

HUG CorA

How to Use a Generic Coordination Architecture in Pervasive System Development

Manfred Bortenschlager

Salzburg Research
Jakob Haringer Straße 5/III, 5020 Salzburg, Austria
manfred.bortenschlager@salzburgresearch.at

Abstract. Recent technological advances in hardware manufacturing led to an increase in heterogeneity and quantity of entities involved in pervasive computing environments. Consequently, more coordination efforts have to be employed in order to keep on improving the level of service quality. Current solutions do not explicitly address the coordination issue and usually cover this by self-contained attempts which lack reusability. To address this drawback, this work proposes a generic coordination architecture (CorA) based on a three-layered structure, which shall facilitate a more effective pervasive system development by providing problem-specific and reusable system components.

1 Introduction

The rising pervasiveness, the users' raised desire for mobility, technological advances in hardware manufacturing and the claim of pervasive computing systems to support human beings in their daily routines as unobtrusively as possible by "weaving themselves into the fabric of everyday life until they are indistinguishable from it" [14] result in a tremendous increase of heterogeneous entities involved in such mobile environments. Consequently, the complexity of such pervasive systems is steadily increasing, too. In order to guarantee a certain level of service quality and usability, some form of coordination of this magnitude of entities is required. So far, however, current approaches usually do not explicitly address the coordination issue [10] which is left as an implicit "effect" of the business logic and workflow-based realisation. As a consequence, these approaches inhere significant drawbacks when it comes to modularity, reusability, exchangeability and extensibility of the implemented coordination mechanisms [2]. Hence, this work proposes a generic coordination architecture based on a theoretical coordination model which particularly aims at the specific characteristics [12] of pervasive computing environments. As a consequence, the resulting coordination framework shall facilitate a more effective pervasive system development by providing problem-specific and reusable system components.

2 The Coordination Model

Coordination connotes an abstract concept where people usually have an intuitive sense what it means. This concept as a subject is inter- and multidisciplinary and is not re-

stricted to computer-based systems but is also of great significance to other fields of research like social sciences, organisational theory, anthropology, biology, sociology and many more [11].

Coordination theory [9] suggests that standardised *coordination mechanisms* can be applied to specific coordination problems. For this, Ciancarini describes a generic coordination model [1] as a triple of $\{E, M, L\}$. In this model, $\{E\}$ represents the coordinable—either physical or logical—entities which have to be coordinated. These can be data (structures), software processes, services, agents, or even human beings interacting with computer-based systems. $\{M\}$ stands for the coordination media (i.e. communication channels), which serve as connectors between the entities and facilitate communication, which is a mandatory prerequisite for direct coordination [6, 13]. Instances of coordination media may be a message-passing environment, pipes, tuple spaces etc. $\{L\}$ is referred to the coordination laws defining how the interdependences have to be resolved and hence, semantically define the coordination mechanisms. The proposed coordination architecture presented in this work embodies this model and is outlined in the following section.

3 Proposed Coordination Architecture CorA

The pervasive middleware representing the architectural glue between sensors and actuators [5] is responsible for the correct triggering of the actuators which interact with the real world. Coordination is essential to beneficially resolve the interdependences between the occurring activities in a pervasive environment. The concept of separating the programming issues from the coordination issues seems to be promising and was also identified in literature [1, 5, 8]. By this, coordination concerns shall be made explicit in the system development of pervasive applications. The goal of this work is to provide a standard coordination framework comprising reusable components which address these concerns and which can be applied to pervasive system development. Figure 1 depicts this framework and illustrates the layered architecture. Each layer represents one tuple of the coordination model and two APIs represent the interface between the layers. The remainder of this section deals with each layer in more details.

Layer 1: Coordination Media $\{M\}$

In order to coordinate $\{E\}$, $\{M\}$ has to provide appropriate means to address the increasingly diverse communication facets. Mamei and Zambonelli [10] identified three basic communication types relevant in this context: (i) direct communication (i.e. message passing), (ii) shared data spaces (i.e. space-based computing or tuple spaces), and (iii) event-based models.

In order to appropriately address the communication requirements of pervasive computing environments [12], the $\{M\}$ of this implementation of CorA fully complies to the P2P paradigm. Hence, this layer has been modelled according to the decentralised space-based computing approach (SBC) [3] which is very similar to Linda-like systems [8]. The advantage of SBC is that it deploys a commonly shared space for data exchange and communication. This approach offers inherent decoupling mechanism such as spatial, temporal and referential decoupling [4] leading to a great extent of flexibility.

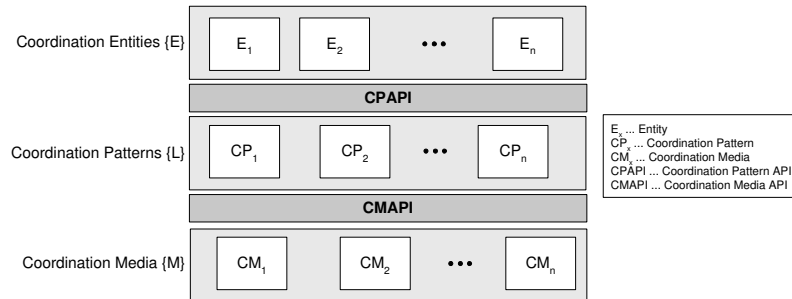


Fig. 1. Diagrammatic overview of the proposed coordination architecture

The Coordinated Shared Objects (CORSO)¹ technology was chosen to represent the concrete instance of this SBC concept for a first prototype. CORSO offers rich distributed transaction mechanisms, replication protocols, persistent data storage, automated garbage collection, support of heterogeneity and programming language independence through various language bindings. The CORSO framework has been adapted to mobile, embedded devices in order to meet the specific requirements of pervasive environments.

Layer 2: Coordination Patterns {L}

{L} defines the actual rules which are necessary to resolve the interdependencies between the entities. Coordination problems share many similarities and thus, it seems useful to identify these and provide standard solution recommendations. With respect to this layer, several coordination patterns (such as supervisor/worker, publish/subscribe, meeting, broker, blackboard, and negotiating) have been identified and described, and further ones are under investigation. Eventually, a coordination pattern catalogue similar to the idea of software design patterns [7] and several reference implementations shall be established offering predefined mechanism to resolve recurrent coordination problems.

Layer 3: Coordination Entities {E}

As mentioned earlier, {E} can comprise a great diversity of entities, which in an abstract sense can be everything which requires coordination. This is true because of the fact that coordination is a domain-independent concept [9] and can be applied to a broad range of applications. Clearly, the more sophisticated the nature of the entities are the more complex is the coordination process.

4 Implementation and Preliminary Results

Until now, a prototype implementing the supervisor/worker pattern using CORSO is under development. To simulate a pervasive environment, it was deployed on mobile devices (i.e. two Nokia 6630 and an IBM Thinkpad) which communicated via Bluetooth. First experiments and preliminary results show that the solution might fulfill the

¹ See TU Vienna, Prof. eva Kuehn: <http://www.complang.tuwien.ac.at/eva>

expectations. Further validations and evaluations are going to be conducted. The next steps would be to identify and describe more patterns and extend the prototype with reference implementations of these patterns. Moreover, further transmission technologies (e.g. WLAN, 3G, IrDA, or RFID) will be incorporated in order to provide the possibility to choose the most appropriate one according to specific requirements.

5 Conclusion and Further Work

To summarise, this paper proposes a generic coordination architecture (CorA) based on a coordination model comprising the triple $\{E, M, L\}$. CorA addresses the three essential aspects of coordination: (i) appropriate coordination media (by availing the SBC paradigm), (ii) problem-specific coordination patterns, and (iii) the entities which require coordination in order to resolve interdependences. This approach shall result in the development of a standardised catalogue of coordination patterns very similar to the idea of software design patterns. CorA provides a modular way of applying coordination patterns to specific coordination problems in pervasive environments. This approach contributes to the pervasive computing community by incorporating methods and means of the interdisciplinary study of coordination in order to improve the quality of services of pervasive computing systems and to alleviate more effective system design by deploying reusable (coordination) components.

6 Acknowledgments

The research cooperation with Prof. eua Kuehn (TU Vienna) and the provision of the CORSO framework is highly appreciated.

References

1. P. Ciancarini. Coordination Models and Languages as Software Integrators. *ACM Comput. Surv.*, 28(2):300–302, 1996.
2. D. Deugo, M. Weiss, and E. Kendall. Reusable Pattern for Agent Coordination. In A. Omicini, F. Zambonelli, M. Klusch, and R. Tolksdorf, editors, *Coordination of Internet Agents: Models, Technologies, and Applications*, pages 347–368. Springer, 2001.
3. e. Kuehn. *Virtual Shared Memory for Distributed Architecture*. Nova Science Publishers, 2001.
4. D. Fensel. Triple-based Computing. Technical Report DERI-TR-2004-05-31, DERI Innsbruck, 2004.
5. A. Ferscha. Coordination in Pervasive Computing Environments. In *Proceedings of the 12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE-2003)*. IEEE Computer Society Press, June 2003.
6. S. Franklin. Coordination without Communication, 1996. Retrieved on 10 January 2006 from <http://www.msci.memphis.edu/franklin/coord.html>.
7. E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1995.
8. D. Gelernter. Generative Communication in Linda. *ACM Trans. Program. Lang. Syst.*, 7(1):80–112, 1985.

9. T. W. Malone and K. Crowston. The Interdisciplinary Study of Coordination. *ACM Comput. Surv.*, 26(1):87–119, 1994.
10. M. Mamei and F. Zambonelli. Field-based Approaches to Adaptive Motion Coordination in Pervasive Computing Scenarios. In *Handbook of Algorithms for Mobile and Wireless Networking and Computing*. CRC Press, 2004.
11. H. S. Nwana, L. Lee, and N. R. Jennings. Coordination in Software Agent Systems. *The British Telecom Technical Journal*, 14(4):79–88, 1996.
12. M. Satyanarayanan. Pervasive Computing: Vision and Challenges. *IEEE Personal Communications*, pages 10–17, Aug. 2001.
13. H. Weigand, F. van der Poll, and A. de Moor. Coordination through Communication. In *8th International Working Conference on the Language-Action Perspective on Communication Modelling (LAP 2003)*, Tilburg, The Netherlands, 2003.
14. M. Weiser. The Computer for the 21st Century. *Scientific American*, 265(3):66–75, Sept. 1991.

