# Context-Aware Artifacts: Two Development Approaches

*To create context-aware artifacts, developers can choose a self-supported or infrastructure-based approach. This article surveys these approaches and their implementations in various devices, also considering associated challenges and future research directions.*

Seng W. Loke
*La Trobe University*

One of the most exciting and important recent developments in context-aware computing is to make everyday appliances, devices, and objects context aware. A context-aware *artifact* uses sensors to perceive the context of humans or other artifacts and sensibly respond to it. Adding context awareness to artifacts can increase their usability and enable new user interaction and experiences.

Creating context awareness generally follows two main approaches. In *self-supported context awareness,* designers build the ability to perceive context, reason with it, and act accordingly into the device or its dedicated hardware support. In *infrastructure-supported context awareness,* designers obtain context-aware capabilities by harnessing a hardware and software infrastructure external to and associated with the device's space. Such an infrastructure might implement the context-aware behavior of specific objects and devices or act as a general context-aware infrastructure.

Developers can also combine the two approaches. They might, for example, use additional dedicated hardware that isn't directly attached to the device but is closely linked to it, along with some external infrastructure to work the sensors. As I discuss here, both approaches have benefits as well as drawbacks, and present particular challenges and opportunities. In addition to surveying the approaches and their implementation in various devices, I consider associated challenges and future research directions.

## Self-supported context-aware artifacts

There are a variety of independent, self-supported context-aware artifacts. Following are a few examples.

### Context-aware mobile phones

Several projects have created self-supported context-aware mobile phones. As the following examples show, such systems show an overlap in the sensors they use to analyze sensor readings and match them with the user and phone situations.

***SenSay.*** The SenSay device provides a comprehensive solution that goes beyond context recognition to model possible phone states and actions.[1] SenSay uses context awareness to automatically control ringer and vibration levels, send short-message-service messages to callers, suggest calls to make, and provide access to the user's electronic calendar. SenSay users wear a combination of sensors, including a voice microphone, an ambient-noise microphone, accelerometers, a temperature sensor, and a light sensor. SenSay uses a finite-state machine to track the phone's internal states, including uninterruptible, high activity, normal, and idle. Sensor readings determine both the user's situation (whether the user is walking, sitting, running, or talking with someone, for example) and the phone's situation (such

 Published by the IEEE CS and IEEE ComSoc ■ 1536-1268/06/$20.00 

as whether it's on a table or in the user's hand or pocket). Knowing the situation, SenSay can determine the appropriate internal state and thereby decide which action to take in response to events (such as incoming calls).

***TEA-II.*** The Technology for Enabling Awareness project developed TEA-II (www.teco.edu/tea/tea_hrd2.html), a self-contained hardware device that plugs into mobile phones to add context-aware capabilities. The plug-in's sensors include two light sensors, two microphones, an accelerometer, a skin-conductance sensor, and a temperature sensor. The device uses the sensors to detect phone situations—such as in hand, on table, in pocket, and outdoors—with up to 87 percent accuracy. However, it adds a delay of up to 30 seconds. The project also explores an exchange of context information between caller and callee. Unlike SenSay, TEA-II doesn't consider internal phone states and action decision making.

***MIThril.*** The MIThril Context-Aware Cell Phone Project uses sensors such as a GPS receiver, a three-axis accelerometer, infrared tagging, and microphones to determine the user's location and activity (www.media.mit.edu/wearables/mithril/phone.html).

***Sensor-box systems.*** Using a mobile phone-based sensor box, Johan Himber and his colleagues studied the use of time-series segmentation to analyze sensor data and identify

- movement-based situations (running, walking, walking fast),
- sound pressure (loud, moderate, silent),
- illumination (total darkness, dark, normal, bright),
- touch (in hand),
- stability (unstable, stable), and
- device orientation (sideways left, sideways right, antenna up, antenna down, display up, display down).[2]

The segments can help distinguish situations such as a user sitting while the device rests on a table. It can also identify the user putting on the device and standing up, then walking simple or complex paths, such as in a corridor, down stairs, outside, into a lobby, and up stairs.

One prototype system (www.inf.vtt.fi/pdf/publications/2000/P412.pdf) uses a sensor box with sensors for acceleration, temperature, humidity, light, and conductance to implement rule-specified behaviors such as "if the phone rings and it is picked up, it should stop ringing," and "if the phone rings, and it is lifted to the ear, it should open the line." The system analyzes sensor readings to determine various situations, including whether the phone is ringing or is in hand and when it's lifted to or away from the ear.

### Other context-aware objects

In addition to communication tools, developers are now endowing everyday objects with context-aware capabilities.

***Dolls.*** The context-aware doll emits different sounds and music according to its situation and how users handle it.[3] The doll can recognize events, such as being lifted up, using a combination of 16 built-in sensors, including touch and bend sensors, a camera, a microphone, an accelerometer, and two infrared proximity sensors. A finite-state machine keeps track of the doll's internal state.

***Spoons and cups.*** The TEA project created the Mediacup (http://mediacup.teco.edu) by embedding sensor hardware into a cup. Mediacup uses temperature and motion sensors to detect the cup's situations, including the cup being stationary, drank from, played with, carried around, and filled up. It can also determine its content's current temperature. The cup's status is transmitted via infrared signals.

MIT's Chameleon Mug (www.media.mit.edu/ci/projects/chameleonmug.html) uses LCDs, bimetal strips, thermoresisters, and thermochromic ink to change color and display safety messages if the fluid in it is hot. Another MIT project, the Intelligent Spoon (www.media.mit.edu/ci/projects/intelligentspoon.html), uses embedded sensors to provide integrated information about any food that touches it. Using sensors that measure temperature, acidity, salinity, and viscosity, the spoon—which is connected to a computer via a cable—will also offer suggestions to improve the food.

***Furniture.*** Specially designed Chameleon Tables contain embedded sensors that aim to determine who is using them—where, how, and when.[4] Users can, for example, have the table's height trigger behaviors on a touch-sensitive display.

Electronic Furniture (www.equator.ac.uk/index.php/articles/632), part of the Equator project, is designing responsive furniture, including a tablecloth that can signal how long things have been left upon it. When people sit on the Context-Aware Couch (www.dsg.cs.tcd.ie/index.php?category_id=350), it can determine their identity and weight, how much they're moving, and their approximate positions. One simple associated action is to greet the person identified.

***Medication dispensers.*** A plastic pill bottle customizes messages to fit elderly users on the basis of knowledge of the drug and the user's medication schedule.[5] A reader connected to a computer reads the bottle's RFID tags. If the user hasn't lifted the bottle off the stand to take the medication, the system issues various alerts—sometimes to family or friends—depending on the situation's urgency. Its researchers continue to investigate related issues such as trust and robustness. Other researchers have developed a context-aware medication-reminding system that uses medication uptake sensors to generate alerts based on the user's location.[6]

***Cameras.*** Researchers are developing a camera that creates visual effects in photographs using context information about sound, air pollution, temperature, and smell (www.viktoria.se/fal/projects/photo/context.html). The project's goal is to enhance the user's experience of photography.

### Discussion

Each example implements various concepts and illustrates enormous possibilities. Researchers can implement the concepts in different ways depending on the sensors they choose, the context information sensed, and the actions taken in response to the perceived context. A smart couch, for example, might select music and set certain television channels depending on the person sitting on it.

It's possible to make many objects and devices context aware, ranging from bowls, chopsticks, carpets, picture frames, umbrellas, kitchen appliances, doors, chairs, cupboards, keys, shoes, TVs, radios, and household robots. It's also possible to use sensors in larger objects such as vehicles that can detect information about the driver or the external environment.

By equipping objects with sensors, we can gain otherwise unavailable information about the object's situation. Examples include

- a camera attached to wearable spectacles that can "see" what the user sees,
- a wearable accelerometer that can effectively determine the user's acceleration,
- a spoon with embedded sensors that can sense what it's being put into, and
- a cup equipped with a thermometer that can easily sense its content's temperature.

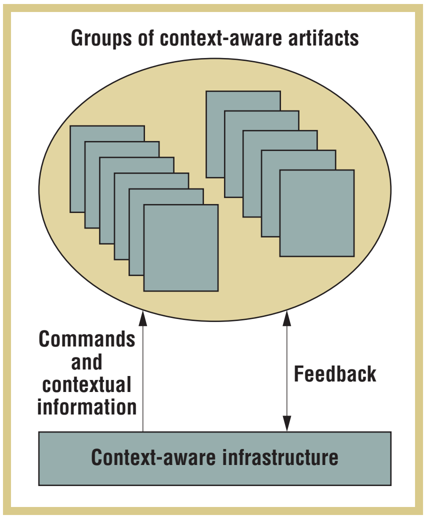



Figure 1. The relationship between context-aware artifacts and the context-awareness infrastructure. The infrastructure uses a feedback mechanism to determine the commands it issues and the context information it shares with various artifacts.

Another advantage of self-supported context awareness is reduced dependency on external infrastructure. Other artifacts that users wear or carry, such as cameras, also benefit greatly from the self-contained feature of self-supported context awareness because it requires little change in the object's appearance or operation. However, implementing self-supported context awareness poses challenges, including

- embedding sensors unobtrusively,
- reasoning with the context efficiently and accurately given limited computational resources, and
- enabling appropriate actions.

The small size of the objects and supporting devices also complicates these issues; putting sensors on a chip and projects such as TEA are steps toward "packaging" context awareness as compactly as possible.

## Infrastructure-supported context awareness

A context-aware infrastructure can support single or multiple objects. Figure 1 offers one perspective on the relationship between a context-aware infrastructure and a context-aware artifact collection. The infrastructure first receives the contextual information and, if appropriate, reasons with it. It then either issues commands or feeds contextual information to the application or artifact for appropriate actions (perhaps after its own reasoning). Finally, the application might provide feedback to the infrastructure.

### Single objects: Mobile phones

The mobile phone is a common infrastructure-supported context-aware device. The SituAwarePhone uses an ontology-based supporting infrastructure to recognize and reason with context.[7] Another context-aware phone views infrastructure as the space in which the phone is situated.[8] It then considers how the space and phone would negotiate to determine appropriate phone actions, such as setting the phone to quiet mode during a lecture.

### Artifact collections

Many interesting projects use context-aware artifact collections supported by dedicated hardware, including

- a clothes hanger system, in which weather-aware hangers light up according to the day's forecast;[9]
- the extrovert-gadgets project (www.extrovert-gadgets.net), in which e-gadgets use "synapses" to sense, act, and link up; and
- a project linking Mediacups with doorplate sensors that infer when meetings are underway.[10]

Such smart-object collaborations can have practical, useful applications. For example, a weather-aware umbrella that you've left behind might call to you if rain is forecasted and your shoe sensors indicate that you're about to walk out the front door.

It's possible to achieve peer-to-peer interaction via an underlying communication network such as Bluetooth. However, an infrastructure is more useful for facilitating interactions and function because it acts as a mechanism for detecting, combining, and reasoning with different objects' disparate context information. Such an infrastructure might consist of middleware running on a dedicated computer and connected wirelessly to various artifacts. In any case, as Marc Langheinrich and his colleagues put it,

> *An infrastructure for smart things should not only consist of an architecture to represent objects and events, but also provide various services. Smart things (or their virtual proxies) may need location information, they want to discover services in their physical proximity, and they may want to communicate to other (possibly remote) physical objects.*[11]

For example, in the RFID Chef prototype,[11] a reader detects RFID-tagged grocery items on a kitchen counter, and a computer connected to the reader then displays a list of recipes using these items as ingredients. If you add or remove items, an event-based infrastructure updates the recipe list.

In another project, Elena Vildjiounaite and her colleagues augment objects with the Smart-Its sensor module.[12] The project infrastructure combines the various objects' context information to determine their relation to each other and their collective context—often in relation to a user-specified task. Information collected about objects includes physical conditions, such as movement; sounds that the object can hear; light, temperature, and pressure; and internal states, such as battery power. Sensors collect other information according to the object type, such as use-by dates for food, and color and washing directions for clothes. Application scenarios for this technology include

- travel—querying items for their locations to determine whether they've been packed,
- cooking—checking if all the ingredients needed for a recipe are available at home, and
- organization—checking to see if you have everything you need when you're walking out the door.

### Benefits of infrastructure support

Jason Hong and James Landay[13] defined infrastructure as "a well-established, pervasive, reliable, and publicly accessible set of technologies that act as a foundation for other systems." They defined a service infrastructure as middleware technologies accessible through a network. Although they address infrastructures for context-aware applications in general, the following benefits can also be applied to context-aware artifacts:

- hardware, platform, and language independence;
- ease of maintenance, in that administrators can change infrastructure-linked services, sensors, and artifacts while the system is running; and
- improved context awareness economics.

Context awareness economics involves reaping maximum effect from a sensor. An infrastructure approach supports this by letting different artifacts and applications use the sensor's data. Moreover, an infrastructure allows computational resource sharing, and it can reason once—such as inferring high-level situations from raw sensed data—and different artifacts can reuse that reasoning. A supporting infrastructure can synthesize contextual information[11,12]—that is, unlike a single artifact's sensors, it can recognize situations using a critical mass of sensory inputs. Various sensors can feed the infrastructure information about the user's context (using, say, a positioning engine), and the infrastructure can then instruct other artifacts to respond accordingly. For example, we might enhance the context-aware doll described earlier using a positioning engine and a supporting Web service, which, when invoked, can make the doll say "hello" to its owner whenever he or she enters the room.

Furthermore, because this approach decouples the infrastructure from the artifacts, users can update infrastructures and add new artifacts to an existing environment. Additionally, the infrastructure provides resources for context-aware capabilities, requiring less of the artifacts themselves. For example, the RFID Chef's grocery items need only RFID tags, rather than embedded computers. Finally, proactive devices and objects with appropriate authorization should be able to subscribe to the context awareness infrastructure to gain awareness about the environment's people or objects. And, just as the same infrastructure can support many artifacts, the same artifact might be supported by different infrastructures at different times (for example, when the artifact changes location).

### Existing context-aware infrastructures

Researchers have also developed many infrastructures that support context-aware applications but don't specifically target everyday artifacts.[14–16] For example, the ContextBroker[17] infrastructure describes situations using an ontology that allows reasoning about actions that context-aware applications might take. Researchers have used such infrastructures to infer when users are in meetings, but you can also use them more generally with artifacts. For example, the infrastructure could make a mobile phone behave in a certain way when its owner is in a meeting.

Additionally, my Monash University colleagues and I describe how developers can use a context-aware infrastructure created for mobile services to automatically push mobile code—containing a user interface to artifacts and appliances—to the user's mobile device.[18] Such an infrastructure can also invoke functionality exposed by artifacts or start workflows among devices. Assuming it adheres to policies, the infrastructure can issue commands to control the artifact, especially if external applications can control the artifacts. One example is a doll with an embedded Web server that supports Web services. The extent to which context can control artifacts depends on the exposed functionality. In any case, even when artifacts lack sensors, the infrastructure's sensors can detect their context and invoke appropriate functionality, thereby endowing the artifacts with context-aware behavior.

### Facilitating interdevice synergies

With infrastructure-supported approaches, we can investigate synergies between networked appliances and context-aware artifacts. Some existing systems let users aggregate and compose networked devices for particular tasks.[19] However, the target devices aren't sensor enriched for context awareness but rather act as service providers—such as a collection of Web services—in the UPnP (universal plug and play) style. Integrating such systems will not only enable context-aware artifact collections, but the collections can also partially drive device interactions. For example, given a colocated set of grocery items, an application might search for recipes and display them on a kitchen screen; once the user confirms the recipe, various appliances would be preset according to the cooking instructions. So, the same infrastructure enables actions on appliances and artifacts based on the contexts of appliances, artifacts, and users. Through the infrastructure, appliances and artifacts become aware of each other.

In general, context information that the system uses in reasoning can concern a particular device, appliance, or user, or a collection of such entities. In turn, an entity might be aware only of its own context or that of a collection's. Furthermore, an entity can respond individually to its perceived context or the group's, or the application might coordinate a response among various devices.

Clearly, infrastructure can support complex context-aware behavior. Artifacts located within the scope of the infrastructure's physical space can either acquire primary context-aware capabilities or enhance their existing capabilities. However, having comprehensive context information for all entities, using sophisticated reasoning, and instigating complex actions in response to perceived context aren't necessarily ideal or even required in all context-aware applications. Most people entering a room full of dolls would prefer that they not all issue greetings, for example.

Before implementing context awareness, developers should consider three questions:

- How can a system best acquire context?
- Should we choose a self-supported or infrastructure-based approach (or some combination)?
- How can the system reason with and use context for a particular application given the constraints of cost, reasoning efficiency, timeliness of action, and user intelligibility?

To achieve situation recognition, developers can choose from many methods using many different sensor combinations. The method they choose depends on factors such as available infrastructure, project costs, required accuracy, and difficulty level.

Among the outstanding challenges in the context awareness area are: How can we program such entities to respond appropriately to richer contextual information? How will users perceive automatic responses from artifacts and devices? Although, for example, it's useful to have artifacts and devices react automatically, users must retain a sense of control and clearly understand what's happening. Too many automated actions could disorient users. Moreover, systems must respond quickly, because many actions might be useless beyond a certain time period. System actions should be both straightforward (so users can understand them) and timely. Finally, users might accept unexpected, but helpful, behaviors.

The sophistication of context-aware behavior and responses depends on the sensing technology's sophistication, the employed reasoning, and the permitted actions. These aren't independent; for example, developers must modify the reasoning component when adding new sensors because they provide new kinds of sensed information. In looking ahead, we might benefit from borrowing the notion of *levels of competence* from robotics,[20] considering three levels of context-aware-artifact competence:

- *Level 1*. Reactive artifacts that simply react with simple actions to sensor readings.
- *Level 2*. Reasoning artifacts that perform reasoning with sensor readings, building a model of situations and acting only when they recognize the appropriate situation.
- *Level 3*. Proactive artifacts that not

only maintain situation models but also plan a series of actions in response, possibly acting proactively. Such artifacts resemble the notion of agents.[21]

Although much work still needs to be done, we can further catalog current and future context-aware artifacts, document design patterns and principles, and develop infrastructures to provide a body of examples, techniques, and tools to support context-aware-artifact developers. An infrastructure can help link inputs from new sensors to new kinds of responsive artifacts, enabling context-aware behaviors that are even more interesting. For example, a wall painted with smart paint might change color, and household robots could adapt their actions to our emotions and movements.[22–24]

## REFERENCES


1. D. Siewiorek et al., "SenSay: A Context-Aware Mobile Phone," *Proc. 7th Int'l Symp. Wearable Computers* (ISWC 03), IEEE CS Press, 2003, pp. 248–250.

2. J. Himber et al., "Time Series Segmentation for Context Recognition in Mobile Devices," *Proc. Int'l Conf. Data Mining* (ICDM 01), IEEE CS Press, 2001, pp. 203–210.

3. T. Yonezawa et al., "Context-Aware Sensor-Doll as a Music Expression Device," *Proc. Conf. Human Factors in Computing Systems* (CHI 01), ACM Press, 2001, pp. 307–308.

4. T. Selker, E. Arroyo, and W. Burleson, "Chameleon Tables: Using Context Information in Everyday Objects," *Proc. Conf. Human Factors in Computing Systems* (CHI 02), ACM Press, 2002, pp. 580–581.

5. A. Agarawala, S. Greenberg, and G. Ho, *The Context-Aware Pill Bottle and Medication Monitor*, tech. report 2004-752-17, Dept. of Comp. Science, Univ. of Calgary, 2004.

6. A. Mihailidis, L. Tse, and A. Rawicz, "A Context-Aware Medication Reminding System: Preliminary Design and Development," *Proc. Rehabilitation Eng. and Assistive Tech. Soc. of North America*, 2003; www.resna.org/ProfResources/Publications/Proceedings/2003/Papers/TSP/Mihailidis_TSP.php.

7. X. Wang et al., "Semantic Space: An Infrastructure for Smart Spaces," *IEEE Pervasive Computing*, vol. 3, no. 3, 2004, pp. 32–39.

8. K. Connelly and A. Khalil, "On Negotiating Automatic Device Configuration in Smart Environments," Proc. PerWare 04 Workshop, 2nd Int'l Conf. Pervasive Comp. and Comm., IEEE CS Press, 2004, pp. 213–218.

9. T. Matthews et al., "Augmenting Collections of Everyday Objects: A Case Study of Clothes Hangers as an Information Display," Proc. Pervasive 2004, LNCS 3001, Springer, 2004, pp. 340–344.

10. H.W. Gellersen, A. Schmidt, and M. Beigl, "Multi-Sensor Context-Awareness in Mobile Devices and Smart Artifacts," *Proc. Mobile Networks and Applications*, 2002; www.comp.lancs.ac.uk/~hwg/publ/monet.pdf.

11. M. Langheinrich et al., "First Steps Towards an Event-Based Infrastructure for Smart Things," Proc. Ubiquitous Comp. Workshop (PACT 00), Distributed Systems Group, 2000; www.vs.inf.ethz.ch/publ/papers/firststeps.pdf.

12. E. Vildjiounaite et al., "Context Awareness of Everyday Objects in a Household," *Proc. Ambient Intelligence, 1st European Symp.* (EUSAI 03), E.H.L. Aarts et al., eds., LNCS 2875, Springer, 2003, pp. 177–191.

13. J.I. Hong and J.A. Landay, "An Infrastructure Approach to Context-Aware Computing," Human-Computer Interaction J., vol. 16, nos. 2–3, 2001, pp. 287–303.

14. G. Biegel and V. Cahill, "A Framework for Developing Mobile, Context-Aware Applications," Proc. 2nd Int'l Conf. Pervasive Computing and Comm. (Percom), IEEE CS Press, 2004, pp. 361–365.

15. K. Henricksen and J. Indulska, "A Software Engineering Framework for Context-Aware Pervasive Computing," *Proc. 2nd Int'l Conf. Pervasive Computing and Comm.* (PERCOM), IEEE CS Press, 2004, pp. 77–86.

16. A.K. Dey, D. Salber, and G.D. Abowd, "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications," *Human-Computer Interaction J.*, special issue on context-aware computing, vol. 16, nos. 2–4, 2001, pp. 97–166.

17. H. Chen, T. Finin, and A. Joshi, "A Context Broker for Building Smart Meeting Rooms," *Proc. Knowledge Representation and Ontology for Autonomous Systems Symp.*, AAAI Press, 2004, pp. 53–60.

18. S.W. Loke, E. Syukur, and P. Stanski, "Adding Context-Aware Behaviour to Almost Anything: The Case of Context-Aware Device Ecologies," *Proc. MobiSys 2004 Workshop Context-Awareness*, ACM Special Interest Group on Mobility of Systems Users, Data, and Computing, 2004; www.sigmobile.org/mobisys/2004/context_awareness/papers/mobisys-ca.pdf.

19. R. Kumar et al., "Selecting Devices for Aggregation," *Proc. IEEE Workshop on Mobile Computing Services and Applications*, IEEE CS Press, 2003, pp. 150–159.

20. R.A. Brooks, *A Robust Layered Control System for a Mobile Robot*, A.I. Memo 864, Massachusetts Institute of Technology, Sept. 1985.

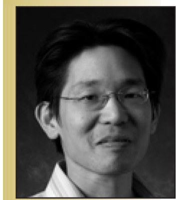
21. M. Wooldridge and N.R. Jennings, "Intelligent Agents: Theory and Practice," *Knowledge Eng. Rev.*, vol. 10, no. 2, 1995, pp. 115–152.

22. H.I. Christensen, "Intelligent Home Appliances," *Robotics Research*, R.A. Jarvis and A. Zelinsky, eds., Springer, 2003, pp. 319–330.

23. R.W. Picard and J. Klein, "Computers that Recognise and Respond to User Emotion: Theoretical and Practical Implications," *Interacting with Computers*, vol. 14, no. 2, 2002, pp. 141–169.

24. R. Headon, "Movement Awareness for a Sentient Environment," *Proc. 2nd Int'l Conf. Pervasive Computing and Comm.* (PERCOM), IEEE CS Press, 2003, pp. 99–106.


### the AUTHOR




**Seng W. Loke** is a senior lecturer in the Department of Computer Science and Computer Engineering at La Trobe University, Melbourne. He's also an honorary associate of Monash University's Center for Distributed Systems and Software Engineering, where he was previously a Faculty of Information Technology Senior Research Fellow. His research interests include context-aware pervasive computing, smart-device interactions, Web technologies, and intelligent agents. He received his PhD in computer science from the University of Melbourne, pioneering an integration of logic programming with the Web. Contact him at the Dept. of Computer Science and Computer Eng., La Trobe Univ., Bundoora, VIC 3086, Australia; s.loke@latrobe.edu.au.