

Chapter 2

Context and Its Awareness

Abstract This chapter aims to introduce the definition of context and its awareness, which are shown through the review of existing works from the early and the current state of context-aware computing. The categories, the characteristics, and the property of context are described. The context-aware architecture is discussed in this chapter. The common components of context-aware applications can be summarized into three components including perceiving component, thinking component, and acting component. Additionally, some perspectives for context definition as social context, context categorization and context awareness as personalized and rationalized awareness are emphasized at the end of this chapter.

Understanding context is crucial because it has the main role in executing the context-aware applications. Basically, it is necessary to understand the definition of context, its categories, its characteristics, its properties, and its awareness in order to obtain proper design and development. This chapter will be discussing all these issues including some perspectives relating to the context for the future requirements and applications.

2.1 Context Definition

For the last decades, there is a significant amount of prototypes, systems, and applications implementing context-aware computing concept with variety kinds of contexts. Before going into detail of how to develop context-aware applications, it is important to understand the definition of context and its evolution. According to the Cambridge Dictionary Online,¹ the context is defined as “the situation within which something exists or happens, and that can help explain it.” The word “within” simply shows something inherently influences something to happen. Therefore, the definition of context-aware computing relates directly to the definition of something having the assertion as a person, a circumstance or a computer

¹<http://dictionary.cambridge.org/>.

system. The context-aware applications are the systems that can adapt their operations or behaviors to the current contexts with or without the explicit intention of the user intervention. The context is thus important as it has the primary role in executing the application. Most of the early research works in context-aware computing aimed to identify what contexts are used in their applications. As mentioned before, the history of context-aware applications started when the Active Badge Location System was introduced (Want et al. 1992). This system could determine the current location of the users and forwarded the calls to the phone closest to the users. The user's location was detected by infrared technology. Consequently, the location was only the context executing the response from the system. At that time, the location was frequently used particularly for any location-aware application especially tour guide applications (Abowed et al. 1997; Sumi et al. 1998; Cheverst et al. 2000).

Once there was a diversity of context-aware applications, more entities were introduced as the contexts. For example, Schilit and Theimer (1994) described context as locations, identities of nearby people, objects, and changes to those objects. Ryan et al. (1999) defined the context as the user's location, environment, identity, and time. Dey (1998) described the context as the user's emotional state, location, orientation, date, and time, as well as objects and people in the environment. Some works use synonyms for context, such as environment and situation (Brown 1995; Franklin and Flaschbart 1998; Rodden et al. 1998; Hull et al. 1997; Ward et al. 1997; Abowd and Mynatt 2000). It can be seen that those works provide the definitions of context that are apparently based on the examples and synonyms. Identifying the general description for context is entirely challenging.

As claimed by Dey (1998), the context definitions at the early stage were too specific, and it could not be used to identify the other contexts in a broader sense. More general definition of context had been introduced consequently (Brown 1995; Pascoe 1998; Dey 2001; Dourish 2004; Bazire and Brézillon 2005; Zimmermann et al. 2007; Jumisko-Pyykkö and Vainio 2012; Alshaikh and Boughton 2013; Perera et al. 2014). For example, Brown (1995) defined the context as the elements of the user's environment which the computer knows about it. Pascoe (1998) introduced the context as a subjective concept that is defined by the entity that perceives it. It could be described as the subset of physical and conceptual states of interest to a particular entity. Dey and Abowd (2000) and Dey (2001) defined the context as "any information that can be used to characterize the situation of an entity" where "an entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." Closely related to the notion of context is the concept of the situation. The relationship between context and situation is illustrated by Dey (1998) that the situation of entity can be determined by the aggregation of context information. In this sense, the situation can be viewed as the higher abstract of context. In conclusion, many works seem to agree with the definition of context for some aspects, although the consensus of general definition of context is still diverse (Baldauf et al. 2007; Alegre et al. 2016). Some common agreements are, for example, the context can be defined as any information for characterizing the

situation of any entity. The context can give meaning to something else by itself or combination with other contexts. Moreover, the context typically means the operational term rather than its inherent properties.

2.2 Context Categories and Characteristics

Not only is the conceptual definition of context diverse, but also the categories and characteristics. Category and characteristics of context are necessary for the application or service designers to discover the context of their applications and services. This section shows the variety of context categories and characteristics used for some various applications. The context characterized as location, identity, time, and activity (Schilit and Theimer 1994) is normally used for describing the situation of a particular entity. The entity can be the place, people or things. These context types not only simply answer the questions of who is doing what, when, and where, but also leads to other sources of contextual information. For example, the personal identification number can provide the other related information such as affiliation, addresses, date of birth, etc. The entity's location can determine other objects or people nearby and what activity is occurring nearby. Therefore, the context can be considered as primary context and secondary context (Perera et al. 2014) where the primary context can be used to find the secondary context of the same entity. More specifically, primary context is any information obtained from the sensors and retrieved without using existing context and any data fusion. On the other hands, secondary context is any information that is derived by the manipulation of primary context. For example, the distance between two sensors with particular data fusion is called secondary context while the sensor data from each sensor is called primary context. Moreover, the retrieved context such as friend list, email address, etc. are also known as secondary context while the name of the user is considered as primary context. For another example, Dey et al. (2001) proposed that the context can be categorized into four categories including identity, location, status, and time. Identity means that each entity has a unique identifier. Location means the entity's position. Status means the intrinsic properties of the entity such as the temperature in the room, the lightness in the car, etc. The status is also considered as the activity. Finally, time is used to define the situation accurately. These ways to characterize the context cannot cover all emerged context types due to the variety of system requirements.

There are many ways to classify context into different categories. For example, the work proposed by Chen and Kotz (2000) defined the categories of the context as computing context, user context, physical context, temporal context, and context history. The computing context includes network connectivity, communication bandwidth, and local computing resources such as printers, displays, etc. User context can be user profile, location, social situation. Physical context can be lighting and noise levels, traffic conditions, and temperature. Temporal context includes time of day, week, month, and season of the year. Context history is the

storage of existing context in different points of time. Another popular way to classify types of context is to classify it into external and internal context (Baldauf et al. 2007; Schuster et al. 2002; Prekop and Burnett 2003). At the same time, they can be called physical and logical contexts respectively (Hofer et al. 2003). The physical or external context refers to the context that can be measured by hardware sensors such as location, light, sound, temperature, etc. On the other hand, the logical or internal context is something specified by the user such as user's goal, task, etc. Henricksen (2003) proposed that the context can be categorized as sensed, static, profiles, and derived context categories. Sensed context is sensor data directly detected by the sensors such as temperature, humidity, speed, etc. Static context is the information that does not change over time such as the identification of sensors from the manufacturer, person identification, etc. Profile context means the information that can evolve over time with low frequency such as the location of the sensor, the status of the person, etc. Finally, the derived context means the information that is computed by using primary context such as the distance between two sensors. Other popular types of context are the operational and the conceptual context (Van Bunningen et al. 2005; Alegre et al. 2016). The operational context means context relating to system's operation and it involves directly to the context acquisition, modeling, and treating. At the same time, the conceptual context covers meaning and relationship and can explain the relationship between contexts.

Nowadays, social context has become popular because there have been the increasing demands of social-aware applications. Social context is used to be defined as the person nearby or the group to which the user belongs. Recently, new definitions of social context are proposed. For example, Liang and Cao (2015) defined the social context as "a set of information derived from direct or indirect interactions among people in both virtual and physical world." Social context plays a significant role in public security and public health as automatic crowd detection, criminal analysis, disease infection, etc. (Eubank et al. 2004). Most of the social-aware applications deal with a large of digital traces of the users. Moreover, the application itself has shifted into networked system interacting with the community rather than single user perspective system (Eubank et al. 2004).

Besides characterizing context from what context is used for the applications, the context can also be classified from acquisition ways. For this point of view, the context can be divided into 2 different types including state information and change event. For state information, the application actively requests (pull) required context and accesses to actual and historical data such as current location, device, etc. For the change event, the application registers for particular change events and waits passively for the events. Then, the context service notifies registered applications about changes of state (push) such as the location changes, network changes, etc. For example, the air conditioner is set to turn on if the temperature is more than 25 °C means that the system uses the stat status to turn on the air conditioner. At the same time, the air conditioner is set to turn on if the temperature increases means that the system uses the change event status to turn on the air conditioner.

2.3 Context Property

Context property is essential for system design and development for context-aware applications. Different properties require different detail for designing and development. Context attributes can have the variety of properties. For example, time-dependent context which represents dynamic information that the values change over time. At the same time, static information like date-of-birth can be interpreted as information with change frequency of zero. Historic context is the context representing values at different points in time. Incorrect context is the context that can be incorrect due to inaccurate sensor information, measurement failure, wrong assumptions for derivation and interpretation. The quality of context depends on uncertainty in measurements and many evitable reasons. Multiple-resource context means that the same information can be gathered in different ways such as the location of a person can be collected from GPS, the position of the device, etc. Multidimensional/Heterogeneous context means that the context can be physical or technical context and private or social context at the same time. Distributed context means that the context occurs everywhere and all the time. Insecure context means that some contexts require security and privacy protections. Imperfect context means that the context always is imperfect especially regarding incomplete and inconsistent. Unforeseeable context means that the unforeseen context always occurs in the real life system.

2.4 Context Awareness

Context awareness represents the ability of the system that can use the context to provide the appropriate response to the users. Many systems can be considered as the context awareness systems, but they are called by other names such as smart system, intelligent system, adaptive system, etc. Since there is the variety of systems that can be considered as the context-aware system and the diversity of the context definition, it is also still challenging to make the consensus of the definition of context awareness nowadays. The term context awareness was firstly called sentient (Schilit and Theimer 1994) and later defined by Dey (2001) as “A System is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task.” Since the term context awareness highly depends on the definition of context, the context awareness can be diverse. Instead of paying much effort into clarifying definitions, many works illustrates the meaning by implementing the system with different levels of awareness.

The context awareness can be identified into 3 levels based on the user interaction (Barkhuus and Dey 2003) including personalization, passive context-awareness, and active context-awareness. Personalization means that the system allows the users to set their preferences to the system manually. For example, the

users can set their preferred coffee taste where the coffee machine can maintain the preferred taste for the users. Passive context-awareness means that the system continuously monitors the environment and proposes the appropriate choices to the users for taking the actions. For example, the promotion or discount messages are sent to the user's browser when they are surfing the Internet and wait for the action from the users. Finally, active context-awareness means that the system continuously and autonomously monitors the situation and acts independently without the intervention from the user. For example, the smart home can watch the intruders and will autonomously notify the owners or the police when someone, not the owner, has broken into the house.

Once the context-aware applications have to deal with more complexity of context, the context-aware application itself requires a deeper understanding of context which involves the complex relation of entities (Alegre et al. 2016). Therefore, the context-aware system can be classified into 2 different modes based on the interaction with the system including execution and configuration modes. Execution mode refers to the system acts or behaves particularly for a particular situation. For example, when the phone receives the call during the meeting, the phone turns automatically to the silent mode. At the same time, configuration mode refers to the adjustment of action or behavior that the system will be performing in the future. For example, the phone can adjust the action accordingly to the user's preference that the phone will not turn to silent mode when receiving the calls from someone. At the same time, the context-aware system can be considered as the active or passive system. The active system means that the system changes its content automatically while the passive system means that the system will change the content when the user has explicit involvement. Alegre et al. (2016) also suggested that context-aware system does not have to be completely active or passive. They can be some degrees in between as called hybrid mode. Following Alegre et al. (2016), there are possible 4 types of context-aware systems accordingly to the interaction with and without the involvement of the users.

Active execution

The system acts automatically depending on the context and can be called as a self-adaptive system. It can adjust its behavior accordingly to the perception of the environment and itself. For example, the screen of the smart device can switch between landscape and portrait depending on its pose automatically. The air conditioner turns on automatically when the room temperature is higher than the particular degree. This system requires less or no effort from the user and no special knowledge to use the system. However, it's hard to ensure that the system will perform appropriate actions or behaviors. At the same time, the users can be uncomfortable because they do not know what their information will be used. The examples of existing systems are context-aware self-adaptive frameworks for mobile application (Cheng et al. 2009; Salehie and Tahvildari 2009; Mizouni et al. 2014), MUSIC project (Hallsteinsen et al. 2012; Rouvoy et al. 2009; Geihs and Wagner 2012), etc.

Passive execution

This system requires the involvement of the users that they have to specify how the application should change or behave in some particular situations. The system can provide all possible information or actions to the user to select the appropriate actions. This system gains the trust from the users because they understand how the system works. The system is designed to take the actions that the user wants, so it is easy to evaluate the system's behavior. However, this system requires higher context understanding and the development of explanation generation which are difficult to develop. At the same time, the system needs more information to explain actions. The examples of existing systems belonging to this mode are some tool-kits for context-aware applications (Lim and Dey 2009).

Active configuration

This system can learn from the user preferences for automatically change or evolve the rules for the future behavior. This system requires less or no effort from the user. At the same time, the user requires no special knowledge to use the system. The system is expected to understand the behavior or the habits of the users. It's hard to determine which rules should be added or deleted for particular changes. The system needs the complex module to deal with inaccurate and uncertain sensed data. The existing systems are, for instance, the context-aware adaptive systems having foreseen and unforeseen types of context (Mori 2011; Inverardi and Mori 2013), the system discovering patterns within the user actions (Aztiria et al. 2013), the system that can generate reasoning rules automatically appropriately (Ibarra et al. 2014), etc.

Passive configuration

This system requires the involvement of the users by manually providing the personalized information to the system such as preferences, likes, expectation, etc. This system offers not only greater control and ownership to the users but also the greater creativity and motivation. It can release the burden of the developer because the users can manage their task, and they know their task the best. However, the users might be forced to contribute and cooperate with something they could lack experience. At the same time, the system may have to deal with more sophisticated methods. The examples of existing systems are Trigger-action programming (Ur et al. 2014; Huang and Cakmak 2015), iCap project (Dey et al. 2006) which is a system that is the intermediate layer between low-level toolkits and users, etc.

For our perspective on context awareness, it is a combination of some degree of the system acting personally and rationally with and without the intervention from the users. System acting personally means that the system would like to act in the manner of user's preference with or without the intervention from the users. On the other hand, system acting rationally means that the system can adapt itself to be able to interact with the new environment appropriately with or without the intervention from the users. It can be seen that, both systems can be triggered with or without the interventions from the users. The detail of this perspective will be later discussed in the last section of this chapter.

2.5 Context-Aware Architecture

Context-aware systems can be implemented in many different ways under many considerations such as individual requirements, the number of users, types of user devices, context acquisition methods, etc. The system architecture is necessary required for system representation and implementation. The system architecture is an abstraction which is used for generalizing the systems without showing the detail of implementation (Alegre et al. 2016). Like any other systems, the context-aware system requires flexible system architecture. The existing works illustrate the evolution of context-aware architectures supporting from specific purpose application to the general purpose application.

Winograd (2001) introduced three different context-aware architecture including widgets, networked services, and blackboard model. The widget is a software component providing the interface for hardware sensors (Dey and Abowd 2001). By hiding the low-level detail of sensing, it is easy to develop the application and obtain reusability. The widget can increase the efficiency but may not be robust for general purpose architecture. Many widgets are controlled by the widget manager. At the same time, the networked services can be considered as more flexible approach. Instead of having widget manager, the networked services can be found by using particular discovery techniques. Finally, the blackboard model represents the data-centric view. The blackboard is a shared media for notifying when some specified event happen. The idea came from many experts sitting around a blackboard and cooperating to solve a particular problem together (Taylor et al. 2009). This architecture is easy to implement, but there is the need of the centralized server to host the blackboard.

There have been other different points of view for proposing system architecture for context-aware applications. For example, three different architectures can be classified based on context acquisition approaches (Chen 2004) including direct sensor access, middleware infrastructure, and context server. For direct sensor access, this approach is used for the devices that have built-in sensors. The software gathers information directly from the sensors. This method can be considered as a tightly coupled approach that may not be suitable for distributed systems. For middleware architecture, it is frequently used by modern software design using encapsulation to separate functionality. The middleware approach introduces a layered architecture for hiding sensing detail at low-level. This approach promotes reusability of hardware sensors and extensibility of the system. For context server, this approach introduces remote management component to the middleware-based architecture. The server is mainly responsible for gathering sensor data by facilitating concurrent multiple accessing. The advantage of this approach is to promote the reusability of sensors and the decrement of resource intensive operation.

Three different architectures can also be classified based on actions required by the context-aware applications (Hu et al. 2008) including acquisition, representation, delivery and reaction of the contexts. The architectures thus include no application-level context model, implicit context model, and explicit context model

respectively. No application-level context model means that the applications perform all actions within the application boundaries. Implicit context model means that the applications use some other resources to carry out all actions such as libraries, frameworks, toolkits, etc. Explicit context model means that the applications use a context management infrastructure or middleware solution for performing all actions outside the application boundary. Context management and application are separated for being developed and extended independently.

The layered architectures have been proposed to satisfy the needs of general architecture that can be tailored to any application appropriately. The particular characteristic of layered architectures is that the functionalities are divided into layers and the components in a meaningful manner. Each component performs a limited task independently to support the extensibility. The well-known layered architecture for the context-aware system was proposed by Baldauf et al. (2007) having five different layers including sensors, raw data retrieval, storage and management, pre-processing, and application layer.

Sensor layer deals with a collection of various sensors including physical, virtual and logical sensors (Indulska and Sutton 2003). Physical sensors for almost every physical measurement are widely available nowadays. For example, location can be sensed by using GPS, Global System for Mobile Communication (GSM) or satellite system. Light can be detected by photodiodes. Temperature can be detected by thermometers, etc. Next, virtual sensors mean the source of context data from software applications or services. For example, the location can be identified by browsing travel booking system besides using only physical sensors as location tracking system. Finally, logical sensors combine some physical and virtual sensors together with additional information to obtain higher level abstraction. For example, the location can be detected by analyzing user login and the mapping of their device locations. At the same time, the logical sensors can be considered as the fusion of physical and virtual sensors. As shown with the name, raw data retrieval layer deals with retrieval of raw context data by using appropriate drivers for physical sensors and Application Programming Interface (API) for virtual and logical sensors. Pre-processing layer is used for preparing the information ready for the applications. Storage and management layer deal with organizing the gathered data, keeping them in the appropriate space and offering them to the clients. There are two ways of data accessing from the clients including synchronous and asynchronous modes. For synchronous mode, the client sends a message to request the data until receiving the answers from the server. For asynchronous mode, the client subscribes to specific interest events which the client will be simply notified when the event happens. Finally, the application layer is responsible for implementation of the applications.

The system architecture can also be implicitly represented by considering the integrated features in any context-aware application. For example, Abowd et al. (1999) identified three features that the context-aware application could support including presentation, execution, and tagging. Presentation means that the context can decide what information or the services should be presented to the users. The general example is the advertisement message sent to the users when they are in the

department store through their smartphones (Institutes 2011). The smart refrigerator (Moses 2012) connects to the mobile and informs what the users should bring into home from the department store. These examples illustrate the idea of selecting suitable methods for representing the appropriate information to the users. Next, the execution means that the actions are taken automatically based on the context. For example, the air-conditioner is turned on when the users start to drive home from somewhere else. Tagging means that the collected sensor data needs to be analyzed, fused, and interpreted so that it can be processed and understood later. Context tagging can also be called context annotation.

Although many works propose different architectures for the context-aware application, those designs frequently share the common ideas. Some similar ideas are, for example, promoting many extraordinary abilities such as the heterogeneity of sensor sources, scalability of the sensors, tractability for controlling and debugging, providing tolerance for the component failure, supporting privacy, promoting mobility of users and applications, enabling ease of development and configuration, etc.

2.6 Common Components

So far, it can be clearly seen that the context-aware applications can respond to any stimuli like other living things or artifacts (Loke 2006) such as the robot, software agents, etc. Especially for context-aware systems, identifying, understanding, and exploiting the context of entities are the primary tasks. Accordingly to Loke (2006), this section describes the standard components among context-aware applications including perceiving or sensing component, thinking component, and acting component.

2.6.1 *Perceiving Component*

Perceiving or sensing is the acquisition of data or information about the physical world which is used by a computer system to determine appropriate actions. It can be both biological and non-biological sensors. Multiple sensors may give a more comprehensive view of the physical world but perhaps more complexity of data manipulation. Information can be sensed through many different sensors such as light sensor, temperature sensors, motion sensors, touch sensors, etc. There are some concerns about sensors. For example, the sensors could be embedded in the environment, and sometimes they can be worn unobtrusively. It is challenging to determine the best reasoning and combination methods for acquiring context information for particular context-aware applications.

2.6.2 *Thinking Component*

The critical component of the context-aware application is to make use of gathered information from the sensors to perform the appropriate response to the users and/or the environment. The thinking component aims to make sense from all collected information. There are two schools of thought widely accepted including rationalists and empiricists. Rationalists use only reasoning to gain knowledge. On the other hand, empiricists used experience through the senses and stored in the memory to acquire knowledge. The combination of two methods is also accepted in which some information is perceived via the sensors and employs reasoning to infer more knowledge. Context-aware applications acquire sensor information and then reason it with other knowledge so that the further knowledge can be assumed. There have been many ways of making sense from information, for example, using mathematical models, using feature-based inference techniques such as pattern recognition, and neural networks, and using cognitive-based models as knowledge bases, and fuzzy logic.

2.6.3 *Acting Component*

Once context information has been gathered, the context is analyzed, and the situations are recognized, the appropriate actions are expected to be taken. Not only is the performance the primary consideration, but also the control action such as override actions, cancel actions or stop actions. The context-aware applications need the appropriate design of the responses of those control actions. Moreover, there have also been varieties of actuators that the application needs to interact with their environment and users. The most concern for acting component is to select the appropriate actuators to satisfy the requirements of the users and the environment which are typically domain-specific applications.

2.7 Common Architecture

It can be seen that all proposed context aware architectures attempt to represent the architecture with different perspectives. However, three primary components can be identified including perceiving, thinking, and acting components respectively. For example, Fig. 2.1 shows that the layered architecture proposed by the works of Baldauf et al. (2007) and Abowd et al. (1999) share three common components.

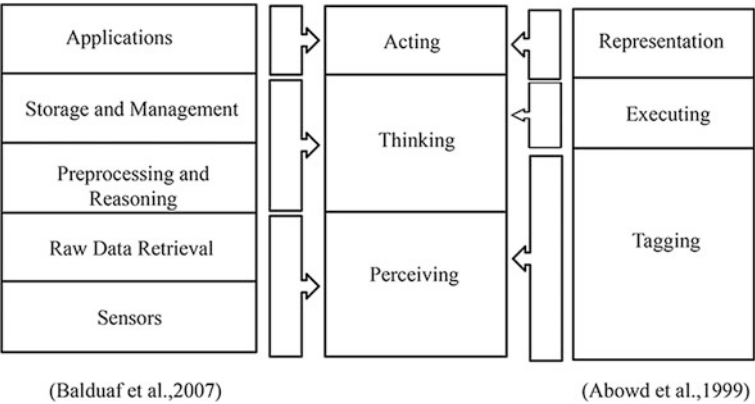


Fig. 2.1 Common components of context-aware architectures

2.8 Perspectives of Context

This section shows the perspectives for context definition, context categorization, and context awareness of this book.

2.8.1 Definition Perspective

As it can be seen from the last section, it is quite clear from the literature that it is not easy to define the definition of context because of variety kinds of context-aware applications. At the early stage, the contexts are easily defined accordingly to what contexts are used in particular applications such as location, time, environment, identity, etc. All among those contexts, location is the frequently used at the early stage. Then more contexts are introduced later such as emotional state, orientation, social context, etc. Later, there has been the effort to define the conceptual definition for context. Most of the research works do agree that the conceptual definition of context is still complicated to be built even nowadays. Figure 2.2 shows the evolution of context definition that the conceptual definition is still going on and new contexts are on the way to be announced.

2.8.2 Categorization Perspective

As the different perspective on context, it can be concluded that there is no single categorization can accommodate all types of context nowadays. For our perspective on the category of the context, the context can be classified as the individual and

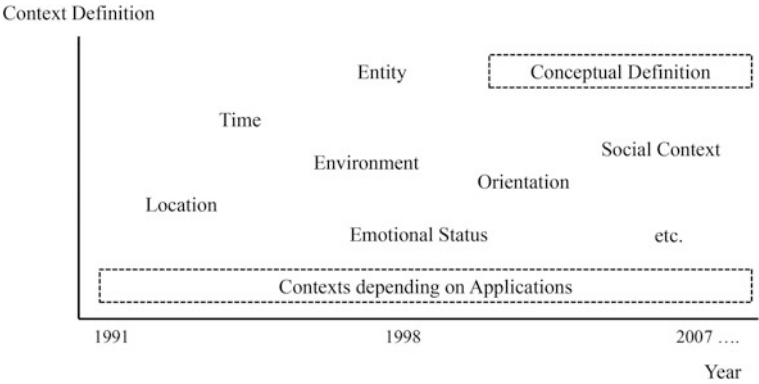
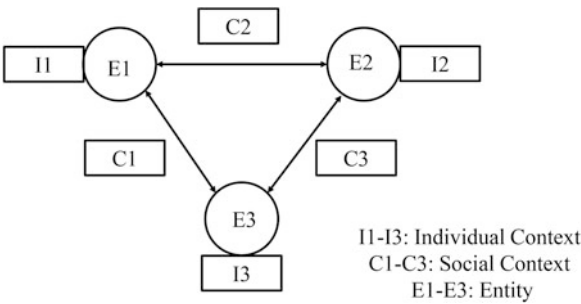


Fig. 2.2 Evolution of context definition

the social context. The individual context is the context that provides the meaning without the interaction with other entities while the social context is the context that needs interaction among entities to provide the meaningful information. The interactions can be the peer or group interactions. Absolutely, the entity can be anything not only people (Liang and Cao 2015) such as sensors, software agents, insects, robots, even the machines as on Internet of Thing (IoT) paradigm. Moreover, each type of context can have its attributes which can be tailored for any applications individually. Figure 2.3 shows the conceptual diagram of individual and social context based on the proposed perspective.

From Fig. 2.3, the interaction between entities does not simply mean the fusion or the aggregation between or among the contexts. For example, to determine the group information which is the social context, it is not just aggregating individual contexts (I1,I2,I3) together. On the other hand, the social context (C1,C2,C3) can be identified by the manipulations of all interactions among all entities (E1,E2,E3) instead. The future context-aware applications requiring the social context with this perspective will be discussed again in Chap. 6.

Fig. 2.3 Context categorization based on context interaction



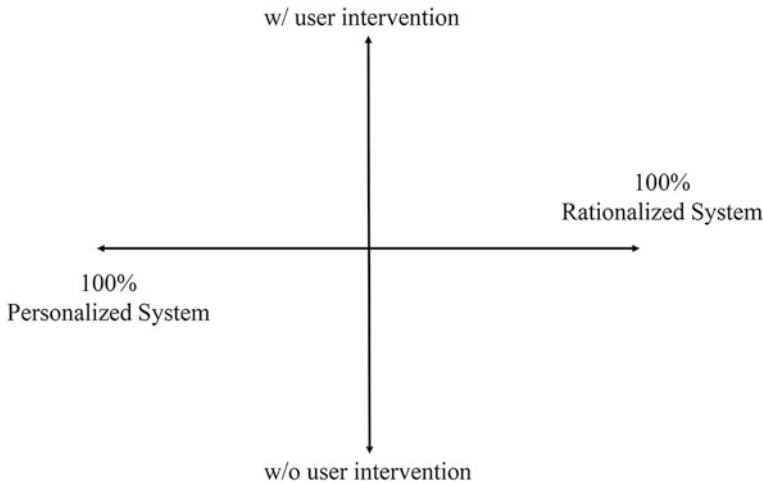


Fig. 2.4 Personalized and rationalized context-aware application

2.8.3 Awareness Perspective

As mentioned before, the context awareness can be considered as the systems acting personally or rationally with and without the intervention from the users. This book would like to identify the awareness model as in between the personal based reflex model and rational based reflex model as shown in Fig. 2.4.

It can be seen from Fig. 2.4 that any system can have the combination of personalized and rationalized systems. Both systems can be executed with or without the intervention from the users. The personalized system means that the system can act as the user's manner no matter what the users have trained the system consciously. On the other hand, the rationalized system means that the system can act appropriately to the situation or environment no matter what the users have trained the system consciously.

References

- Abowd, G. D., Atkeson, C. G., Hong, J., Long, S., Kooper, R., & Pinkerton, M. (1997). Cyberguide: A mobile context-aware tour guide. *Wireless Networks*, 3(5), 421–433.
- Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M., & Steggles, P. (1999, January). Towards a better understanding of context and context-awareness. In *Handheld and ubiquitous computing* (pp. 304–307). Berlin, Heidelberg: Springer.
- Abowd, G. D., & Mynatt, E. D. (2000). Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer—Human Interaction (TOCHI)*, 7(1), 29–58.
- Alegre, U., Augusto, J. C., & Clark, T. (2016). Engineering context-aware systems and applications: A survey. *Journal of Systems and Software*, 117, 55–83.

- Alshaikh, Z., & Boughton, C. (2013, October). Notes on synthesis of context between engineering and social science. In *International and Interdisciplinary Conference on Modeling and Using Context* (pp. 157–170). Berlin, Heidelberg: Springer.
- Aztiria, A., Augusto, J. C., Basagoiti, R., Izaguirre, A., & Cook, D. J. (2013). Learning frequent behaviors of the users in intelligent environments. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 43(6), 1265–1278.
- Baldauf, M., Dustdar, S., & Rosenberg, F. (2007). A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4), 263–277.
- Barkhuus, L., & Dey, A. (2003, October). Is context-aware computing taking control away from the user? Three levels of interactivity examined. In *International Conference on Ubiquitous Computing* (pp. 149–156). Berlin, Heidelberg: Springer.
- Bazire, M., & Brézillon, P. (2005, July). Understanding context before using it. In *International and Interdisciplinary Conference on Modeling and Using Context* (pp. 29–40). Berlin, Heidelberg: Springer.
- Brown, P. J. (1995). The stick-e document: A framework for creating context-aware applications. *Electronic Publishing-Chichester-*, 8, 259–272.
- Chen, G., & Kotz, D. (2000). A survey of context-aware mobile computing research (Vol. 1, No. 2.1, pp. 2-1). Technical Report TR2000-381, Department of Computer Science, Dartmouth College.
- Chen, H. (2004). *An intelligent broker architecture for pervasive context-aware systems*. Baltimore County: University of Maryland.
- Cheng, B. H., de Lemos, R., Garlan, D., Giese, H., Litoiu, M., Magee, J., & Taylor, R. (2009, May). Seams 2009: Software engineering for adaptive and self-managing systems. In *Proceedings of the 2009 31st International Conference on Software Engineering: Companion Volume* (pp. 463–464). IEEE Computer Society.
- Cheverst, K., Davies, N., Mitchell, K., Friday, A., & Efstratiou, C. (2000, April). Developing a context-aware electronic tourist guide: Some issues and experiences. In *Proceedings of the ACM, SIGCHI conference on Human Factors in Computing Systems* (pp. 17–24).
- Dey, A.K. (1998, March). Context-aware computing: The CyberDesk project. In *Proceedings of the AAAI 1998 Spring Symposium on Intelligent Environments* (pp. 51–54).
- Dey, A. K., & Abowd, G. D. (2000). Towards a better understanding of context and context-awareness. In *Workshop on the What, Who, Where, When, and How of Context Awareness, affiliated with the 2000 ACM Conference on Human Factors in Computer systems*.
- Dey, A. K. (2001). Understanding and using context. *Personal and Ubiquitous Computing*, 5(1), 4–7.
- Dey, A. K., Abowd, G. D., & Salber, D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, 16(2), 97–166.
- Dey, A.K., Sohn, T., Streng, S., & Kodama, J. (2006, May). iCAP: Interactive prototyping of context-aware applications. In *International Conference on Pervasive Computing* (pp. 254–271). Berlin, Heidelberg: Springer.
- Dourish, P. (2004). What we talk about when we talk about context. *Personal and Ubiquitous Computing*, 8(1), 19–30.
- Eubank, S., Guclu, H., Kumar, V. A., Marathe, M. V., Srinivasan, A., Toroczkai, Z., et al. (2004). Modelling disease outbreaks in realistic urban social networks. *Nature*, 429(6988), 180–184.
- Franklin, D., & Flaschbart, J. (1998, March). All gadget and no representation makes jack a dull environment. In *Proceedings of the AAAI 1998 Spring Symposium on Intelligent Environments* (pp. 155–160).
- Geihs, K., & Wagner, M. (2012, November). Context-awareness for self-adaptive applications in ubiquitous computing environments. In *International Conference on Context-Aware Systems and Applications* (pp. 108–120). Berlin, Heidelberg: Springer.
- Hallsteinsen, S., Geihs, K., Paspallis, N., Eliassen, F., Horn, G., Lorenzo, J., et al. (2012). A development framework and methodology for self-adapting applications in ubiquitous computing environments. *Journal of Systems and Software*, 85(12), 2840–2859.

- Henricksen, K. (2003). *A framework for context-aware pervasive computing applications*. Queensland: University of Queensland.
- Hofer, T., Schwinger, W., Pichler, M., Leonhartsberger, G., Altmann, J., & Retschitzegger, W. (2003, January). Context-awareness on mobile devices-the hydrogen approach. In *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on IEEE* (p. 10-pp).
- Hu, P., Indulska, J., & Robinson, R. (2008, March). An autonomic context management system for pervasive computing. In *Pervasive Computing and Communications, 2008. PerCom 2008. Sixth Annual IEEE International Conference on IEEE* (pp. 213–223).
- Huang, J., & Cakmak, M. (2015, September). Supporting mental model accuracy in trigger-action programming. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (pp. 215–225). ACM.
- Hull, R., Neaves, P., & Bedford-Roberts, J. (1997, October). Towards situated computing. In *Wearable Computers, 1997. Digest of Papers., First International Symposium on IEEE* (pp. 146–153).
- Ibarra, U.A., Augusto, J.C., & Goenaga, A.A. (2014, June). Temporal reasoning for intuitive specification of context-awareness. In *Intelligent Environments (IE), 2014 International Conference on IEEE* (pp. 234–241).
- Indulska, J., & Sutton, P. (2003, January). Location management in pervasive systems. In *Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003-Volume 21* (pp. 143–151). Australian Computer Society, Inc.
- Institutes, C. (2011). Smart networked objects and internet of things. *Carnot Institutes' Information Communication Technologies and Micro Nano Technologies alliance*, White Paper.
- Inverardi, P., & Mori, M. (2013). A software lifecycle process to support consistent evolutions. In *Software Engineering for Self-Adaptive Systems II* (pp. 239–264). Berlin, Heidelberg: Springer.
- Jumisko-Pyykkö, S., & Vainio, T. (2012). Framing the context of use for mobile HCI. *Social and Organizational Impacts of Emerging Mobile Devices: Evaluating USE: Evaluating Use*, 217–219.
- Liang, G., & Cao, J. (2015). Social context-aware middleware: A survey. *Pervasive and Mobile Computing*, 17, 207–219.
- Lim, B.Y., & Dey, A.K. (2009, September). Assessing demand for intelligibility in context-aware applications. In *Proceedings of the 11th ACM International Conference on Ubiquitous computing* (pp. 195–204).
- Loke, S. (2006). *Context-aware pervasive systems: Architectures for a new breed of applications*. Boca Raton: CRC Press.
- Mizouni, R., Matar, M. A., Al Mahmoud, Z., Alzahmi, S., & Salah, A. (2014). A framework for context-aware self-adaptive mobile applications SPL. *Expert Systems with Applications*, 41(16), 7549–7564.
- Mori, M. (2011, September). A software lifecycle process for context-aware adaptive systems. In *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European conference on Foundations of Software Engineering* (pp. 412–415).
- Moses, A. (2012). Lg smart fridge tells you what to buy, cook and eat. *The Sydney Morning Herald*, January.
- Pascoe, J. (1998, October). Adding generic contextual capabilities to wearable computers. In *Wearable Computers, 1998. Digest of Papers. Second International Symposium on IEEE* (pp. 92–99).
- Perera, C., Zaslavsky, A., Christen, P., & Georgakopoulos, D. (2014). Context aware computing for the internet of things: A survey. *IEEE Communications Surveys & Tutorials*, 16(1), 414–454.
- Prekop, P., & Burnett, M. (2003). Activities, context and ubiquitous computing. *Computer Communications*, 26(11), 1168–1176.
- Rodden, T., Cheverst, K., Davies, K., & Dix, A. (1998, May). Exploiting context in HCI design for mobile systems. In *Workshop on human computer interaction with mobile devices* (pp. 21–22).

- Rouvoy, R., Barone, P., Ding, Y., Eliassen, F., Hallsteinsen, S., Lorenzo, J., & Scholz, U. (2009). Music: Middleware support for self-adaptation in ubiquitous and service-oriented environments. In *Software engineering for self-adaptive systems* (pp. 164–182). Berlin, Heidelberg: Springer.
- Ryan, N., Pascoe, J., & Morse, D. (1999). Enhanced reality fieldwork: The context aware archaeological assistant. *Bar International Series*, 750, 269–274.
- Salehie, M., & Tahvildari, L. (2009). Self-adaptive software: Landscape and research challenges. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 4(2), 14.
- Schilit, B. N., & Theimer, M. M. (1994). Disseminating active map information to mobile hosts. *Network, IEEE*, 8(5), 22–32.
- Schuster, S., Marhl, M., & Höfer, T. (2002). Modelling of simple and complex calcium oscillations. *European Journal of Biochemistry*, 269(5), 1333–1355.
- Sumi, Y., Etani, T., Fels, S., Simonet, N., Kobayashi, K., & Mase, K. (1998). C-map: Building a context-aware mobile assistant for exhibition tours. In *Community computing and support systems* (pp. 137–154). Berlin, Heidelberg: Springer.
- Taylor, R.N., Medvidovic, N., & Dashofy, E.M. (2009). *Software architecture: Foundations, theory, and practice*. New York: Wiley.
- Ur, B., McManus, E., Pak Yong Ho, M., & Littman, M. L. (2014, April). Practical trigger-action programming in the smart home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 803–812). ACM.
- Van Bunningen, A. H., Feng, L., & Apers, P. M. (2005, April). Context for ubiquitous data management. In *Ubiquitous Data Management, 2005. UDM 2005. International Workshop on IEEE* (pp. 17–24).
- Want, R., Hopper, A., Falcao, V., & Gibbons, J. (1992). The active badge location system. *ACM Transactions on Information Systems (TOIS)*, 10(1), 91–102.
- Ward, A., Jones, A., & Hopper, A. (1997). A new location technique for the active office. *Personal Communications, IEEE*, 4(5), 42–47.
- Winograd, T. (2001). Architectures for context. *Human-Computer Interaction*, 16(2), 401–419.
- Zimmermann, A., Lorenz, A., & Oppermann, R. (2007, August). An operational definition of context. In *International and Interdisciplinary Conference on Modeling and Using Context* (pp. 558–571). Berlin, Heidelberg: Springer.

Context-Aware Communication and Computing:

Applications for Smart Environment

Temdee, P.; Prasad, R.

2018, XI, 151 p. 47 illus., Hardcover

ISBN: 978-3-319-59034-9