3rd International Workshop on Pervasive and Context-Aware Middleware

Hamed Vahdat-Nejad

Pervasive Computing lab, Department of Computer Engineering, University of Birjand,
Birjand, Iran
Vahdat.nejad@gmail.com

1 Introduction

Generally, middleware is a software layer, which by residing between the operating system and application layer in each node, provides new capabilities and facilitates development of applications [1]. Obviously, pervasive computing middleware inherits general tasks of traditional middleware in distributed systems such as coordination, communication, security and tolerance for component failures and disconnections; however, there are some factors separating context-aware middleware from traditional middleware systems [2], as outlined below:

- The infrastructure of pervasive computing is different from that of distributed systems. Besides, heterogeneity is much more here from the standpoint of hardware and devices, software and operating systems, and communication networks. In pervasive computing, a diverse range of hardware devices and software tools has been employed to realize everywhere/ every-time computation vision.
- The notion of context introduces a major distinction between context-aware middleware systems and their predecessors in distributed systems. A context-aware middleware is responsible for context acquisition from context sources, context processing, and context dissemination to the interested applications.

Research on context-aware middleware has been performed under different titles such as architecture, framework, or middleware for context-aware or pervasive systems, infrastructure or platform for context-aware applications, and even context toolkit [3] or engine [4]. Moreover, they originate from different problems and have different capabilities.

Before designing context-aware middleware the characteristics of the environment should be specified. In the ideal situation, a context-aware middleware system, should consider multiple domains [5] such as user personal, mobile, social, urban, and ad hoc environments and also domains that are physically limited to a geographical zone like smart places [e.g. smart room, home, office, university, etc). Each of these domains can support a limited kind of context-aware applications; hence the middleware solution that has been developed for just one specific single domain is very limited in supporting general context-aware applications. On the other hand, these domains are different in nature, so it seems impossible to design generic context-aware middleware for handling all the domains in an identical way.

J.C. Augusto et al. (Eds.): AmI 2013, LNCS 8309, pp. 311–312, 2013.

2 Challenges

In the multiple-domain environments, handling mobility of entities is a challenging issue. Entities such as people, mobile phones, laptops, etc are continuously roaming between different domains such as home, office, university, etc. These entities are usually sources of contextual data, and sometimes hosts for residing context-aware applications (e.g. mobile phone). In either case, they impose new tracing and communication challenges to the system (especially in distributed architecture).

Considering the fact that mobile entities roam between different domains and their context is produced in different domains imposes that these entities should be recognized by a unique name all over the environment. Most of multiple-domain studies ignore this problem by assuming that all the entities already have a unique name, which is known all over the system.

Most of the devices available in the environment are mobile phones and PDAs with limited memory, computational, communication and availability capacities. They cannot play the role of high performance computers and also may sometimes be inaccessible or even off. Therefore, any plan for using them as infrastructure of the middleware should consider these limitations.

Handling a dynamic environment is a traditional problem in the distributed systems. However, a pervasive computing environment is highly dynamic in the meaning that devices join and leave frequently and context changes. Some of the devices such as mobile phones may be regularly switched off and on. Similarly, some of the context types such as a vehicle's location, temperature of a room, activity of a person are changing continuously and rapidly.

References

1. Edwards, W.K., et al.: Stuck in the Middle: Bridging the Gap Between Design, Evaluation, and Middleware. Intel research Berkeley. Intel Corporation, California, IRB-TR-02-013 (2002)
2. Vahdat-Nejad, H., Zamanifar, K., Nematbakhsh, N.: Context-Aware Middleware Architecture for Smart Home Environment. International Journal of Smart Home 7(1) (2013)
3. Dey, A.K., Abowd, G.D., Salber, D.: A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Human-Computer Interaction 16(2), 97–166 (2001)
4. Carrizo, C., et al.: Design of a context aware computing engine. In: Proceedings of IET 4th International Conference on Intelligent Environments, pp. 1–4. IEEE Computer Society, Seattle (2008)
5. Guo, B., Sun, L., Zhang, D.: The Architecture Design of a Cross-Domain Context Management System. In: 8th IEEE International Conference on Pervasive Computing and Communications Workshop, pp. 499–504. IEEE Computer Society, Mannheim (2010)