# Dynamic Composition of Process Federations for Context Aware Perception of Human Activity

James L. Crowley and Patrick Reignier
Laboratoire GRAVIR-IMAG, INRIA Rhône-Alpes
Grenoble, France

**Abstract**—*This paper describes a distributed software model for context-aware perception of human activity. The basic building blocks in this model are perceptual modules, composed of a data transformation component and a control component. Modules are assembled into perceptual processes controlled by a reflexive process controller. Process controllers regulate computation, and provide a reflexive description of their internal state and capabilities. Explicit models of context are used to assemble federations of processes for observing and predicting activity. As context changes, the federation is restructured. Restructuring the federation enables the system to adapt to a range of environmental conditions and to provide services that are appropriate over a range of activities.*

## 1. Introduction

In this paper, we describe a data-flow architecture based on dynamically assembled federations [1], [2]. Our model builds on previous work on process-based architectures for machine perception and computer vision [3], [4], as well as on data flow models for software architecture [5]. We propose a model in which a user's context is described by a set of roles and relations. A context is translated into a federation of processes for observing the entities that satisfy roles as well as the relations between these entities. This model leads to an architecture in which reflexive elements are dynamically composed to form federations of processes for observing and predicting the situations that make up a context. As context changes, the federation is restructured. Restructuring the federation enables the system to adapt to a range of environmental conditions and to provide services that are appropriate over a range of activities. The result is a software architecture for building systems that act as a silent partner to assist humans in their activities in order to provide appropriate services without explicit commands and configuration.

## 2. Modules, Processes and Federations

The most basic unit in our system is a module. A module us defined as a transformation applied to a synchronous data stream of to asynchronous events. The transformation may depend on a set of parameters. The data stream may be accompanied by meta-data. In our model, all modules are designed with the capability report on their state. Examples of module state include computation time and quality of result. Module state is discussed below.

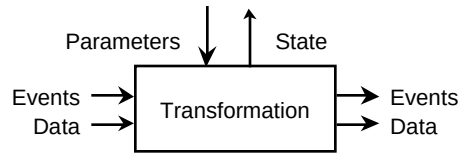


**Fig. 1**. Modules are defined as transformation over events and data.

Modules are assembled into processes, shown in figure 2.

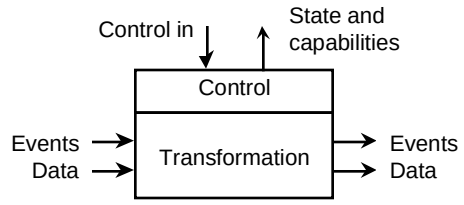


**Fig. 2.** A Observational process combines transformation with a control component.

A process has two functional facets: A transformation component and a control component. As with modules, the transformation component may be defined to transform data received in a synchronous stream or asynchronous events. The transformation component of a process is generally a composition of transformations provided by modules. The input data to the transformational component is generally composed of some raw numerical values, generally arriving in a synchronous stream, accompanied by meta-data. Meta data includes information such as a time-stamp, a confidence factor, a priority or a description of precision. An input event is a symbolic message that can arrive asynchronously and that may be used as a signal to begin or terminate the transformation of the input data. Output data and the associated meta-data is a synchronous stream produced from the transformation of the input data. We also allow the possibility of generating asynchronous output messages that may serve as events for other processes. This model is similar to that of a *contextor* [28], which is a conceptual extension of the context widget implemented in the Context Toolkit [29].

The control component of a process enables reflexive control of observational processes and thus provides a number of important functions. The control component receives commands and parameters, supervises the execution of the

transformation component, and responds to queries with a description of the current state and capabilities.

Figure 3 shows an example of a process for observing skin colored regions, using a robust tracking algorithm [30]. A probabilistic skin detection module transforms a color image into an image in which each pixel represents the probability of skin. Regions of probabilities are grouped into blobs described by their first and second moments. These blobs are then tracked using a recursive tracking process based on a Kalman Filter.

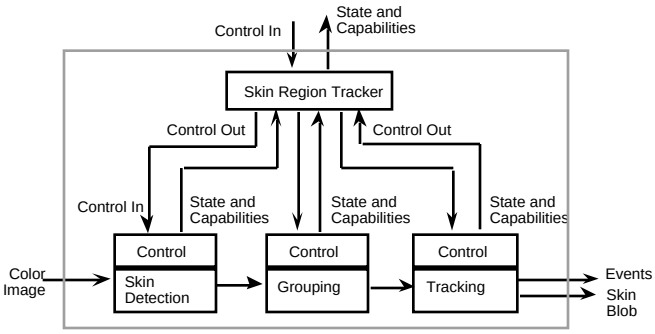


**Fig 3.** Processes for observing skin colored blobs using robust tracking.

A process federation is assembled by a supervisor controller, as illustrated in figure 4. Supervisory controllers invoke and configure processes to perform the transformations required to observe a context. The states of processes are monitored by the supervisory controller and process parameters are adapted in response to events.

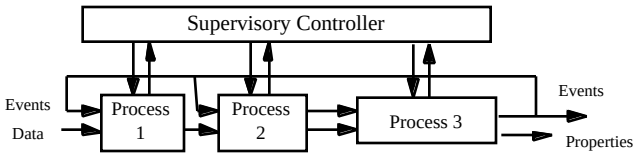


Fig 4. A process federation is assembled and controlled by a supervisory controller.

Supervisory controllers may be assembled into hierarchies in order to observe human activity. The exact assembly depends on the task that the system is to perform as described by a model of the users task and context.

## 3. CONTEXT AND SITUATION.

The context for a user and task is a composition of situations. Situations are defined by a configuration of a set of entities, roles and relations. A context model specifies the collection of roles and relations to observe, and thus the process federation that must be invoked. Process federations are created to observe the relevant entities, to assign entities to roles, and to determine relations for entities assigned to roles.

### *3.1 A Brief History of Context*

Winograd [6] points out that the word "Context" has been adapted from linguistics. Composed of "con" (with) and "text", context refers to the meaning that must be inferred from the adjacent text. Such meaning ranges from the references intended for indefinite articles such as "it" and "that" to the shared reference frame of ideas and objects that are suggested by a text. Context goes beyond immediate binding of articles to the establishment of a framework for communication based on shared experience. Such a shared framework provides a collection of roles and relations with which to organize meaning for a phrase.

Early researchers in both artificial intelligence and computer vision recognized the importance of a symbolic structure for understanding. The "Scripts" representation [7] sought to provide just such information for understanding stories. Minsky's Frames [8] sought to provide the default information for transforming an image of a scene into a linguistic description. Semantic Networks [9] sought to provide a similar foundation for natural language understanding. All of these were examples of what might be called "schema" [10]. Schema provided context for understanding, whether from images, sound, speech, or written text. Recognizing such context was referred to as the "Frame Problem" and became known as one of the hard unsolved problems in AI.

In computer vision, the tradition of using context to provide a framework for meaning paralleled and drew from theories in artificial intelligence. The "Visions System" [12] expressed and synthesized the ideas that were common among leading researchers in computer vision in the early 70's. A central component of the "Visions System" was the notion of a hierarchical pyramid structure for providing context. Such pyramids successively transformed highly abstract symbols for global context into successively finer and more local context terminating in local image neighborhood descriptions that labeled uniform regions. Reasoning in this system worked by integrating top-down hypotheses with bottom-up recognition. Building a general computing structure for such a system became a grand challenge for computer vision. Successive generations of such systems, such as the "Schema System"[13] and "Condor" [14] floundered on problems of unreliable image description and computational complexity. Interest in the 1990's turned to achieving real time systems using "active vision" [15], [16]. Many of these ideas were developed and integrated into a context driven interpretation within a process architecture using the approach "Vision as Process" [17].

The term "Context Aware" was introduced to the mobile computing community by Schilit and Theimer [18]. In their definition, context is defined as "the location and identities of nearby people and objects and changes to those objects". While this definition is useful for mobile computing, it defines context by example, and thus is difficult to generalize and apply to other domains. Other authors, such as [19] [20] and [21] have defined context in terms of the environment or situation. Such definitions are essentially synonyms for context, and are also difficult to apply operationally. Cheverest [22] describes context in anecdotal form using scenarios from a context aware tourist guide. His system is considered one of the early models for a context aware application.

Pascoe [23] defines context to be a subset of physical and conceptual states of interest to a particular entity. This definition has sufficient generality to apply to a recognition system. Dey [24] reviews definitions of context, and provides a definition of context as "any information that can be used to characterize situation". This is the sense in which we use the term context. Situation refers to the current state of the environment. Context specifies the elements that must be observed to model situation. However, to apply context in the composition of perceptual processes, we need to complete a clear semi-formal definition with an operational theory.

*3.2 Entities and Relations*

A fundamental aspect of interpreting sensory observations is grouping observations to form entities. Entities may generally be understood as corresponding to physical objects. However, from the perspective of the system, an entity is an association of correlated observable variables. This association is commonly provided by an observational process that groups variables based on spatial co-location. Correlation may also be based on temporal location or other, more abstract, relations.
Thus, an entity is a predicate function of one or more observable variables.

Entity-process($v_1$, $v_2$, …, $v_m$) Entity(Entity-Class, ID, CF, $p_1$, $p_2$,…, $p_n$)

Entities may be observed by an entity grouping processes, as shown in figure 5.

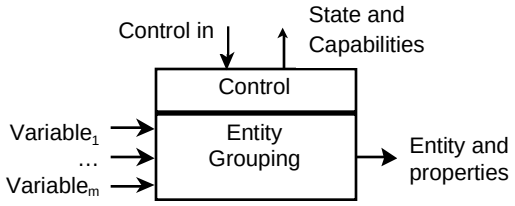


**Fig 5.** Entities and their properties are detected and described by entity grouping processes.

The input to an entity grouping process is typically a set of streams of numerical or symbolic data. The output of the transformation is a stream including a symbolic token to identify the kind of the entity, accompanied by a set of numerical or symbolic properties. These properties allow the system to define relations between entities. The detection or disappearance of an entity may, in some cases, also generate asynchronous symbolic signals that are used as events by other processes.

A fundamental aspect of interpreting sensory observations is determining relations between entities. Relations can be formally defined as a predicate function of the properties of entities. Relations that are important for describing context include 2D and 3D spatial relations, as well as temporal relations [32]. Other sorts of relations, such as acoustic relations (e.g. louder, sharper), photometric relations (e.g. brighter, greener), or even abstract geometric relations may also be defined. As with observable variables and with entities, we propose to observe relations between entities using observational processes. Such relation-observation processes are defined to transform entities into relations based on their properties, as illustrated in figure 6.

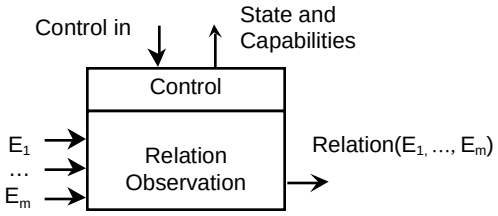


**Fig 6.** Relations between entities are detected by relation detection processes

As before, this transformation may be triggered by and may generate asynchronous symbolic messages that can serve as asynchronous events.

Relation-observation($E_1$, $E_2$, …, $E_m$) (Relation-Class, ID, $E_1$, $E_2$,…, $E_n$)

The concept of role is perhaps the most subtle concept of this model. Entities may be assigned to roles based on their properties. Thus roles may be seen as a sort of "variable" placeholder for entities. Formally roles are defined as entities that enable changes in situations. Such a change corresponds to an event.

When an entity enables an event, it is said to be able to "play" the role. An entity is judged to be capable of playing a role if it passes an acceptance test based on its properties. For example, a horizontal surface may serve as a seat if it is sufficiently large and solid to support the user, and is located at a suitable height above the floor. An object may serve as a pointer if it is of a graspable size and appropriately elongated. In the user's environment, pens, remote controls, and even a wooden stick may all meet this test and be potentially used by the user to serve the role of a pointer.

The set of entities that can provide a role may be open ended. In the users' context, the user determines if an entity can satisfy a role for a task by applying the acceptance test. The system may anticipate (and monitor) such entities based on their properties. In the system's context, the system may assign entities to roles. Such assignment is provided by a process that applies a predicate function defined over entities and their properties.

Role($E_1$, $E_2$, …, $E_m$) (Role-Class, ID, CF, $E_1$, $E_2$,…, $E_n$)

When the test is applied to multiple entities, the most suitable entity may be selected based on a confidence factor, CF.

The set of entities is not bijective with the set of roles. One or more entities may play a role. A role may be played by one or several entities. The assignment of entities to roles

may (often will) change dynamically. Such changes provide the basis for an important class of events.

The situation is a particular assignment of entities to roles completed by a set of relations between the entities. Situation may be seen as the "state" of the user with respect to his task. The predicates that make up this state space are the roles and relations determined by the context. If the relation between entities changes, or if the binding of entities to roles changes, then the situation within the context has changed. The context and the state space remains the same.

For the system's observation of the world, the situation is the assignment of observed entities to roles, and the relations between these entities. However, this idea may be extended to the system's reflexive description of its internal state. In a reflexive description of the system, the entities are the observational processes, and the relations are the connections between processes.

Thus a context can be seen as a network of situations defined in a common state space. A change in the relation between entities, or a change in the assignment of entities to roles is represented as a change in situation. Such changes in situation constitute an important class of events that we call Situation-Events. Situation-Events are data driven. The system is able to interpret and respond to them using the context model. They do not require a change in the federation of observational processes. Situation events may be contrasted with context events that do require a change to the federation.

## 4. Properties for Observational Processes

In order to dynamically assemble and control observational processes, the system must have information about the capabilities and the current state of component processes. Such information can be provided by assuring that supervisory controllers have the reflexive capabilities of auto-regulation, auto-description and auto-criticism.

A process is auto-regulated when processing is monitored and controlled so as to maintain a certain quality of service. For example, processing time and precision are two important state variables for a tracking process. These two may be traded off against each other. The process controllers may be instructed to give priority to either the processing rate or precision. The choice of priority is dictated by a more abstract supervisory controller.

An auto-descriptive controller can provide a symbolic description of its capabilities and state. The description of the capabilities includes both the basic command set of the controller and a set of services that the controller may provide to a more abstract controller. Thus when applied to the system's context, our model provides a means for the dynamic composition of federations of controllers. In this view, the observational processes may be seen as entities in the system context. The current state of a process provides its observational variable. Supervisory controllers are formed into hierarchical federations according to the system context. A controller may be informed of the possible roles that it may play using a meta-language, such as XML.

An auto-critical process maintains an estimate of the confidence for its outputs. For example, the skin-blob detection process maintains a confidence factor based on the ratio of the sum of probabilities to the number of pixels in the ROI. Such a confidence factor is an important feature for the control of processing. Associating a confidence factor to all observations allows a higher-level controller to detect and adapt to changing observational circumstances. When supervisor controllers are programmed to offer "services" to higher-level controllers, it can be very useful to include an estimate of the confidence for the role. A higher-level controller can compare these responses from several processes and determine the assignment of roles to processes.

A crucial problem with this model is how to provide a mechanism for dynamically composing federations of supervisory controllers that observe the entities and relations relative to the user's context. Our approach is to propose a reflexive meta-supervisor. The meta-supervisor is designed for a specific domain. As described above, the domain is composed of a network of possible user contexts, and the associated systems contexts. The meta-supervisor maintains a model of the current user's context. This model includes information about adjacent contexts that may be attained from the current context, as well as the user and system context events that may signal such a change.

The meta-supervisor may be seen as a form of reactive expert system. For each user context, it invokes and revokes the corresponding highest-level supervisory controllers. These controllers, in turn, invoke and revoke lower level controllers, down to the level of the lowest level observational processes. Supervisory controllers may evoke competing lower-level processes, informing each process of the roles that it may play. The selection of process for a role can then be re-assigned dynamically according to the quality of service estimate that each process provides for its parent controller.

## 5. An Example: A Video Collaboration tool

As a simple example, consider a video based collaborative working environment. Two or more users are connected via high bandwidth video and audio channels. Each user is seated at a desk and equipped with a microphone, a video communications monitor and an augmented work surface. Each user's face and eyes are observed by a steerable pan-tilt-zoom camera. A second steerable camera is mounted on the video display and maintains a well-framed image of the user's face. The augmented workspace is a white surface, observed by a third video camera mounted overhead.

The entities that compose the user's context are 1) the writing surface, 2) one or more pens, 3) the other users, and 4) the other users' writing surfaces. The roles of the user's context are 1) the current focus of attention, 2) the drawing tool, and 3) the pointer. The focus of attention may be

"assigned" by the user to the drawing surface, to another user, or to another user's workspace. Relations for entities include "looking at", "pointing at", "talking to", and "drawing on". Situations include "user speaking", "user listening", "user drawing", "user pointing while speaking", and "user drawing while speaking". If the system can properly evaluate and respond to the user's situation, then other objects, such as the video display, disappear from the users focus of attention.

The system's model of context includes the users and the entities that make up their contexts. It also includes three possible views of the user: a well-centered image of the user's face, the user's workspace and an image of the user and his environment. Observable variables include the microphone signal strength, and a coarse resolution estimation of the user's face orientation. The system context includes the roles "speaker" and "listener". At each instant, one of the users is assigned the role of the "speaker". The other users are assigned the role of "listener". The system uses a test on the recent energy level of the microphones to determine the current speaker.

Each user may place his attention on the video display, or the drawing surface or "off into space". This attention is manifested by the orientation of his face, as measured by positions of his eyes relative to the center of gravity of his face (eye-gaze direction is not required). When the user focuses attention on the video display, his output image is the well-framed image of his face. When a user focuses attention on the work surface, his output image is his work-surface. When the user looks off "into space", the output image is a wide-angle view of the user's environment. All listeners receive the output image of the speaker. The speaker receives the mosaic of output images of the listeners.

This system uses a simple model of the user's context completed by the system's context to provide the users with the appropriate video display. Because the system adapts its display based on the situation of the group of users, the system, itself, fades from the user's awareness.

## 6. Conclusions

A context is a network of situations concerning a set of roles and relations. Roles are services or functions relative to a task. Roles may be "played" by one or more entities. A relation is a predicate defined over the properties of entities. A situation is a particular assignment of entities to roles completed by the values of the relations between the entities. Entities and relations are predicates defined over observable variables.

This ontology provides the basis for software architecture for the observational components of context aware systems. Observable variables are provided by reflexive observational processes whose functional core is a transformation. Observational processes are invoked and organized into hierarchical federations by reflexive supervisory controllers. A model of the user's context makes it possible for a system to provide services with little or no intervention from the user. Applying the same ontology to the system's context provides a method to dynamically compose federations of observational processes to observe the user and his context.

## Acknowledgment

This work has been partly supported by the EC project TMR TACIT (ERB-FMRX-CT-97-0133) and by the IST-FET GLOSS project (IST-2000-26070) and IST FAME project (IST-2000-28323). It has been conducted with the participation of Joelle Coutaz and Gaetan Rey.

## References

[1] Software Process Modeling and Technology, edited by A. Finkelstein, J. Kramer and B. Nuseibeh, Research Studies Press, John Wiley and Sons Inc, 1994.

[2] J. Estublier, P.Y.Cunin, N. Belkhatir, "Architectures for Process Support Ineroperability", ICSP5,Chicago, 15-17 juin, 1997.

[3] J. L. Crowley, "Integration and Control of Reactive Visual Processes", Robotics and Autonomous Systems, Vol 15, No. 1, décembre 1995.

[4] J. Rasure et S. Kubica, "The Khoros application development environment ", in Experimental Environments for computer vision and image processing, H. Christensen et J. L. Crowley, Eds, World Scientific Press, pp 1-32, 1994.

[5] M. Shaw and D. Garlan, Software Architecture: Perspectives on an Emerging Disciplines, Prentice Hall, 1996.

[6] T. Winograd, "Architecture for Context", Human Computer Interaction, Vol. 16, pp401-419.

[7] R. C. Schank and R. P. Abelson, Scripts, Plans, Goals and Understanding, Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1977.

[8] M. Minsky, "A Framework for Representing Knowledge", in: The Psychology of Computer Vision, P. Winston, Ed., McGraw Hill, New York, 1975.

[9] M. R. Quillian, "Semantic Memory", in Semantic Information Processing, Ed: M. Minsky, MIT Press, Cambridge, May, 1968.

[10] D. Bobrow: "An Overview of KRL", Cognitive Science 1(1), 1977.

[11] R. Brooks, , "A Robust Layered Control System for a Mobile Robot", IEEE Journal of Robotics and Automation, RA-2, no. 1, 1986.

[12] A. R. Hanson, and E. M. Riseman, , VISIONS: A Computer Vision System for Interpreting Scenes, in Computer Vision Systems, A.R. Hanson & E.M. Riseman, Academic Press, New York, N.Y., pp. 303-334, 1978.

[13] B. A.Draper, R. T. Collins, J. Brolio, A. R. Hansen, and E. M. Riseman, "The Schema System", International Journal of Computer Vision, Kluwer, 2(3), Jan 1989.

[14] M.A. Fischler & T.A. Strat. Recognising objects in a Natural Environment; A Contextual Vision System (CVS). DARPA Image Understanding Workshop, Morgan Kauffman, Los Angeles, CA. pp. 774-797, 1989.
[15] R. Bajcsy, Active perception, Proceedings of the IEEE , Vol. 76, No 8, pp. 996-1006, August 1988.
[16] J. Y. Aloimonos, I. Weiss, and A. Bandyopadhyay, "Active Vision", International Journal of Computer Vision, Vol. 1, No. 4, Jan. 1988.
[17] J. L. Crowley and H. I Christensen, Vision as Process, Springer Verlag, Heidelberg, 1993.
[18] B. Schilit, and M. Theimer, “Disseminating active map information to mobile hosts”, IEEE Network, Vol 8 pp 22-32, 1994.
[19] P. J. Brown, “The Stick-e document: a framework for creating context aware applications”, in Proceedings of Electronic Publishing, ’96, pp 259-272.
[20] T. Rodden, K.Cheverest, K. Davies and A. Dix, “Exploiting context in HCI design for mobile systems”, Workshop on Human Computer Interaction with Mobile Devices 1998.
[21] A. Ward, A. Jones and A. Hopper, “A new location technique for the active office”, IEEE Personal Comunications 1997. Vol 4. pp 42-47.
[22] K. Cheverest, N. Davies and K. Mitchel, “Developing a context aware electronic tourist guide: Some issues and experiences”, in Proceedings of ACM CHI ’00, pp 17-24, ACM Press, New York, 2000.
[23] J. Pascoe “Adding generic contextual capabilities to wearable computers”, in Proceedings of the 2nd International Symposium on Wearable Computers, pp 92-99, 1998.
[24] Dey, A. K. “Understanding and using context”, Personal and Ubiquitous Computing, Vol 5, No. 1, pp 4-7, 2001.
[25] Newell, A. "The Knowledge Level", Artificial Intelligence 28(2), 1982.
[26] Nilsson, N. J. Principles of Artificial Intelligence, Tioga Press, 1980.
[27] R. Korf, "Planning as Search", Artificial Intelligence, Vol 83, Sept. 1987.
[28] J. Coutaz and G. Rey, “Foundations for a Theory of Contextors”, in Computer Aided Design of User Interfaces, Springer Verlag , June 2002.
[29] D. Salber, A.K. Dey, G. Abowd. The Context Toolkit: Aiding the development of context-enabled Applications. In Proc. CHI99, ACM Publ., 1999, pp. 434-441.
[30] K. Schwerdt and J. L. Crowley, "Robust Face Tracking using Color", 4th IEEE International Conference on Automatic Face and Gesture Recognition", Grenoble, France, March 2000.
[31] M. Storring, H. J. Andersen and E. Granum, "Skin color detection under changing lighting conditions", Journal of Autonomous Systems, June 2000.
[32] J. Allen, "Maintaining Knowledge about Temporal Intervals", Journal of the ACM, 26 (11) 1983.
[33] D. Hall, V. Colin de Verdiere and J. L. Crowley, "Object Recognition using Coloured Receptive Field", 6th European Conference on Computer Vision, Springer Verlag, Dublin, June 2000.
[34] R. Kalman, "A new approach to Linear Filtering and Prediction Problems", Transactions of the ASME, Series D. J. Basic Eng., Vol 82, 1960.
[35] J. L. Crowley and Y. Demazeau, “Principles and Techniques for Sensor Data Fusion“, Signal Processing, Vol 32 Nos 1-2, p5-27, May 1993.
[36] J. L. Crowley and F. Berard, "Multi-Modal Tracking of Faces for Video Communications", IEEE Conference on Computer Vision and Pattern Recognition, CVPR '97, St. Juan, Puerto Rico, June 1997.
[37] J. L. Crowley, J. Coutaz and F. Berard, "Things that See: Machine Perception for Human Computer Interaction", Communications of the A.C.M., Vol 43, No. 3, pp 54-64, March 2000.
[38] Schilit, B, N. Adams and R. Want, “Context aware computing applications”, in First international workshop on mobile computing systems and applications, pp 85 - 90, 1994.
[39] Dey, A. K. “Understanding and using context”, Personal and Ubiquitous Computing, Vol 5, No. 1, pp 4-7, 2001.