The Autonomic Computing Paradigm in Adaptive Building / Ambient Intelligence Systems

Aliaksei Andrushevich, Stephan Tomek, and Alexander Klapproth

CEESAR-iHomeLab, Lucerne University of Applied Sciences and Arts, Technikumstr. 21, 6048 Horw, Switzerland
{aliaksei.andrushevich,stephan.tomek, alexander.klapproth}@hslu.ch

Abstract. This work is devoted to the classification and adaptation of current ambient intelligence (AmI) research activities from the viewpoint of the autonomic computing paradigm. Special attention is given to the implementation of AmI's user-centric focus in autonomic computing.

Keywords: Ambient Intelligence, Autonomic Computing, self-adaptive system, user-centric requirements, human-centered design.

1 Introduction and Motivation

AmI systems are typically based on the adaptation of artificial intelligence (AI) techniques on low-power, low-cost, heterogeneous and physically distributed pervasive / ubiquitous computing infrastructure. The main aim of AmI systems is however in supporting the end-user's daily activities in an unobtrusive and easy way through user-centric architecture and human-centered design.

Building Intelligence (BI) is a type of AmI implemented on underlying technical architecture and infrastructure of buildings, homes and other construction objects. Considering constantly changing user-needs different building automation applications may have partially contradictory control strategies for commonly shared resources. Moreover, many building automation applications pursuing energy efficiency, comfort, user safety and security or ambient assistance are only widely accepted as a part of an AmI system if the configuration and maintenance efforts will be kept to a minimum. Considering constantly changing user requirements during the whole lifecycle of AmI applications, self-adaptive system properties quickly become inevitable for AmI adoption.

In order to make the system self-adaptive IBM introduced in 2001 the paradigm of autonomic computing (AC) [1] that systemizes and formalizes the necessary properties and functional components of *self-managing* system architecture. An extensive amount of focused research [2] has resulted in usage of AC paradigm in different applications areas including power management in wireless sensor networks (WSN), dynamic resource management and administration in GRID computing systems, and

D. Keyson et al. (Eds.): AmI 2011, LNCS 7040, pp. 98–104, 2011.

pervasive/ubiquitous computing vision aiming at building intelligent environments by usage of heterogeneous sensing-computing-actuating devices [3].

2 Autonomic Computing

The high-level core self-management properties of the AC paradigm include self-configuration, self-optimization, self-healing and self-protection. The intuitive sense of the autonomic system is in reflexing of the current environmental context or in reflecting dynamism in the system [2].

Generally, the implementation of AC paradigm is defined in the IBM's reference model for autonomic control loops and called MAPE-K loop - Monitoring, Analysis, Planning, Execution, and gathered Knowledge.

Five AC Adoption Model Levels were also introduced by IBM in 2003 to be able to measure the systems on their way to autonomicity. These levels include 1 - Basic, 2 - Managed, 3 - Predictive, 4 - Adaptive and 5 - Autonomic. A recent AC-focused survey [2] on self-managing systems (those of AC Levels 4 and 5) suggests to classify the ongoing AC research through the following four key autonomicity elements:

- Support – when improving the complete system performance by focusing on one aspect or component;
- Core – when end-to-end self-management solution drives the core application without heading higher-level human based goals;
- Autonomous – when full end-to-end self-management typically agent-based solution self-adapts to the environment but not measuring own performance;
- Autonomic – when full architecture is reflecting its own performance and adapting itself considering higher-level human based goals.

3 AmI Research in Components of AC Architecture

Figure 1 shows our understanding about the way different works of current AmI research can contribute to the implementation of intelligent self-managing and adaptive to the user building management system (BMS).

The state-of-the-art in context-aware ubiquitous computing middleware [4, 5] looks promising in terms of performing the data acquisition from different sensors and implementing data fusion mechanisms in the *monitoring* functional block of the AC control loop architecture.

The introduction of context, the usage of context-modeling as well as context deriving procedures based on consistent previously prepared sensor data can be used for implementations of the *analyzing* component of the MAPE-K loop.

Following the AC paradigm and mainly focusing on monitoring and analyzing, the AutoHome project [6] is a successful example on creation of the context-aware BMS.

The algorithmic know-how, gathered by AmI researchers offers a number of AI approaches meant for generation of correction action schedules or *plans* for adaptive systems. Useful for AmI techniques mainly address pattern recognition, unsupervised machine learning and scheduling. Some of these AI methods are included in [7]:

- Event-Condition-Action (ECA) rules with first order logic or fuzzy logic;
- Artificial neural networks (ANN) perform well but unable to explain output;
- Classification techniques based on decision trees, ECA-rules or ANNs;
- Probabilistic Bayesian networks;
- Sequence discovery techniques based on Markov models or temporal logic;
- Instance based learning using Case-Based Reasoning;
- Evolutional/genetic algorithms requiring the cost function for optimization;
- Reinforcement learning allowing run-time adaptation of recognized patterns.

Our approach to implement the AC *Planning* component is to apply multiple scheduling algorithms, listed above, possibly encapsulated in multi-agent intelligence unit(s). Every intelligence unit agent would generate its own correction action schedule based on the same data and context model delivered from the *Analysis* component of MAPE-K.

The *Execution* component is actually responsible for the best effort selection mechanism for the next action from multiple schedules. To enable action selection each generated *Planning* schedule will be associated with Quality of Schedule meta-information containing qualitative characteristics of the generating algorithm. Several characteristic examples can be algorithm convergence speed, uniqueness of schedule, sensitivity to computation errors, schedule correctness confidence level, etc.

One possible architectural integration approach of multi-algorithmic set for achieving the optimal trade-off between competing application domains can be taken from the hierarchical concept of Semantic Buildings [8]. Different definitions for the optimum have to be considered. As a result, the proposed self-adaptive management system framework will allow choosing the most appropriate for the user and building action planning intelligence approach at runtime of BMS. Technically we plan to integrate the algorithms from the AI libraries like WEKA.

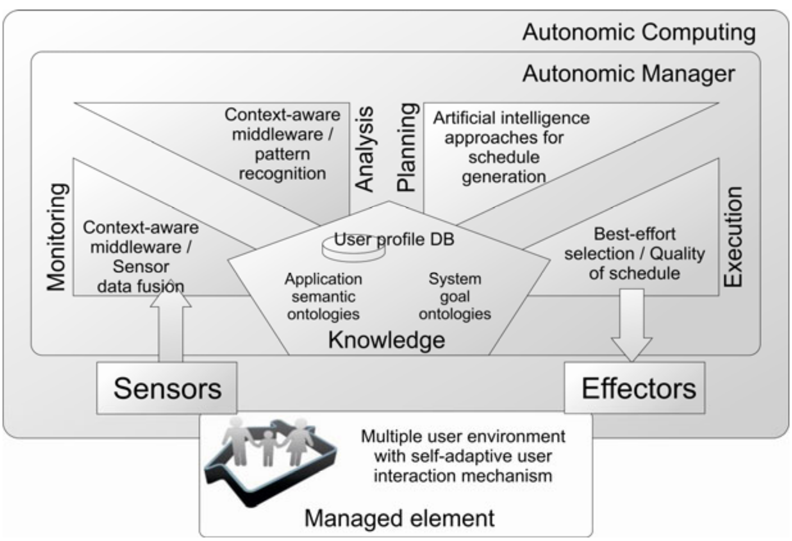

Fig. 1. Intelligent self-managing and adaptive to the user building management system

The simplest form of a *Knowledge* component in AC is rule-based. The main difficulty of the rule-based approach however is in ensuring consistency between all the rules. The more structured way of storing the application knowledge is nowadays implemented through dedicated ontologies, each containing the interconnected data between application terms and their relations. We suggest the following structuring into ontologies for the Building Intelligence Knowledge [8]: higher-level human based BMS semantic goals, application domain activity knowledge (Energy Efficiency, Ambient Assisted Living, Comfort), user type characters, system states, service functionalities, device classification, building space plans and activity time schedules. One of the drawbacks of the ontologies-based approach is the typical need to construct them manually. However, several research activities have been recently devoted to the automatic construction of ontologies [9].

4 Embedding the User-Centric Focus in AC Paradigm

Since the human activity [10] and building performance parameters [11] (BPPs) can be recognized [12] using a unified system of sensors, we suggest that the end-users and buildings are seen as one entity of „Managed Element". From the AC viewpoint this approach will allow considering building's user behavior in the MAPE-K loop without the need to introduce an additional architectural component. The *interaction* between human and building going through standard information input/output devices will be recognized as *logical* sensors and effectors input/output in the AC architecture. Here logical sensors and effectors conform to their definition in context-aware middleware [4, 5] and consist of physical and software sensors and effectors.

Since some parameters of the building cannot be directly influenced by the user, the sensitivity analysis of BPPs to dynamic user behavior [11] has to recognize the sensitive BPPs that may lead to conflicts between BMS decisions and user wishes.

The simulations of indoor user behavior combined with context-modeling and 3D visualization allow finding out the optimal configuration parameters for the control algorithms used by building management systems. Possible solution is given in [11].

The important topic of interaction with user in building automation and control is placed under the research area of human-building interaction (HBI) Current research [13] distinguishes between explicit and implicit human computer interaction (HCI).

The main argument in favor of implicit HCI is in possibility for the user to focus on the task itself and not on dealing with the user interfaces as in case of traditional explicit HCIs like ones with keyboard/mouse [13]. Since the user does not always perceive as s/he interacts with a system, the implicit HCIs are also called disappearing or transparent. As a price for its advantages, the trade-off between invisibility and added value of implicit HCIs has to be considered to ensure the user acceptance.

To ensure the personalized adaptive property of HBI, the system has to rely on the current user context. One implementation challenge in regard of context-sharing between users, applications and user interfaces (UI) is in the choice between context-push and context-pull architecture. In practice however mixed architectures are used to combine the advantages from both approaches in performance and required communication bandwidth. One more challenge of using context as an input is possible problems with predictability of the UI. Generally, the trade-off between stability and adaptation of the UI has to be always found [13].

The other aspect in adaptive HCI is using the current user context for run-time filtering and/or mediating the communication in cases like switching the privacy level upon the arrival of persons without access rights to the previously shown content [13].The promising approach for adaptive user interaction with separating the applications from its user interaction is used by open distributed framework of the EU IST project PERSONA [14]. Using this framework the applications can be developed completely independent from the input / output devices physically available in the smart environment. The framework is also responsible for the personalized adaptive property of HBI by implementing the modality adaptation including modality fusion for input and modality fission for output. The content adaptation by for instance privacy level or location change is also implemented by the framework.

From the AC viewpoint the adaptive user interaction implementation in PERSONA has all the components of the MAPE-K control loop as well.

5 Conclusion, Discussion and Future Work

We proposed an AC-based self-adaptive ambient intelligence framework that could provide increased adaptability in control strategies in building and home automation. This allows achieving better trade-off between constantly changing user needs, building assistive functionality and resource usage. A promising from the AC viewpoint implementation for adaptive user interaction in AmI has been recognized.

We have also identified the research potential in implementation of *Planning* and *Execution* components of AC paradigm in regard of BI / AmI applications. It is clear so far that no AI approach can fully satisfy the AmI requirements on self-adaptability to the constantly changing user needs. Moreover, no *united* run-time AmI simulation tool possibly similar to Building Controls Virtual Test Bed (BCVTB) [15] but integrated with some available context-aware ubiquitous / pervasive computing middleware with the following features together was found:

- Multiple users' involvement by run-time behavior simulation in 3D-space
- Integration of different AI algorithms worth to apply in control strategies
- Integration of self-adaptive user interaction mechanisms

According to the identified gap we suggest the following future research steps:

1. An overview of environments for simulating and modeling of different BI / AmI components like context-aware middleware, AI-algorithms libraries, building information modeling, user behavior models and HBI frameworks;
2. Careful selection, integration and partial development of the BI binding platform like BCVTB [15] based on the results from the first step;
3. Integrating run-time multiple-user behavior in selected BI/AmI environment;
4. Integration of the state-of-the-art adaptive user interaction mechanisms;

5. Unified Integration of the existing AI simulation libraries/tools in order to create the basis for the interoperability of the I/O data between several AI-algorithms from adjacent tasks. The goal is to learn from the user behavior, to generate correcting action plans and control strategies in BMS;
6. Scientifically-proven definition of the performance criteria set of the AI-algorithms from the previous steps. This criteria set can contain for example the convergence speed, accuracy or sensitivity to computational errors and will serve as a ground of BI's action plan selection mechanism. Quality of Schedules generated by every AI algorithm can be introduced;
7. Explore the "best-effort" strategy for action selection based for example on voting, wave propagation or probability mechanisms. As a result we are going to implement this strategy in the BI / AmI action selection algorithm;
8. Defining the 3-4 application benchmarks (for energy efficiency, comfort, safety, security, AAL) within selected BI / AmI environment;
9. Performance evaluation and optimization for our "best effort" selection strategy / algorithm has to be created through simulation framework using application benchmarks. The visualization through performance plots for every application domain and for the whole action selection strategy / algorithm will help to recognize the optimization opportunities;

References

1. Kephart, J.O., Chess, D.M.: The Vision of Autonomic Computing. ACM Computer 36(1), 41–50 (2003)
2. Huebscher, M.C., McCann, J.A.: A Survey of Autonomic Computing\—Degrees, Models, and Applications. ACM Computing Survey 40(3), article 7 (2008)
3. Weiser, M.: The Computer for the 21st Century. ACM SIGMOBILE Mobile Computing and Communications 3(3), 3–11 (1999)
4. Baldlauf, M., Schahram, D., Rosenberg, F.: A survey on context-aware systems. Journal of Ad Hoc and Ubiquitous Computing 2(4) (June 2007)
5. Kjaer, K.E.: A survey of context-aware middleware. In: Proceedings of the 25th Conference on IASTED International Multi-Conference: Software Engineering. ACTA Press (2007)
6. Bourcier, J., Diaconescu, A., Lalanda, P., McCann, J.A.: AutoHome: an Autonomic Management Framework for Pervasive Home Applications. ACM Trans. Auton. Adapt. Syst. 6(1), article 8, 1(212), 1–9 (2011)
7. Aztina, A., Izaguirre, A., Augusto, J.C.: Learning Patterns in Ambient Intelligence Environments: a Survey. Artificial Intelligence Review 34(1), 35–51 (2010)
8. Andrushevich, A., Staub, M., Kistler, R., Klapproth, A.: Towards semantic buildings: Goal-driven approach for building automation service allocation and control. In: Proceedings of IEEE Conference on Emerging Technologies and Factory Automation, pp. 1–6 (September 2010)
9. Drumond, L., Girardi, R.: A Survey of Ontology Learning Procedures. In: Proceedings of the 3rd Workshop on Ontologies and Their Applications, SBIA, Brazil (2008)
10. Kim, E., Helal, S., Cook, D.: Human activity recognition and pattern discovery. IEEE Pervasive Computing 9(1), 48–53 (2010)

11. Hoes, P., Hensen, J.L.M., Loomans, M.G.L.C., de Vries, B., Bourgeois, D.: User behavior in whole building simulation. Energy and Buildings 41(3), 295–302 (2009)
12. Rashidi, P., Cook, D., Holder, L., Schmitter-Edgecombe, M.: Discovering activities to recognize and track in a smart environment. IEEE Transactions on Knowledge and Data Engineering 23(4), 527–539 (2011)
13. Riva, G., Vatalaro, F., Davide, F., Alcañiz, M.: Interactive Context-Aware Systems Interacting with Ambient Intelligence. In: Schmidt, A. (ed.) IOS Press (2005)
14. Tazari, M.-R.: Open Distributed Framework for Adaptive User Interaction in Ambient Intelligence. In: de Ruyter, B., Wichert, R., Keyson, D.V., Markopoulos, P., Streitz, N., Divitini, M., Georgantas, N., Mana Gomez, A. (eds.) AmI 2010. LNCS, vol. 6439, pp. 227–238. Springer, Heidelberg (2010)
15. Wetter, M., Haves, P., Building Controls Virtual Test Bed, `https://gaia.lbl.gov/bcvtb`