Reasoning Over Spatial Relations for Context-Aware Distributed User Interfaces

Petr Aksenov, Kris Luyten, and Karin Coninx

Hasselt University – transnationale Universiteit Limburg
Expertise Centre for Digital Media – IBBT
Wetenschapspark 2, 3590 Diepenbeek, Belgium
{petr.aksenov, kris.luyten, karin.coninx}@uhasselt.be

Abstract. Considering the amount of devices a user owns nowadays, a distributed user interface can become increasingly important. This requires reasoning techniques that allow making predictions of future values in the spatial model because these devices can be expected to change their location during usage. Our primary attention will be devoted to the problem of re-distribution of user interfaces in a constantly changing environment. So that a change in spatial topology, i.e. in the way the devices are located relative to one another, will be detected on time and interpreted in a proper way, resulting in redistribution of a user interface the devices are sharing.

1 Introduction

Distributed User Interfaces are user interfaces that can be considered as one logical whole but are distributed among a set of devices, often referred to as a device federation. Considering the amount of devices a user owns nowadays, a distributed user interface can become increasingly important. To get distributed, the user interface is split up in several parts and each part is allocated to a device from the device federation. A device federation contains two types of devices: stationary and mobile devices. For a distributed user interface we should be aware of other important properties of each device, such as the *interaction range* and the possibility and access rights for simultaneous access by several users. The former indicates the range within which a user can interact with the device. For example, a touch screen has a smaller interaction range than a microphone for speech input has. The latter indicates whether the device is shared among multiple users or is reserved for usage by one user only. This can have an important influence on the possible ways a user interface can be distributed over the devices of a device federation.

From a model-based point of view, a user interface is the presentation of a set of tasks the system supports. On a task level, interaction with the device

federation is the same as if the user interface were presented on a single device. As such, a distributed user interface is one logical whole despite the fact that it is spread over different devices. Furthermore, a distributed user interface is logically equivalent to a single-device user interface if it represents the same task model.

The work in this paper presents our initial step towards supporting distributed user interfaces for dynamic environments. We aim to enable reasoning over the spatial relationships between devices in a device federation. Spatial relations are described by a generic spatial model that structures the information so that it allows easy querying and reasoning. The presence of mobile devices in a device federation requires reasoning techniques that allow making predictions of future values in the spatial model because these devices can be expected to change their location during usage. A mobile device usually belongs to a user and is held by the same user but in the context of our work we can equally consider a user without one. If a spectator watching a presentation on a big screen on the wall decides at a certain moment to leave his current room and to continue watching the presentation in an adjacent room that has a display, this situation must also be treated correctly.

In our previous works we have created a set of software components that are required to realise effective distributed user interfaces. First, in [9, 17] we showed that using an XML-based language to describe the user interface made decomposition of the user interface for distribution much easier. In [16] a communication middleware for distributed heterogeneous environments was introduced. In addition, there is a wide variety of discovery and device identification solutions that can be used in combination with the aforementioned solutions. In [10], we extended the UPnP[1] protocol with location information of the devices being discovered which turned out to be a straightforward and useful way to set up an initial spatial model of the environment.

In particular, our primary attention will be devoted to the problem of re-distribution of user interfaces in a constantly changing environment. So that a change in spatial topology, i.e. in the way the devices are located relative to one another, will be detected on time and, what is more important, interpreted in a proper way, resulting in redistribution of a user interface the devices are sharing.

Since moving objects bring a degree of unpredictability into the behaviour of the system, an interesting problem is the prediction of their behaviour to support dynamic user interface distribution, i.e. analysing the steps the system has performed in order to try to predict the next possible steps. This allows us to determine a better configuration for redistribution among several available ones (e.g., a candidate for taking over a part of the user interface may be leaving the area as well), and it gives an opportunity to pay special attention to the mitigation of the impact of task interruption caused by a redistribution event.

1 http://www.upnp.org

The latter is important to ensure the usability of such a system: a task interruption is an important extra cognitive load for the user, which is already high because of the complexity of the environment [11].

2 Related Work

In [1] the authors presented a reference model for distributed, migratable and plastic user interfaces together with a middleware for integration of such interfaces in a heterogeneous environment. This kind of middleware is necessary for the successful distribution of a user interface in an ambient environment. [4] provides a reference model for reasoning about distributed UIs. Several basic types of distribution were illustrated with the help of exemplary discussions of the model which were supposed to give clues on reasoning about suitable ways of distributing the UI.

In [6], Gostner gives a detailed overview of the spatial concepts, such as spatial knowledge, spatial relations and spatial awareness, and also discusses context awareness in general. This work helped to gain better insight about the diversity and the level of complexity of the spatial information in a computing environment. In [8], Kortuem et al. deal with the problem of utilising spatial information in creating new types of user interfaces and used a graph to model spatial arrangements of the system. The graph represented the spatial infrastructure of a system at a certain moment of time, and the system was then described over time by a sequence of these graphs. From this perspective, the proposed scheme is a very similar approach to that what we use in our work for the same purpose.

A lot of research has been done to address the problem of modelling context with the help of ontologies. [3] provides a way of modelling human activity in which context was represented by a set of roles and relations the user may have and an ontology for context awareness was constructed using these concepts. [14] introduces the “Aspect-Scale-Context” model to deal with the context information and presented the CoOL language as a collection of projections of three other ontology languages, which could allow means for contextual exchange in a distributed system architecture. In their other work, the authors demonstrated the application of CoOL to make service interaction architectures context-aware. [2] describes CoBrA, a framework for a pervasive computing environment resulting from a set of common OWL-based ontologies. This model is built around a centralised server called Context Broker that is used to store and reason about context information. [19] proposes an upper-ontology called CONON to model context in a pervasive environment and provided the ways this ontology can be extended to specific ontologies. In addition, they made some observations about the feasibility of ontological reasoning, which

complexity could be crucial to time-critical applications. Both [7] and [15] used probabilistic approach to address the problem of uncertainty in ontology-based reasoning about contextual information in pervasive environments. However, most of the models and reasoning techniques were introduced in order to be implemented on a specific domain and none of them have explicitly addressed the problem of utilising the context to reason about the distribution.

3 Description of the Problem Domain

The structure of a system is much more complicated than just a matter of location and timing. Computing objects consume power, communicate via various network connections, have varying capabilities, are configured according to a specific user profile, etc. All of it may influence the way the user interface is distributed among these computing objects. We focus on the spatial relationships and dependencies in this paper since we believe there are many challenges here that deserve attention. Since we started our approach from the model-based interface development domain, the techniques that are developed to deal with changes in spatial relationships fit in a larger overall framework. Spatial relationships and dependencies are merely an aspect of the other models that are often used in model-based interface development. When talking about the spatial (sub)model, it is important to note that other models commonly used to describe an interactive system (e.g. the device model, the task model, the presentation model, and the user model) might also contain information of spatial nature and this information can also be taken into account when creating a distributed user interface. So that the very word “spatial” has been chosen to emphasise that only spatial relations among the diversity of all relations existing in the environment come into play in the work we present here.

We will be using a scenario of distribution of the user interface of an image viewer to illustrate our work with a practical example. We assume that the image viewer is an application that supports the display of images, switching to the next or the previous image in a collection, zooming in and out, and has an extra functionality to add comments on the image currently displayed, and the comments are assumed to be always displayed together with the image. Thus, the user interface consists of four components which can be split among a set of up to four devices. There are four logical units in our environment: a projector and display that are considered together as one static unit, a laptop, and two PDAs. Initially, the distribution is as follows: images are projected on the display, the switching and zooming capabilities are each distributed to one of the PDAs, and the laptop is used to deal with comments. The PDAs and the laptop can move, thus setting a target of preserving any of the functionalities of the complete user interface in the case when any of the three mobile devices

becomes unreachable. For example, when the PDA responsible for the zooming bit is leaving the area, the application must be prepared to add this part of the interface to either the remaining PDA or the laptop.

3.1 Graph of Spatial Relations

Among many possible ways of describing the state of a set of objects located in space, the graph-based graphical representation seems to be the most convenient one. It allows us to create visualisations that are easy to understand and provides us with a data structure that is well known for reasoning and transformation purposes. In our work a system of interacting objects is presented as a graph. One possible freely gained advantage of such representation is that the graph theory is a very well studied area of computing and therefore extra opportunities about reasoning, optimisation, etc. may come into play.

A graph can be considered as a data structure that consists of a set of nodes and a set of edges which stand for relationships between the nodes. In our approach, each node corresponds to a certain device that exists in the environment and is considered a hypothetical interaction resource, and each edge stands for a spatial relationship between the two corresponding nodes. A node may also represent a person, since, as we have discussed before, there can be a user without a device. An example of such a graph is shown in Fig. 1, depicting the initial situation of the image viewer application scenario.

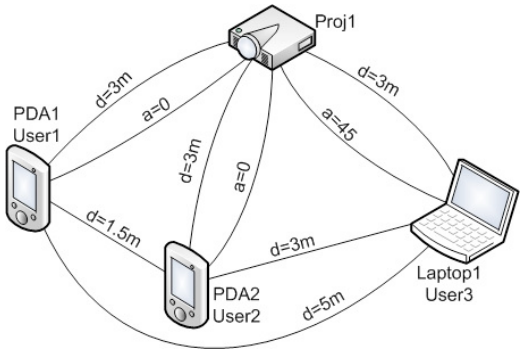

Fig. 1. Graph-like representation of spatial relationships.

To understand and predict the behaviour, it is necessary to observe how the graph of a given system changes over time. Overall, the following changes can happen to the graph: 1) a new node/edge appears, 2) an existing node/edge disappears, 3) the value of a property of a graph object – either a node or an edge

– changes. The changes of types 1) and 2) are qualitative since the corresponding graph gets a new structure whereas the change of type 3) is quantitative since it preserves the graph topology. It is important to note that no time constraints are taken into account, and the only meaningful time factor is that of differences of spatial nature that have taken place in the system at another moment of time.

3.2 Device Availability Function

As a direct consequence of the graph approach, we become able to define a function that measures the degree of availability of each device.

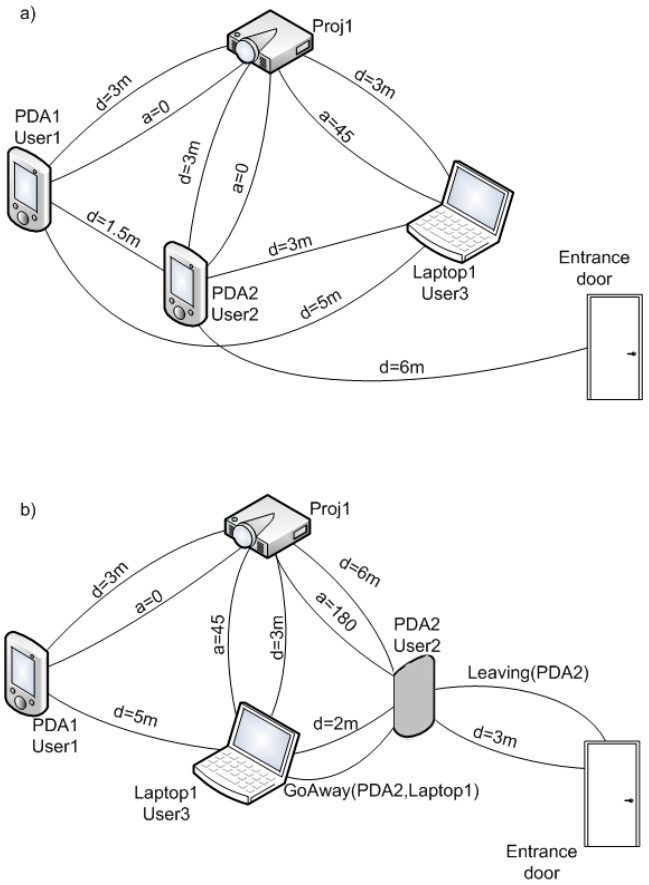

Fig. 2. (a) graph-of a system at time t_1; (b) graph of the same system at time $t_2 > t_1$.

To make the behaviour of this function more precise, consider the two situations presented in Fig 2. At time t_2 PDA2 has changed its location and is now three metres further from the projector, i.e. three metres closer to the entrance door. At some other moment t_3 PDA2 is already five metres away from the projector and is two additional metres closer to the entrance door. Now, if we consider other time moments between t_1 and t_2 as well as between t_2 and t_3, we can draw a dependency of the distance between PDA2 and the entrance door over time (see Fig. 3(a)), and then, using the already known values, assume the expected behaviour of PDA2 from the point of view of leaving the room (see Fig. 3(b)).

Analysing the plot in Fig. 3(b), we may conclude that device PDA2 is leaving the area and will be out of range at time t_4 onwards. Since it currently owes a part of the user interface of the image viewer application, the application must start deciding where to transfer PDA2's bit of the user interface prior to the moment when PDA2 has left the room. So that by the time it has gone, the viewing process will continue in full functionality.

However, extrapolation alone cannot give us the entire view of the situation since the behaviour can, for example, simply be a result of the peculiarity of the device in the current environment. We must also look at the orientation of the devices relative to other devices or objects in the area. In Fig. 2(b) PDA2 is depicted with its back in the front, which means that the device is no longer oriented towards the display and thus User2 is not looking at it. This fact is also shown in the change of the value of *a*, the orientation angle between Proj1 and PDA2, from 0 to 180. More discussion on some other aspects that could be important in predicting the behaviour is also given later in section 4.3.

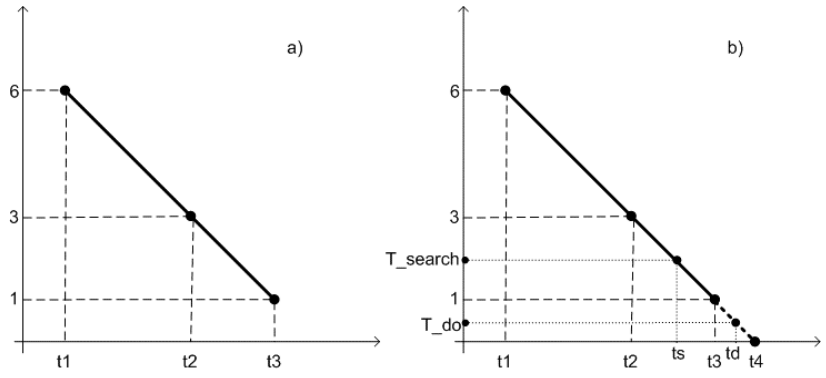

Fig. 3. (a) dependency of *D(PDA2,door)* between t_1, t_2, and t_3; (b) extrapolating *D* to time t_4.

The main difficulty is to decide how to represent the continuity of time using discrete time intervals: these intervals need to be small enough so that within an interval the changes to the context can be neglected. We say that a time interval $[t_1,t_2]$ has static context, which means the same description graph is applicable between t_1 and t_2. This has some implications on the validity of our approach for runtime reasoning. However, this approach is valid for our goals since we are currently targeting the *prediction* of required UI redistribution events rather than the redistribution strategies required to effectively present a distributed user interface.

The described approach is of primary importance to an environment with many moving objects, so that the application does not redistribute the user interface among the devices that are going to leave the area as well. This way, it allows fulfilling the main purpose of the current work: determine the proper redistribution of a user interface among available devices with a minimal impact on the currently active distribution in the running applications. We see this purpose divided into two subsequent ones of finding out the best answers to the questions of 1) *when* to begin looking for new candidates for taking over a part of the user interface, and 2) *when* to start redistribution.

3.3 Device Importance

Two thresholds will be introduced on the device availability function: one for each of the "when's" (*T_search* and *T_do* in Fig.3(b)). In this regard, it is necessary to understand how critical the loss of a particular device is to a system, and this is the concept of the importance of device that matters. Importance can be treated as irreplaceability of a certain device to the current system, thus implying a different scale grade applied to the spatial changes happening to this device. This concept will be implemented in a set of factors crucial to the situation in question. For example, it may include the number of active devices of the same kind in the environment, the possibility to transfer a part of the user interface to a device of another kind. A higher threshold value can then be set to a redistribution work to start in the case of a very important device leaving.

We have mentioned in the introduction that the human aspect is the most important factor in building a successful application that can be accessed using a distributable interface. Automatic changes in user interface configuration can confuse the user and often make an application harder to use because of an increased cognitive load. In this case, even the physical load of using the interface could increase to an uncomfortable level since we aim for redistribution of a user interface over space. We plan to study the effects of user interface (re)distribution on the usability of an application and use these findings to adapt the distribution strategies that are implemented. In addition, a mixed

initiative approach can be helpful to keep the user in control [18] and avoid situations in which the distribution results in a system that is no longer accessible.

4 Reasoning Approach

In this section we discuss different reasoning techniques which we find suitable for describing spatial relations in our system.

4.1 Ontology

Ontologies are widely used in the domain of semantic web. The underlying concept behind ontology is description logic, a language for representing the information about a system in a structured way.

In section 2 we noted the diversity of approaches for modelling context and for building ontologies to reason about context in a pervasive and ubiquitous computing environment. But usually such models as well as the corresponding ontologies are discussed in full aspect, i.e. various aspects of modelling the situation are considered. Our work does not aim at covering the ontological part as a whole but rather tries to utilise the spatial bit wherever possible and relevant. And it is not trivial to give an answer to the question of which model suits the problem of distribution in terms of spatial relations better. Therefore we believe that the most valuable approach is not to confine oneself to one particular context model, but to absorb the advantageous points of other approaches with respect to the problem we are trying to solve and come up with our own ontological view of the problem that could fit with these approaches.

In addition, there are techniques, such as RCC8 [13] and the 9-intersection model [5], that are specifically meant to reason about spatial relations between regions in a topological space and that have separated into important fields of study within the area of spatial reasoning. In general, these could also be involved in developing an ontology for spatial relations, but they deal with a different level of granularity than we have adopted in our work at a current step, and therefore seem to be of little help, if not out of scope.

4.2 Fuzzy Logic

Since we are to deal with uncertainty to a large extent in our work, description logic only will not be enough. To model imprecision, other techniques are required, and fuzzy logic is one of them.

Fuzzy logic [20] is an extension of the concepts of classical logic and set theory that goes beyond the usual TRUE and FALSE values and introduces the concept of membership of an element to a certain set. For example, if a quarter of the glass is filled with water, we may say that it is 0.25 full and 0.75 empty. Furthermore, this division is rather subjective, for the glass may be considered empty if it is filled with no more than one-tenth of its volume as well as all glasses that have more than 90% of their volume filled with water are said to be full.

From our point of view, fuzzy logic is a binder between the device importance and the threshold values. It is important to note here that in spite of the fact that the thresholds are applied on the plot of the device availability function in order to determine the two critical moments for redistribution events, they depend on the system infrastructure and are determined irrespective of the device availability function behaviour. To clarify on this, consider two devices with identical availability functions but different results of their importance calculations. The application of the fuzzy logic concepts to one device returns the threshold set up at x metres off the door, and at $x+5$ metres for the other. Or, using fuzzy logic terms, one device would be considered a device that is leaving when it gets 5 metres closer to the door than the other.

4.3 Probabilistic Reasoning

We also want to involve probabilistic logic [12] into our reasoning scheme. This type of reasoning will allow concentrating on the aspect of redistribution itself by realising what to consider and what not to. The device availability function will be of help.

In our work, probabilistic reasoning will aim at determining how likely it is that the device is going to become inactive. This is expected to be done by means of analysing the device's previous behaviour, i.e. its availability function. In other words, we want to try to predict how probable it is that the device will behave in space in accordance with the extrapolated part of the availability function.

Although probabilistic and fuzzy logic may really look alike, there is a considerable difference between them: probabilistic logic will be involved in making predictions and the fuzzy one will not. Fuzzy logic says what the distribution thresholds are for a device, whereas probabilistic reasoning tries to determine the likelihood of these and similar events.

We have already mentioned in section 3.2 that extrapolating alone is not sufficient for looking at the behaviour of the device. There are many other factors that have an impact on it. If the user wants to occupy a chair situated near the door but not exactly in the doorway and is moving towards it then the device availability function will show behaviour very close to that of a leaving device.

But the actual situation will be different due to an external factor (from the device's point of view) and the probability of matching the predicted behaviour should decrease. Taking into account this kind of subtleties is another means by which we want to apply probabilistic reasoning in our work.

5 Case Study

Consider two distributions in Fig.2(a): i) PDA2 has the zooming functionality, ii) PDA2 has the navigating functionality. These two situations are to get the same device availability function for two devices with different importance values. In both distributions the measurements of device importance (*di*) resulted in the following values for the interacting devices: di(Projector)=3, di(Laptop)=2, di(PDA_Nav)=1.6, and di(PDA_Zoom)=1.3. For the PDAs, this could be, for example, because the zoom panel is easier to get migrated to another device, (and) it is easier to append it to the existing interface of one of the other interacting devices, (and) it is less severe for the core functionality of the application, (and) etc.

Then we calculate two thresholds: *T_search* determines when to start looking for possible distributions, and *T_do* determines when to commence on the redistribution work. These thresholds are *context-sensitive*: they are computed in the beginning of each time interval and are based on device importance. For simplicity, they do not change in our example, but in general these are dynamic values that might change, for example, when a new device enters in the vicinity of the user. The computation of the thresholds for PDA_Nav results in T_search=2.5 metres off the door and T_do=0.8 metres off the door whereas for PDA_Zoom the values are T_search=2 metres and T_do=0.5 metres, respectively. If we now apply the thresholds (for example the PDA_Zoom's ones) on the device availability function (see Fig.3(b)), we will get the time values corresponding to the moments when the system should start looking for a new distribution (t_s) and when to actually start it (t_d). Then for all time values in the interval [0; t_s] the device is considered to be fully available, for the time values later than t_d – fully unavailable, and for all time values between t_s and t_d the device is considered partially available with different degrees of availability within the time interval. Thus we use the terms of fuzzy logic to determine which device and in which conditions belongs to one or another set of availability, how the sets' boundaries change, and in which way the membership flows.

The second step considers behaviour of PDA2 which holds, for example, the zooming interface. There will also be two cases. In the first case the spatial graph is as in Fig.2(b) and application of probabilistic logic to the context of time t_3 resulted in the value of 93% that the device will behave as the availability

function extrapolated. In the second case, there is a chair very near the entrance door and the probability calculations returned 71% because there is a spatial relation between User2 and the chair. This relation is external for PDA2, i.e. it is unable to influence the device's availability function, but it means that the user may just be approaching the chair in the next moments. Generalising the spatial relation between User2 and the chair to a relation between any two objects, we can introduce the concept of weight of a relation. It is clear that the weight of the relation "User2-chair" is big whereas the weight of the relation "chair-Proj1" is probably 0. Similarly, the relation "Proj1-EntranceDoor" is meaningless since the devices do not interact. The concept of weight could be incorporated into our graph model. The idea behind the probabilistic reasoning is that on the base of different probability values the system will set different priorities to the redistribution events for every leaving device.

6 Future Work

Elaborating the concept of the importance of device is one of our nearest goals. We have already given above a couple of factors that can be included, but in order to come up with an extensive list of determinants, a situation must be investigated from numerous viewpoints.

The other crucial point to the work is the proper definition of the availability function. This function is not about mere distances, but also about the specifics of the device. However, the device specifics should not be mixed with the factors which are not directly related to the device since the latter, as we have stated above, is a matter of probabilistic reasoning.

As a possible improvement of a redistribution event, it may be promising to make the application learn from the previous system behaviour and the resulting decisions about redistribution and optimise the redistribution algorithm.

7 Conclusions

In this paper we propose a way to address the problem of redistribution of user interfaces in a dynamic computing environment by means of predicting the possible behaviour of the system. We use the spatial relations existing in the environment for making appropriate predictions. Since the context of spatial nature is considered to be of major importance, our approach will allow for a more detailed investigation of the problem. To reason about spatial context, we suggest using a combination of different reasoning techniques, including those capable of dealing with uncertainty, the concept crucial to ubiquitous dynamic computing environments.

Acknowledgments. Part of the research at EDM is funded by EFRO (European Fund for Regional Development) and the Flemish Government. Funding for this research was also provided by the Research Foundation -- Flanders (F.W.O. Vlaanderen, project CoLaSUE, number G.0439.08N).

References

1. Balme, L., Demeure, A., Barralon, N., Coutaz, J., Calvary, G.: CAMELEON-RT: A Software Architecture Reference Model for Distributed, Migratable, and Plastic User Interfaces. In: Markopoulos, P., Eggen, B., Aarts, E.H.L., Crowley, J.L. (eds.) EUSAI2004, LNCS, vol. 3295, p. 291–302, Springer, Berlin (2004)
2. Chen, H., Finin, T., Joshi, A.: Semantic Web in the Context Broker Architecture. In: Proc. of the 2nd IEEE International Conference on Pervasive Computing and Communications (PerCom2004), pp.277–286, IEEE Computer Society Press (2004)
3. Crowley, J., Coutaz, J., Rey, G., Reignier, P.: Perceptual Components for Context Aware Computing. In: Borriello, G., Holmquist, L.E. (eds.) UbiComp 2002, LNCS, vol. 2498, pp.117–134, Springer-Verlag, London (2002)
4. Demeure, A., Calvary, G., Sottet, J-S., Vanderdonckt, J.: A Reference Model for Distributed User Interfaces. In: Proc. of the 4th International Workshop on Task Models and Diagrams (TAMODIA '05), pp. 79–86, ACM Press, New York (2005)
5. Egenhofer, M.J., Herring, J.: Categorizing Binary Topological Relations between Regions, Lines and Points in Geographic Databases. Technical Report, Department of Surveying Engineering, University of Maine (1991)
6. Gostner, R.: Collaboration with Spatial Aware UIs. First Year Report, Lancaster University (2007)
7. Gu, T., Pung, H., Zhang, D.: A Bayesian Approach for Dealing with Uncertain Contexts. In: Advances in Pervasive Computing, Austrian Computer Society, vol. 176, ISBN 3-85403-176-9, Vienna (2004)
8. Kortuem, G., Kray, C., Gellersen, H.: Sensing and Visualizing Spatial Relations of Mobile Devices. In: Proc. of the 18th Annual ACM Symposium on User Interface Software and Technology (UIST2005), pp. 93–102, ACM Press (2005)
9. Luyten, K., Vandervelpen, C., Coninx, K.: Migratable User Interface Descriptions in Component-Based Development. In: Forbrig, P., Limbourg, Q., Urban, B., Vanderdonckt J. (eds.) DSV-IS2002, LNCS, vol. 2545, pp.44–58, Springer-Verlag, London (2002)
10. Luyten, K., Van den Bergh, J., Coninx, K., Vandervelpen, C.: Designing Distributed User Interfaces for Ambient Intelligent Environments Using Models and Simulations. Computers and Graphics 30(1), pp.702–713 (2006)
11. Massink, M., Faconti, G.: A Reference Framework for Continuous Interaction. Universal Access in the Information Society 1(4), pp.237–251 (2002)
12. Nilsson, N.J.: Probabilistic Logic. Artificial Intelligence 28(1), pp. 71–87 (1986)
13. Randell, D.A., Cui Z., Cohn, A.G.: A Spatial Logic Based on Regions and Connections. In: Proc. of the 3rd International Conference on Knowledge Representation and Reasoning, pp. 165–176, Morgan Kaufman, San Mateo (1992)
14. Strang, T., Linnhoff-Popien, C., Frank, K.: CoOL: A Context Ontology Language to Enable Contextual Interoperability. In: Stefani, J.-B., Demeure, I.M., Hagimont, D. (eds.) DAIS2003, LNCS, vol. 2893, pp. 236–247, Springer (2003)
15. Truong, B.A., Lee, Y.-K., Lee, S.: A Unified Context Model: Bringing Probabilistic Models to Context Ontology. In: Enokido, T., Yan, L., Xiao, B., Kim, D., Dai Y.-S., Yang, L.T. (eds.) EUC2005, LNCS, vol. 3823, pp. 566–575, Springer (2005)

16. Vanderhulst, G., Luyten, K., Coninx, K.: Middleware for Ubiquitous Service-Oriented Spaces on the Web. In: Proc. of AINA Workshops 2007, pp. 1001–1006, IEEE Computer Society Press (2007)
17. Vandervelpen, C., Vanderhulst, G., Luyten, K., Coninx, K.: Light-weight Distributed Web Interfaces: Preparing the Web for Heterogeneous Environments. In: Lowe, D., Gaedke, M. (eds.) ICWE2005, LNCS, vol. 3579, pp. 197–202, Springer (2005)
18. Verpoorten, K., Luyten, K., Coninx, K.: Mixed Initiative Ambient Environments: A Self-Learning System to Support User Tasks in Interactive Environments, In: Proc. of 3rd Workshop on Context Awareness for Proactive Systems (CAPS 2007), Guildford, UK
19. Wang, X., Zhang, D., Gu, T., Pung, H.: Ontology Based Context Modeling and Reasoning using OWL. In: Proc. of PerCom Workshops 2004, pp. 18–22, IEEE Press (2004)
20. Zadeh, L.A.: Fuzzy sets. Information and Control 8(3), pp. 338–353 (1965)