Resources for Situated Actions

Gavin Doherty[1], Jose Campos[2], and Michael Harrison[3]

[1] Trinity College Dublin, Dublin 2, Ireland
[2]University of Minho, Portugal
[3]Newcastle University, UK

Gavin.Doherty@cs.tcd.ie, Michael.Harrison@ncl.ac.uk, jose.campos@di.umhinho.pt

Abstract. In recent years, advances in software tools have made it easier to analyze interactive system specifications, and the range of their possible behaviors. However, the effort involved in producing the specifications of the system is still substantial, and a difficulty exists regarding the specification of plausible behaviors on the part of the user. Recent trends in technology towards more mobile and distributed systems further exacerbates the issue, as contextual factors come in to play, and less structured, more opportunistic behavior on the part of the user makes purely task-based analysis difficult. In this paper we consider a resourced action approach to specification and analysis. In pursuing this approach we have two aims - firstly, to facilitate a resource-based analysis of user activity, allowing resources to be distributed across a number of artifacts, and secondly to consider within the analysis a wider range of plausible and opportunistic user behaviors without a heavy specification overhead, or requiring commitment to detailed user models.

1 Introduction

It is typical in human computer interaction when specifying the system to describe the tasks that are the proposed basis for the work to be supported. A process of task analysis elicits the tasks that people carry out with the existing system used as a basis for designing the tasks for which the new design is intended. The problem with this approach is that the way the user actually uses the proposed system in practice may differ from what the designer expects.

In order to reason about the usability of the system we must introduce some notion of plausible user behavior. However, if we introduce overly restrictive or unrealistic assumptions about user behavior, the value and validity of our analysis can be questioned. For example, consider an analysis of whether the user is likely to put the proposed system into an unsafe or undesirable state. We need to introduce assumptions about the behavior of the user because exhaustively checking the system model alone will throw up an unlimited number of spurious problems. Exhaustive analysis corresponds to the assumption that the user will interact with the system (e.g. push buttons) at random. Hence, in looking at the effect of a sequence of user actions on the system, we do not want to consider traces which the user is unlikely to carry out (irrespective of whether they are “good” or “bad” actions).

Published as: G. Doherty, J. Campos and M.D. Harrison, Resources for Situated Actions, Interactive Systems: Design, Specification and Verification, 2008, Springer LNCS vol. 5136, pp.194-207, 2008.

If we combine the system model with a task model, we assume that the user will follow the pattern of interaction defined by the structure of the task. While this may still correspond to a large number of possible behaviors, the resulting set can still be criticized as being too prescriptive. This approach can still ignore many highly plausible behaviors and will be unsuitable for many goal directed situations for which the tasks are not well defined. Furthermore, in the real-world, users often behave opportunistically according to the situation they are in, and the resources and actions available to them in that situation.

An alternative approach is to start the other way round. Here the resources that are expected to help the user: to achieve goals; to make choices between actions; to carry out specific activities [10] are considered explicitly. Resources are codified in terms of: status or state; action possibility; action effect information; the plans that are appropriate to achieve goals and goal information. These resources act as constraints on the user and under certain assumptions will create the circumstances in which the goals are achieved. The model makes explicit how these resources are organized and defined in the interface. This can be used in analysis to explore the possible paths that are permitted by the resource organization. In [3] we looked at the resourcing of actions within a task structure; in this paper we develop the analysis a step further, and examine the feasibility of a purely action-based analysis in which we do not commit to a particular task structure, similar to that described in [10] but in this case applied to a formal model. We explore an approach to modeling and analysis based on resource constraints in two ways. We first consider the dyadic relationship between the user and the device. The user has goals and the device supports them in achieving these goals. We explore this relationship and the constraints that are imposed by resources. The device is in practice embedded within a context. This context may additionally constrain the user. Hence the second part of the paper explores the user embedded within a smart environment. We explore a control system where the operator is only able to control aspects of the system when they are within a certain proximity of the system or if they have saved the control for future use. We explore different assumptions about the resources provided to users within this environment, and the potential effects on user strategies and behaviour.

We propose that by looking at the resourcing of individual actions, we can selectively introduce constraints on user behaviors which need not be as restrictive as a task model. We propose that this is also a natural and useful vehicle for analysis of a design, and particularly suited to recent trends towards more mobile, distributed and heterogeneous systems. An added advantage is that we can take advantage of tool support for exploring the consequences of these assumptions.

2 The resourced action approach

Individual user actions are taken as the basic units of analysis. The resourcing of each of these actions is specified independently. The focus of analysis then becomes whether each individual user action is appropriately resourced, or whether appropriate combinations of resourced actions will lead to the achievement of user goals. The starting point is that for an action to be afforded in a particular context, certain

information resources must be present in that context. For example, if a mobile phone (the device) has an action to save a draft text message, we could specify that (1) action availability is resourced (the "save" option is currently on the screen), (2) the action is enabled (the message memory is not full), (3) action-effect information is available (is the label "save to drafts" or just "save"?), and (4) required information about the current state is available (have I saved it already?). Regardless of how I ended up editing a text message (did I reply to another message, is it a group text?), or higher level user tasks and goals (which may be varied), the basic resourcing for this action remains much the same.

The specification of the system is thus structured as a set of actions, which affect the state of the system, accompanied by an appropriate model of system state. Various forms of interactive system specification (including interactor models) could provide a means to build this specification, and indeed Modal Action Logic [4] focuses on the actions supported in an interface, but the additional structuring provided by interactor models is not a necessary part of the approach. A difference from other approaches to interactive system analysis is the addition of resourcing requirements to accompany each action. We can consider more sequentially constrained interactions if needed, whether this is through the structure of the system, or due to likely plan-based behavior by the end user. Even if we take the view that actions are situated [9], we can still allow for the possibility by considering plans themselves as resources [10]. It is important to note that this approach is not just a vehicle for automated analysis of behavior, but also leads us to consider, in a methodical fashion, the resourcing of situated user actions. The possibility of tool support however, allows us to more easily and comprehensively identify situations where actions may be inadequately resourced.

The rest of this section considers the steps involved in the analysis. The approach is comparable with a number of other evaluation techniques. For example cognitive walkthrough [8] takes a task or scenario and requires the analyst to ask questions systematically of the interface. The questions have similarities with those that are used in this paper. The main difference between this work and cognitive walkthrough techniques in general is that (i) the information that resources the interaction is considered in more detail in terms of the type of information that it is and (ii) the aim of the activity is to move towards a formal analysis and representation of these resources. Observational techniques on the other hand such as distributed cognition [5] explore the environment in which the work is carried out to characterize how action is resourced. Elements of distributed cognition are also captured in the approach described in the paper. Our basic premise is to specify and examine the resourcing of individual actions. This approach can form a useful vehicle for goal based analysis, as one can ask questions such as whether resourced actions are available which will support achievement of the user's goal. The basic process proposed is as follows:

- specify the actions
- specify the resourcing of actions, and perform initial analysis, possibly redesigning and refining specification
- consider and specify potential user goals
- formulate properties, including those surrounding user goals

- run the properties over the model, and analyze the results, possibly redesigning and refining the specification

Of importance for mobile applications is the fact that actions may only be resourced in particular locations; in this case a location model (however simple) must be included within the analysis. Likewise, certain actions, including those to access particular resources, may only be available in certain locations. We will explore the issue of context and location modeling further in Section 3. We have a choice to make in terms of analysis regarding how much of the user's mental state we wish to include in the analysis; if a current state of knowledge of the user is important to the analysis, then this state of knowledge must be propagated through the steps of the interaction. For the purposes of this paper, we do not pursue this form of user modeling, although it is an attractive proposition for certain types of analysis, for example mode error.

2.1 Specifying Resources

The specification progresses by defining actions. Having specified the actions, we move on to consider the resources which are required for the user to carry out these actions. To do this effectively we must know whether information which is potentially available through the system is visible when the action is to be carried out. Thus some visibility model must be included; this can include what is seen in the environment as well as the device. We have a choice of specifying the exact information to be displayed, or simply indicating the availability of the resource. Existing mechanisms for denoting visible state, such as those in interactor models can be used. Within an automata-based specification language such as Uppaal we can associate resources with states, although use could also be made of integer variables and synchronizations. The system specification defines two things: the resources which are available in a given state, and the actions which can be performed, which affect the set of available resources. Following [10], we consider here what form these may take in terms of typical interfaces.

- *status/visible information* - a resource may simply consist of a piece of information, for example the display indicates that a message is waiting (a resource) in order for the user to perform an action to read the message. This is distinct from the system being in a state where reading a message is possible. The same mechanism can also indicate system status if this is being used in the user's interaction strategy.
- *action possibility* - a resource may consist of information that an action is available. There are two issues here, one is the information that the possibility for carrying out the action exists (e.g. the resource lets the user know they can save an unsent message for resending later, a feature they were unaware of), the second is that the action is enabled (or not) in the current state - perhaps the message memory is full.
- *action effect information* - a resource may let the user know what the likely effect of an action will be. The same piece of information on action availability may also convey information on action effect; "press ok to save" conveys information both on action possibility and on action effect.

- *plan information* - some resources provide plan information, that is, they aid in the sequencing of user actions. For example, interfaces in which an overall task performance sequence is made explicit ("You are in step 3 of 5") are providing a plan resource. We could deal with plan resources in much the same way as for tasks, and either trigger a hardcoded sequence or simply constrain certain aspects of the behavior or sequence - effectively providing a partial model.
- *goal information* - some resources may correspond to user goals, helping the user to formulate and keep track of multiple goals. For example, "there are new messages" could act as a goal resource within the interaction. In complex, real-world situations, there may well be a hierarchy of different goals, and goals may possibly conflict, so denoting resources as goal resources is only a small part of the analysis of goals.
- *internal resources* - some resources may be internal to the user - knowledge in the user's head instead of the world. In terms of modeling, we would be introducing resources and updating them with actions (such as reading the system display).

A question in terms of specification is whether any element of this categorization is contained within the model? Given that a resource may play a number of different roles, this could be problematic, however there is also the issue that a particular presentation of the information may support some uses better than others. While specifying the resourcing for particular actions, it is natural to identify obvious resourcing issues. As the analyst must consider each action and appropriate resources, it may be clear that a particular resource would not be available in the proposed design, and an immediate consideration would be given to the problem. However, many resourcing problems may be more subtle in their evolution, and will not be clear from inspection, particularly if the user has multiple goals, and interleaves actions which contribute to different goals. Other issues could relate to the impact of interruptions on the resourcing of particular actions.

2.2 Using goals in analysis

Without assuming a set of predefined tasks we assume the interaction is purposeful in the sense that the user has a goal. The user carries out a set of actions to achieve several goals through simple action or a complicated orchestration of activities. Well designed systems provide relevant information that can be acted upon by the user. This information might remind the user of their goal or the means by which they are to achieve the goal or the possibilities for action or how to invoke the action itself. Our analysis will be carried out with respect to user goals to include:

1. Goal is to obtain information - is it possible to reach a state or resource configuration in which the information resource is available?
2. Goal is to perform a procedure - the actions of the procedure are resourced, and the sequencing of the procedure is possible while providing appropriate resources at each point.
3. Goal is to put the system in a particular state or set of states - fully resourced sequences exist in which state is reached.

By default, this form of analysis will view usability problems in terms of insufficiently resourced actions, and suggest increased resources at key points in the interaction.

2.3 Tool support for analysis

If we specify the system state, in terms of resources, and the behavior of the system, in terms of the effects on available resources, we can examine the resource requirements of individual actions. A question which then arises is how tool support can be used to support the analysis. With respect to the three analyses in Section 2.2 above, property (1) is simple reachability - can we reach a state in which the resource is available. This however, does not tell us anything about whether it is plausible that the user would get to this state. Property (2) is the form of analysis introduced in [3] - we have a task structure to be followed, and we need to check that each step in the task is appropriately resourced. With this form of analysis the behaviors considered are plausible, but many plausible behaviors are ignored. The final property (3) tells us that we can reach the goal through an appropriately resourced sequence of actions, but does not constrain this sequence. In terms of the mechanics of the analysis process, we might well split complex goals up into a number of sub-goals, and look at these sub-goals independently and in combination. As stated previously our analyses will generally introduce assumptions about the user behavior - in this case, that the resources we specify for the actions are used by the user in selecting and carrying out those actions. There are a number of distinct modes of analysis based on these:

[Assumptions+Starting situation+Model+Task+Goal -> Boolean] When we combine these assumptions with a model of the system and a model of user behavior (e.g. a task model) and a starting situation we can ask whether the goal state is always reached when we carry out the task.
[Assumptions+Model+Task+Goal -> Starting situations] If we leave the starting situation undefined, we can ask for which starting situations we can/will reach the goal by performing the task.
[Assumptions+Model+Goal+Starting situations -> Behaviours] Alternatively, we can simply give the starting situation and system model, and analyze the range of possible behaviors which result in both positive and negative outcomes. The analysis in this case would focus on the strategies represented by this behavior and if they can be improved or added to by altering the resourcing of user actions.
[Assumptions+Model+Goal -> Starting situations] If we do not specify the starting situations, just as for the task based analysis, we can ask under which conditions we are resourced sufficiently to reach the goal.

Model checking enables exploration of the behavior of a (finite) model of the system. Modeling assumptions and tasks as restrictions on the system's behavior, we can determine whether specific (goal) states can always be reached. This corresponds to the first type of analysis identified above. Regarding the second type of analysis, if the starting situation is left undefined, model checking will attempt to provide counter examples. However these counter-examples identify situations under which the

system does not exhibit the desired behavior. The alternative, then, is to generate all possible starting situations (remember that the models must be finite for model checking to work) and reduce the analysis to a series of instances of the first type. The exhaustive generation of these initial situations can, of course, be tool supported. Regarding the third type of analysis, model checking enables, as already noted, the identifications of behaviors that do not result in the achievement of a goal. These behaviors can then be analyzed to understand how the resourcing can be changed to prevent them. In the last type of analysis, and because we are not prescribing a behavior, we can perform an analysis similar to the previous one, but paying attention to the initial states of the behaviors being generated by the tool. The iterative aspect of the process provides an additional advantage over cognitive walkthrough (beyond considering all the behaviors rather than just one).

3. Smart Environment Example

In this section we illustrate the role of resources in specification through a ubiquitous system designed to support a process control system [7]. For the analysis, we make use of a set of Uppaal models [1] which define the state of the system and the mobile device, including its location. There is no space in this paper to describe Uppaal in detail. A detailed explanation of the models is not required to appreciate the approach (see http://homepages.cs.ncl.ac.uk/michael.harrison/papers/pucketmobobsnr2.xml for one version of the model compatible with Uppaal 4.0.6). As stated previously, the analysis is to a large extent independent of the formalism, as long as we can reason about the availability of both actions and information within a particular situation. We can see our model of the setting as comprising a number of components, the plant, incorporating tanks and pumps, the mobile device, and the context.

The details of the process are irrelevant to the current consideration and can be found in [6]. Our concern is how the goal of the process is achieved by an operator, along the lines discussed in section 2.3, as she moves with her mobile device around the plant carrying out appropriate actions. The process that is carried out is depicted in Figure 1. Two goals can be achieved by the process, namely to produce product C or to produce product D. To produce product C a material (A) must be pumped into tank 1 using pump 1 (and the tanks involved must be empty for this process to be carried out successfully). Once tank 1 is full then pump 3 is put into forward mode (pump 3 is directional) to move the material from tank 1 to tank 2 thereby filling tank 2. The pumps then pause while tank 2 cooks the material, changing it from A to C. The flow of pump 3 is then reversed and tank 1, which had previously been emptied, is filled with the product. The final stage involves using pump 5 to remove the product from tank 1. The second goal is achieved in a similar manner. Tank 1 is also used in this process but this time it is fed from pump 2 and the cooking process takes place in tank 3 producing product D.

The questions that our analysis raises are (1) how do we arrive at plausible behaviors for achieving these two different goals? (2) given a specific proposal for the design as represented in a specification, how are these behaviors resourced and should

further features of the design be introduced in order to support the actions that the user must carry out?

The model that is illustrated in this paper is designed to demonstrate that a resource based approach will aid the process of design and the exploration of alternatives. The model describes the underlying process (a part of this process is described in the model of Figure 2). Figure 2 describes the bi-directional pumps (3 and 4). This timed automaton captures the actions that are supported by the pumps (*back?, forward?, off?, on?*), the type of material contained in the tanks represented at each side of the pump (*tk1t, tk2t*) and volume of material in the two tanks (*t1, t2*). It also models the time it takes to pump the material (*t*). This process information therefore reflects an abstraction of the actual state of the pumps and tanks and describes the actions that are available at any given state, regardless of how this information is resourced. This information combined with further aspects of the model, that will be discussed next, represent the system state without any concern for the interface to the operator. The focus has been how to provide a faithful though abstract description of the system.

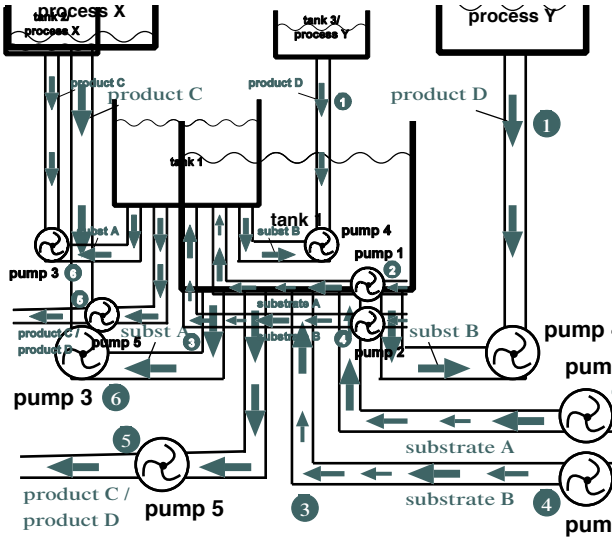

Figure 1: The process three tanks and five valves

The important feature of the model from the perspective of the paper is to capture those aspects of the system that combine resource information with the states and available actions. The model of the process as a whole should also include where the operator is in relation to the pumps that are distributed around the physical area of the plant. This information is captured by Figure 5. This model represents where the operator is (LCR represents the control room and LPi the location of each pump i). It represents the physical topology of the space in the sense that for example if the operator is near to pump 5 then it is possible to move to pump 3 or pump 1 without visiting any other locations. Hence in the case of this system the *possibilities* for

action will include where the operator is located (it might reasonably be assumed that the operator will know where the values are located in relation to these actions).

The most important part of the model from a resource point of view is the mobile device (see Figure 3, first described in [7]). The model in Figure 4 describes six types of interaction sequence. It simplifies the notion of its location in the sense that there is no notion of being in transit (*move?* moves from one location to another). All "download" (*download?*) actions mapped to the "component selector" therefore act on the location at which the device is and download the controls that are available for the proximal pump (see Figure 3). They appear in the larger display indicated (hence pump 1 is currently available). The switch (*switch?*) feature mapped to the "bucket selector" allows the operator to save controls for future use wherever the device is located. Hence in the example (Figure 3) pump 5 has previously been saved using a switch. It is possible in this case to switch again and make use of pump 5 controls. These features are modeled in Figure 4. The more complicated part of the model describes the actions that are supported by the different types of pump. Depending on the value of "valve" the operator is able to carry out actions that are appropriate to the type of model in the main display. Hence in the present example valve will have the value 1 and if the on button is selected then the model reaches a state (*dp*) where the actions available are all the actions available to the directional pump. These actions themselves control the model described in Figure 2.

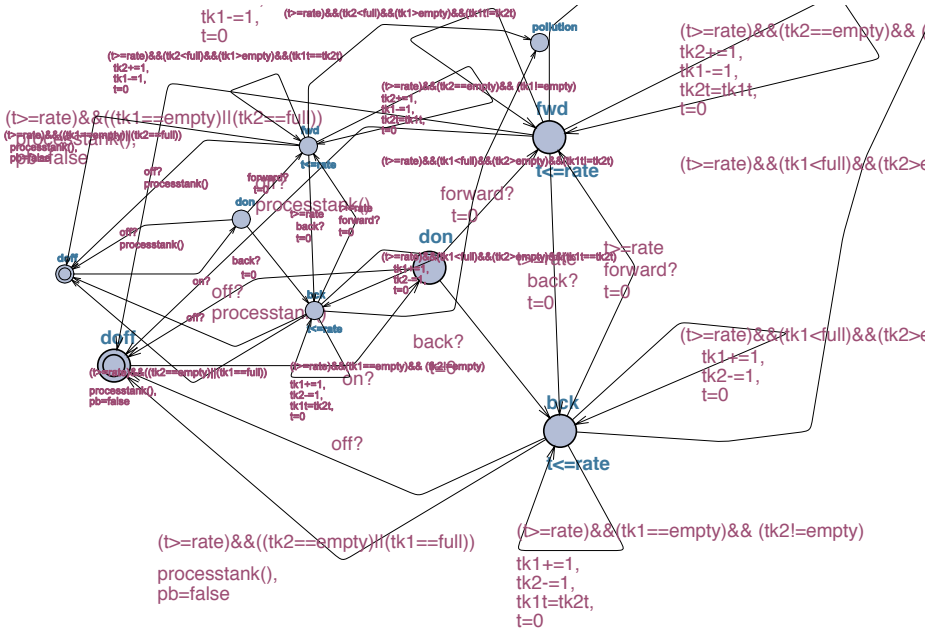

Figure 2: The bi-directional pump, note the fwd and bck states.

The first level of exploration of resourcing involves simply inspecting the models, identifying how the operator's activity is resourced. This involves asking questions

about the state of the system, whether the operator should be aware of the state and whether the possible actions appropriate to achieving a goal are clear in that state. In practice it would be feasible to label actions or states to emphasize the role that they play as resources as was discussed in [3]. In this particular case it makes sense to make distinctions between:

Movement actions that change the context of interaction, and the actions available via the mobile device. In this specification we have produced a separate model of location (Figure 5).

Downloading a control affects both the state of the device, and the available information for the end user.

Operating a control affects the state of the plant and also the device.

Reading the display does not affect the system or device models, but could affect the user model if one is included in the analysis. For an analysis based on Uppaal it is potentially convenient to include such actions to facilitate analysis within the tools.

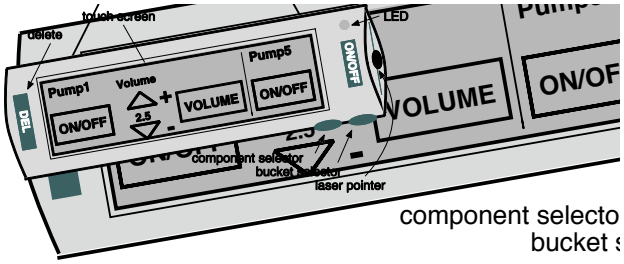

Figure 3: the hand-held device

The second level of exploration is described in Section 2.3 as the approach: *[Assumptions+Model+Goal+Starting situations -> Behaviors]*. Several assumptions have already been made in the model (for example assumptions about the location of the pumps and the nature of the underlying process). The starting situations are also assumed in the model, that the various tanks are empty for example. The process is iterative. Once behaviors have been considered this leads to the addition of further assumptions about the model to explore more "efficient" behaviors. The goal of producing product C is explored through the LTL property: `E<>((tank1==empty)&&(tank1m==C))`. This property is satisfied when pump 5 has been used to evacuate tank 1 and the type of the material is C. The model checker generates a trace in which the operator starts at LCP, visits LP4, then LP1 and uses pump 1 to fill tank 1. The operator then moves to LP5 followed by LP3, using pump 3 first to fill tank 2 from tank1 and then reversing the direction of the pump and filling tank1 from tank 2 with material C, and then going back to LP5 to evacuate tank 1.

The Uppaal system enables the designer to explore the path and at each step to explore each action, asking questions about how each step is resourced. How does the operator know which action to carry out to achieve the goal? Does the operator need to know the status of the process before deciding which pump to progress to? Does

the operator know where the relevant pump is? Does the operator know or need to know that the tank is empty? These questions suggest possible modifications to the interface.

Once this trace has been explored, the analyst should observe that this is not the most effective path to achieve the goal. In particular the operator does not make use of the switch facility and therefore it is necessary redundantly to revisit LP5. Further assumptions are therefore added to check that it is always possible to achieve the goal without unnecessarily revisiting pumps. This exploration was carried out by adding constraints to the model, where *updpath(i)* forces the operator to visit any location only once. The goal continues to be achievable and the path generated leads to further exploration of the resources required to encourage the operator to save the pump information at the relevant moment. Further analysis in relation to producing product D, when the locations are not revisited, produces a longer path than necessary. Further constraints enable exploration of shorter paths.

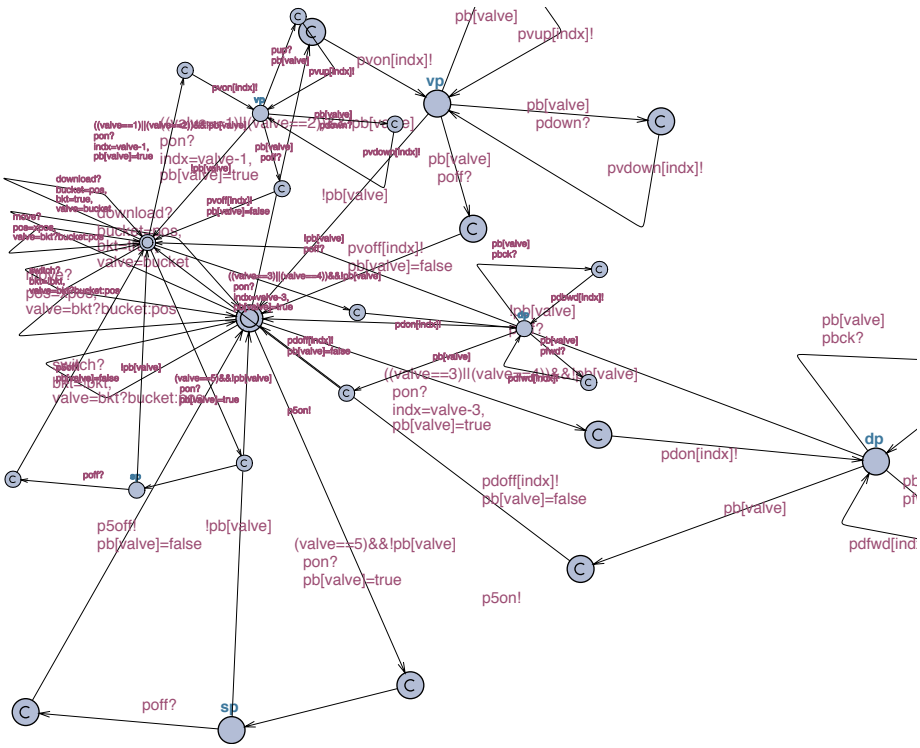

Figure 4: the model of the hand-held device

For each path the same questions are asked (corresponding to the list described above). In terms of *movement actions*, how do I know where to go? In terms of *switching a control*, is the save action enabled and visible, is it clear which control will be saved, is it clear what the effect on the device will be of saving the control? In terms of *resourcing for operating a control* is it clear that the action is enabled and visible, is it clear what the effect of the action will be? Appropriate information could

be specific values (the operator must know that Tank 2 contains product D), or simply that information is available (the operator can see the level within the tank, regardless of what the value is). The requirement for resourcing of the action of turning a pump on includes the system constraints on it being enabled, plus the mobile device having the pump loaded, plus the display showing the necessary information on the status of the pump.

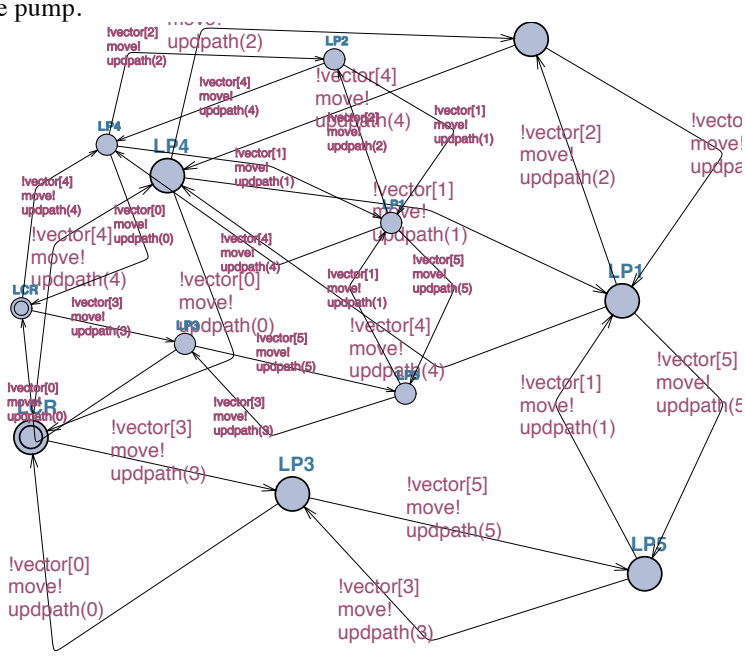

Figure 5: the model of the space.

4. Discussion

The approach presented has opened a number of avenues for further exploration.

Resources and visibility model - We can associate resource availability with particular states (as in Uppaal models), but direct support within the specification language would enable more explicit analysis. While visible state in interactor style models (such as MAL interactors [4]) provides a useful mechanism, support for dynamic visibility within the specification language would make the specifications easier to work with. A more sophisticated approach to the availability of resources would take into account the salience of information, for example visibility of information combined with goal relevance. Information may be potentially available, but the user may have to forage for it; such resource finding activity is much more plausible if

cues are provided to the user. For example, in systems in which display space is limited, and multiple actions are available, some interaction may be necessary in order to obtain action-effect information and this itself must be resourced.

Specialized analyses - Although we have concentrated on resource based analysis in the presence of intentional goal based behavior other analyses are advisable. Mode concerns continue to be important and are not revealed directly by the analysis described. Some mode errors will arise from insufficiently (externally) resourced actions, such as lack of mode indicators. Mode errors arising out of user confusions may require some consideration of internal resources, and user mental models. Conflicting activities often provide a setting which is conducive to mode error, and this would be a promising direction for future investigation. For example, where there are two goals to be achieved, opportunistic strategies for achieving both in an interleaved fashion could be explored.

Level of detail - Many analyses can be conducted looking simply at the configuration of information resources, without specifying precisely the content and associated application logic. While this is very attractive from the perspective of reducing the amount of specification and focusing the analysis on the aspects of interest, it is possible that some classes of problem will be missed as a result of this.

Interaction strategies, goals and resources - While we have dealt with the resourcing of actions as dependent only on the situation and the actions themselves, and while such cases are those of most interest to this paper, there are potential dependencies between the interaction strategy taken by the user, the different goals the user might have, and the resourcing of a particular strategy. This issue needs to be addressed in the context of the overall approach to analysis, and in particular the categorization of resources as part of the analysis. In terms of specifying required resources, this should be taken into account, but may also have an explicit role to play in the models (perhaps some requirements should be parameterized with respect to user goals). It would also be worth looking at the analysis in the context of a broader methodology such as DiCoT [2]. The three themes of the DiCoT analysis regarding physical layout, information flow, and use of artefacts all provide potential points of contact with the proposed approach, with tool support allowing us to investigate emergent properties of the space, the dynamic availability of information within an interaction, and the use of (resource-providing) artefacts which exhibit complex behaviour.

5. Conclusions

A conclusion from the example is that the approach appears to be a viable one, and seems to present some particular advantages when considering mobile systems. For situations with less clearly defined tasks or where there are many ways of performing a task, there is the obvious advantage over an analysis where there is no structure to user behavior. However, as can be seen above, even where there is structure to user tasks, the approach still presents advantages, as the focus of the analysis is quite

different, and there is no heavy specification overhead. We could also investigate situations where user behaviour arises from a mix of well defined tasks and more opportunistic goal-directed behaviour. The resourced-action based approach is attractive in that it considers opportunistic, situated actions, which are nonetheless purposeful, that is, they are directed towards some goal. Analyst insight obviously comes in to play in the resource analysis, but having an explicit activity can help to make this a more organized and concrete activity. While support for the analysis in tools has been considered, several issues regarding such support require further investigation, particularly support for more sophisticated visibility models, and tool support for more specific analyses (e.g. mode analysis).

Acknowledgments We acknowledge with thanks EPSRC grant EP/F01404X/1 and FCT/FEDER grant POSC/EIA/56646/2004. Michael Harrison is grateful to colleagues in the ReSIST NoE (www.resit-noe.org).

References

1. G. Behrmann, A. David, and K.G. Larsen. A tutorial on Uppaal. In M. Bernardo and F. Corradini, editors, Formal methods for the design of real-time systems, number 3185 in Springer Lecture Notes in Computer Science, pages 200-236. Springer-Verlag, 2004.
2. A. Blandford and D. Furniss. DiCoT: A Methodology for Applying Distributed Cognition to the Design of Teamworking Systems, Proceedings of DSV-IS '05, Lecture Notes in Computer Science, vol. 3941, pp. 26-38, Springer-Verlag, 2006.
3. J.C. Campos and G. Doherty, *Supporting resource based analysis of task information needs,* Proceedings of DSV-IS '05, Lecture Notes in Computer Science, vol. 3941, pp. 188-200, Springer-Verlag, 2006.
4. J.C. Campos and M.D. Harrison. Model checking interactor specifications. *Automated Software Engineering*, 8(3-4):275-310, August 2001.
5. E. Hutchins. Cognition in the Wild, MIT Press, 1995.
6. Loer, K. & Harrison, M.D. Analysing user confusion in context aware mobile applications. In proceedings Human-Computer Interaction - Interact 2005: IFIP TC13 International Conference. Lecture Notes in Computer Science 3585 pages 184-197, Springer-Verlag 2005.
7. J. Nilsson, T. Sokoler, T. Binder, and N. Wetcke. Beyond the Control Room: Mobile Devices for Spatially Distributed Interaction on Industrial Process Plants. Handheld and Ubiquitous Computing, Lecture Notes in Computer Science, Volume 1927, pp.1-30, Springer-Verlag, 2000.
8. PG Polson, C Lewis, J Rieman, C Wharton. Cognitive walkthroughs: a method for theory-based evaluation of user interfaces, International Journal of Man-Machine Studies, Volume 36(5), 1992.
9. L.A. Suchman. Plans and Situated Actions: The Problem of Human-Machine Communication, Cambridge University Press, 1987.
10. P.C. Wright, R.E. Fields. M.D. Harrison. Analyzing human-computer interaction as distributed cognition: the resources model. Human Computer Interaction, 15(1):1-42, 2001.