})$$

The reaction rule in Figure 5 can be written:

$$/z.S_z(-_0) \mid /y.m_y \;\rightarrow\; /z.S_z(-_0 \mid /y.m_y)$$

2. An Overview of FollowMe

Consider an environment that can be decomposed into zones, where each zone has networked-enabled display devices (and interaction methods such as a touch screen, mouse, keyboard, *etc.*), consider a user, or users, who migrate from zone to zone, and assume that there is a means to detect that migration. The task is to migrate the users' current set of application interfaces from the devices they leave to those at which they arrive. Devices themselves may be mobile and move from zone to zone. This is an example of context-aware computing, of course, and raises additional issues of security (does the user have the right to use the devices), trust (can the user rely on the devices), discovery (what are the devices usable within a zone?), and multiple users (issues of sharing, priority, availability, *etc.*)

This topic, of a persistent GUI in a context-aware scenario, has a history, including *teleporting* using the X-window system to direct client application input/output between different displays [11]. This makes use of an active badge to resolve user location and to prompt display migration, but is not able to dynamically switch the display device used by the application. Another approach is the *Everywhere Displays* project of IBM [12,13] which removes the need for multiple network-enabled display devices and is based on projecting interfaces onto surfaces and using cameras to track the user. There are however, limitations regarding the scale of deployment of such an approach beyond a single zone.

The *FollowMe* approach [6,14] is based on HTML5 technology that permits page-to-server and server-to-page asynchronous event notifications - and by achieving this within a browser, is essentially a device-independent approach.

In this paper we do not by any means reach the level of detail discussed in [6] in terms of the underlying technology, remaining at a much higher level of description and simplifying enormously in order to illustrate some fundamental ideas of modelling in this framework.

3. Basic Modeling

We imagine an *intelligent environment S* with a single port representing a network local to that space. Within the environment there are *zones*. Zones have a single port representing their local network connection. An example might be the *iSpace* at the University of Essex, which is an apartment comprising a living room, kitchen, hall, study, bedroom, and bathroom. Within the environment there are *devices D*, which may be mobile - moving from zone to zone. A device has two ports - the first representing a local (to a zone) network connection and the the second an attachment to a user of the device. Users *U* may also move from zone to zone. Each has a single port representing a potential attachment to a device. This crude picture captures very little - but it is the basic backdrop against which more specific details of the FollowMe approach can be modelled. We have effectively outlined a signature and can describe a simple BRS that captures these high-level considerations. In fact the signature is probably insufficient - as we may wish to avoid zones containing environments, though perhaps not zones containing zones. Such constraints can be enforced, but we will not dwell on the technical details in this paper.

3.1. Mobile Devices

In this section we will assume only one intelligent environment (one S node) at the outermost level of any system state and we will assume no nested zones.

We will look at the issue of a device migrating from one zone to another. Here is our first reaction rule (Rule R1).

$$Z_{x_0}(D_{x_1x_2} \mid -_0) \mid Z_{x_3}(-_1) \rightarrow Z_{x_0}(-_0) \mid Z_{x_0}(D_{x_1x_2} \mid -_1)$$

This is already quite complex. The outer names (the x_i) may connect variously to entities comprising the context into which the redex of the rule matches. For example, consider the following situation.

$$/x_0x_1x_2x_3.S_{x_0}(Z_{x_0}(D_{x_1x_2} \mid -_0) \mid Z_{x_3}(-_1))$$

The local network connection of the leftmost zone into the local network of the space. The device is neither connected to a user nor to the network. The rightmost zone is not connected to the network. However, consider this slightly different situation.

$$/u_0u_1.S_{u_0}(Z_{u_0}(D_{u_0u_1} \mid -_0) \mid Z_{u_0}(-_1))$$

In this case both the zones and the device connect to the network connection of the space. The device does not connect to any user.

To see the reaction rule in action, consider the following bigraph. Once again, note that we create the edges at the outermost level.

$$/v_0v_1v_2.S_{v_0}(Z_{v_0}(U_{v_1} \mid D_{v_0v_1}) \mid Z_{v_2}())$$

When the reaction takes place, the device moves from one zone to the other, stretching (so to speak) the links to the user and the leftmost zone's network (note that the rightmost zone is unconnected).

$$/v_0v_1v_2.S_{v_0}(Z_{v_0}(U_{v_1}) \mid Z_{v_2}(D_{v_0v_1}))$$

Although we shall not provide any, we might develop rules that specify user interaction across the network through a connected device. If we insist that all such interaction requires that the user and the device are in the same zone, and that the device is connected to the local (zone) network. Then the stretched links we have generated in our example are merely *notional* connections (that is, they are not usable). The only reactions we might provide would be to break such notational links. For example, this reaction rule removes the unusable connection between a device in a zone other than its (former) user.

$$/z.Z_{x_0}(-_0 \mid U_z) \mid Z_{x_1}(-_1 \mid /x.D_{xz}) \rightarrow Z_{x_0}(-_0 \mid /z.U_z) \mid Z_{x_1}(-_1 \mid /xz.D_{xz})$$

Note that the closure of x at the device, prevents the device from connecting to any entity in an instance of the site $-_1$ and to anything within an enclosing context. We are insisting that the device, having moved unceremoniously away from its user, may not engage in any network interactions. Thus this reaction cannot take place until the unusable network link has been removed. That would be captured by this reaction rule.

$$/z.Z_z(-_0) \mid Z_u(-_1 \mid D_{zv}) \rightarrow Z_z(-_0) \mid Z_u(-_1 \mid /x.D_{xv})$$

This analysis illustrates well the subtleties of both the example and the framework we are using. The reader may find it useful to draw the “node and link" diagrams that correspond to the bigraphical terms we have presented here.

3.2. Cloud and Spaces

We now want to model a situation in which the users' data is stored in a cloud that is potentially remote from the environment. This data is served to the device to which a user is connected. We will add, then, an entity C of arity 1, to represent the cloud server. The port is the wide-area network connection to the environment (and probably much else besides). An environment can connect and disconnect to its cloud server. This is captured by the following reaction rules.

$$\begin{array}{lcl} /x.C_x(-_0) \mid\mid /x.S_x(-_1) & \rightarrow & /x.(C_x(-_0) \mid\mid S_x(-_1)) \\ /x.(C_x(-_0) \mid\mid S_x(-_1)) & \rightarrow & /x.C_x(-_0) \mid\mid /x.S_x(-_1) \end{array}$$

Note that the combination is distal. The two sites allow both the cloud and the environment to contain arbitrary entities. We have specified a disconnection that is not graceful - there is no guarantee that a user's session will terminate smoothly.

Using the distal combination does not force the cloud server to be remote from the environment. If B is a single building then $B(-_0 \mid -_1)$ can host both the server and the environment locally. On the other hand if *Mexico* and *Portugal* are the countries one imagines, then $Mexico(-_0) \mid\mid Portugal(-_1)$ can separately host the cloud and the environment.

We shall keep the model simple, not identifying *specific* cloud-based data for a particular user. We want to bring over the cloud-based data when a user is connected to a device which is connected to the cloud through its enclosing zone and environment.

$$\begin{array}{l} /x.(C_x(-_0) \mid\mid S_x(Z_x(/y.(U_y \mid D_{xy}) \mid -_1) \rightarrow \\ /x.(C_x(-_0) \mid\mid S_x(Z_x(/y.(U_y \mid D_{xy}(-_0)) \mid -_1) \end{array}$$

A user may disconnect from a device.

$$/y.(U_y \mid D_{xy}(-_0)) \rightarrow /y.U_y \mid /y.D_{xy}$$

That is, when a user disconnects from a device, her session is immediately terminated. This is, perhaps, too abrupt. To improve on this, first let us permit the user to change the local data. Let d be a datum of arity 0.

$$/y.(U_y \mid D_{xy}(-_0)) \rightarrow /y.(U_y \mid D_{xy}(d \mid -_0))$$

Of course, this is the barest indication only as to how a full account would be constructed - though the reaction here allows an arbitrary number of such data changes.

Given this, we can modify the disconnection reaction above, so that the local data is saved - transferred to the cloud on disconnection.

$$\begin{array}{l} /x.(C_x(-_0) \parallel S_x(Z_x(/y.(U_y \mid D_{xy}(-_1)) \mid -_2) \rightarrow \\ /x.(C_x(-_1) \parallel S_x(Z_x(/y.U_y \mid /y.D_{xy}) \mid -_2) \end{array}$$

If the user were later to move to another zone and connect to an available device, the previous session would then be reinstated to that new device.

4. Refinement

Consider very simple reactions that would allow a user to connect to, and disconnect from, an available device.

$$\begin{array}{lll} /y.U_y \mid /y.D_{xy} & \rightarrow & /y.(U_y \mid D_{xy}) \\ /y.(U_y|D_{xy}) & \rightarrow & /y.U_y \mid /y.D_{xy} \end{array}$$

We might wish to insist that a user has the necessary credentials to make such a connection. So we might refine the system - requiring that the user has a *code* (a nullary node K) with which to attach to an appropriate device (with the same code) . The code disappears from the device once the user connects. The code should become available again once the user relinquishes control of the device.

$$\begin{array}{lll} /y.U_y(K) \mid /y.D_{xy}(K) & \rightarrow & /y.(U_y(K) \mid D_{xy}) \\ /y.(U_y(K)|D_{xy}) & \rightarrow & /y.U_y(K) \mid /y.D_{xy}(K) \end{array}$$

In the first regime, any user can connect to any available device. In the second, a user may have no code at all - or a code different from that of a given device - so there is a potential reduction of non-determinism in moving from one model to the other.

There should be a *formal* relationship between these models. Is the implementation (that includes the security code) correct? That is, does it conform to the specification (without the security code)?

An answer might be: when any behaviour of the implementation is consistent with a behaviour of the specification. The notion of behaviour is obviously captured by the reaction rules. So imagine, for a BRS A, a set $Tr(A)$ (we make this a little more precise below) of *traces* each of which is a record of the evolution of a bigraph of A under its reaction rules. Then to a first approximation we want $Tr(C) \subseteq Tr(A)$. That is, every behaviour of the implementation BRS, C, is a behaviour of the specification BRS, A. However, this cannot be right, since the implementation comprises bigraphs different from those of the specification (in our example they contain K nodes that do not exist in the specification). So we need some abstraction mapping F from the implementation to the specification that negotiates the differences - and then we want $F(Tr(C)) \subseteq Tr(A)$.

A little work has been undertaken in the area of refinement, notably, in [9], where, among other things, a notion of *safe vertical refinement* is introduced, which builds on the narrative above. We provide a précis of the definitions and results here; the reader is encouraged to consult the source for full details.

Definition 4.1 *(from [9]) A* trace *for a BRS A is a (possibly infinite) sequence of bigraphs of A $\langle a_0, a_1, ... \rangle$ such that $a_i \to a_{i+1}$ is a reaction of A. We write Tr(A) for all the traces of a BRS A. If F is a functor from BRS A to BRS A' then we extend F to traces of A by applying it point-wise.*

First, note that $Tr(A)$ is *prefix closed*. That is, if t' is a prefix of the sequence (trace) t and $t \in Tr(A)$ then $t' \in Tr(A)$. In particular, the empty trace is in $Tr(A)$. Second, we need the notion of a *functor*. A full description is beyond the scope of this paper - however, essentially, a functor F from A to A' is a pair of maps from interfaces of A to interfaces of A' and a map from bigraphs of A to bigraphs of A' (respecting interfaces) that, in particular, preserves composition (as described in Section 1.5 above).

Definition 4.2 *(from [9])*

$$A \sqsubseteq_F C =_{df} F(Tr(C)) \subseteq Tr(A)$$

That is, abstraction (specification) A refines to concrete (implementation) C when the traces of C (abstracted to the signature of A via F) are all traces of A.

What kinds of functors lead to refinements? Clearly not every functor will do - since $F(t)$ for a trace t must be a trace of the specification BRS, and not just some sequence of bigraphs of that specification. The authors of [9] introduce a notion of a *safe abstraction functor*, which has the following pleasant property: if F is a safe abstraction functor from C to A then C is a refinement of A. One form of safe abstraction functor is a *hiding functor*, and that is what we need here.

Proposition 4.1 *(adapted from [9]) Let C be a BRS over signature Σ and let H be a set of controls of Σ. Let A be a BRS over the signature Σ restricted to controls not in H. Then the* hiding functor *F defined as follows is a safe abstraction functor. F is the identity mapping on interfaces. Then, for any bigraph β, of C: $(V, E, cntrl, prnt, link) : \langle k, X \rangle \to \langle m, Y \rangle$ we define $F(\beta)$ to be $(V', E, cntrl', prnt', link) : \langle k, X \rangle \to \langle m, Y \rangle$ where V' is that subset of V whose controls are not in H, $cntrl'$ is cntrl restricted to V', and where:*

$$prnt'(l) =_{df} \begin{cases} prnt(l) & when\ cntrl(prnt(l)) \notin H \\ prnt'(prnt(l)) & otherwise \end{cases}$$

Since a hiding functor is a safe abstraction functor, it leads immediately to a refinement. In our example we simply take H to be $\{K\}$ - the security code.

5. Conclusions, Related and Future Work

The objective of this paper is to introduce the basis notions of bigraphical reactive systems as a basis for modelling systems within an intelligent environment framework, and we have done little more than hint in the general direction of how a sophisticated modelling of the FollowMe framework could be taken.

Previous work in modelling context-aware systems has established that reconfigurations of systems (such as we have illustrated here) are not difficult to model. However, querying the state of a system - establishing in particular that some state of affairs is *not* the case - is hard. Moreover, the straightforward approach we have taken here takes it as given that the user has perfect knowledge of the context - for example is fully aware of what devices are available in the zone they occupy. A more sophisticated approach would allow the user to have a representation of the context. That representation may be incomplete - for example, some devices present in the actual context may not be present in the representation. Such an approach has been termed *Plato-graphic* [15,16]. Such models comprise three components: a *context*, a *proxy*, and an *agent*. The context and agent are *disjoint*, meaning that their signatures have no common entities: the agent interacts with the world (context) only through a representation (proxy).

Safe refinement is not sufficient, since the empty trace (the system that does nothing) is always a safe refinement of any specification. [9] consider *liveness* too - something that needs considerably more attention.

Consideration of such issues is beyond the scope of this paper, but one of the authors (Whittington) is working on a systematic analysis of the FollowMe framework which will involve modelling the user's representation of the actual environment plato-graphically, the refinement of descriptions at various level of detail, and using refinement in order to provide system perspectives - for example from the user's point of view as they navigate the environment. This analysis will also capture the additional issues of security, trust, discovery, and competition between multiple users.

Acknowledgements

The authors wish to thank Hani Hagras, Gaëtan Pruvost, and Christian Wagner, who collaborated on the development of the FollowMe design and implementation. This work has been undertaken as part of the *ScaleUp* project, which is funded by King Abdulaziz University, KSA, and as part of the EU FP7 funded ATRACO project.

References

[1] B. Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and Ph. Schnoebelen, *Systems and Software Verification, Model-Checking Techniques and Tools*, Springer, 2001.

[2] J. C. Augusto and P. J. McCullagh, Ambient intelligence: Concepts and applications,*Comput. Sci. Inf. Syst.* **4**, 1 (2007), 1–27.

[3] G. J. Holzmann, *The SPIN Model Checker - primer and reference manual*, Addison-Wesley, 2004.

[4] J. C. Augusto, H. Zheng, M. D. Mulvenna, H. Wang, W. Carswell, and W. P. Jeffers, Design and Modelling of the Nocturnal AAL Care System, in *Ambient Intelligence - Software and Applications - 2nd International Symposium on Ambient Intelligence* 2011, 109–116.

[5] R. Milner, *The Space and Motion of Communicating Agents*, Cambridge University Press, 2009.

[6] J. Dooley, C. Wagner, H. Hagras, and G. Pruvost, FollowMe: The Persistent GUI, in [14], IEEE Computer Society 2011, 123–126.

[7] M. Henson, J. Dooley, L. Whittington, and A. A. G. A. Malaise, Towards Simple and Effective Formal Methods for Intelligent Environments, in [8] *(forthcoming)*, 2012.

[8] J. C. Augusto and V. Callaghan, Eds., *Eighth International Conference on Intelligent Environments (IEâĂŹ12)*, IEEE Computer Society, 2012.

[9] G. Perrone, S. Debois, and T. Hildebrandt, Bigraphical refinement, *Electronic Proceedings in Theoretical Computer Science* **55** (2011), 20–36.

[10] R. Milner, Axioms for bigraphical structure, *Mathematical Structures in Computer Science* **15**, 6 (2005), 1005–1032.

[11] T. Richardson, F. Bennett, G. Mapp, and A. Hopper, A Ubiquitous, Personalized Computing Environment for All: Teleporting in an X Window System Environment, *IEEE Personal Communications* **1**, 3 (1994), 6–12.

[12] G. Pingali, C. Pinhanez, T. Levas, R. Kjeldsen, and M. Podlaseck, User-following displays, in *IEEE International Conference on Multimedia and Expo (ICME)*, 2002.

[13] G. Pingali, C. Pinhanez, A. Levas, R. Kjeldsen, M. Podlaseck, H. Chen, and N. Sukaviriya, Steerable interfaces for pervasive computing spaces, in *Pervasive Computing and Communications (PerCom)* 2003, 315–322.

[14] *Sixth International Symposium on Parallel Computing in Electrical Engineering (PARELEC)*, IEEE Computer Society, 2011.

[15] L. Birkedal, S. Debois, E. Elsborg, T. Hildebrandt, and H. Niss, Bigraphical models of context-aware systems, in [16], 2006.

[16] L. Aceto and A. Ingólfsdóttir, Eds., *Foundations of Software Science and Computation Structures, 9th International Conference (FOSSACS)* LNCS 3921, Springer, 2006.