A Context Awareness Model Enhanced with Autonomic Features

Ioan Salomie, Ionut Anghel, Tudor Cioara, Mihaela Dinsoreanu
Technical University of Cluj-Napoca
Ioan.Salomie@cs.utcluj.ro

Abstract

The increasing complexity of the context sensitive systems, and the difficulties of their management, administration and adaptation have headed us towards the necessity of integrating self- autonomic computing paradigms (self–configuring, self–healing, self–optimizing and self-protecting) into the development of context sensitive pervasive system's functional components. This paper introduces and defines the concepts of isotropic context space, context granule and context model entropy as the basic features to formally describe and evaluate the RAP context model autonomic characteristics. The paper also propose a methodology for enhancing the RPA context model with self-configuring and self-healing autonomic properties. The self-configuring property or context adaptation is obtained by detecting / configuring / integrating new context resources / actors into the context specific model. The context model self-healing properties are obtained by continuously monitoring the real context for detecting broken context policies and executing compensating actions.*

1. Introduction

Pervasive systems are complex heterogeneous distributed systems that feature a large number of devices and services that need to dynamically adapt to the conditions and changes of their environment. Pervasive, context sensitive systems, continuously monitor / capture and interpret the environment related information in order to assure high context awareness.

Due to the complexity and continuous evolutions of the environment where the systems are integrated and executed, their management has become extremely difficult which headed us towards using the autonomic computing paradigms (self–configuring, self–healing, self–optimizing and self-protecting) for the development and integration of self-* enhanced components into context sensitive systems.

The concept of autonomic computing and its impact to nowadays computing systems has been introduced by IBM [1] and refers to the adoption of self-* paradigms in the development of highly complex IT systems, similar to the biological autonomic systems. Currently, the development of autonomic components and systems is an open research direction.

The objective of this paper is to enhance the RPA basic context model with autonomic features (figure 1). To achieve this objective the following new concepts are defined: *the isotropic context space*, *the context granule* and the *context model entropy*. Based on these concepts, the self-configuration and self-healing autonomic computing paradigms are specified for the RPA context model.

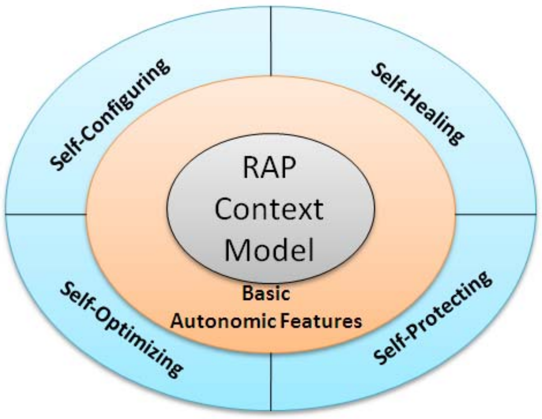

Figure 1. Autonomic enhanced context model

The rest of the paper is organized as follows: in section 2, we present a research domain review by highlighting the most important research directions. In section 3 the RPA context model and its main elements are briefly presented. In section 4 we will define the concepts used to enhance the context model with autonomic properties. Sections 5 and 6 show how the self-configuration and self-healing properties are added to the context model while section 7 gives conclusions and promising future work.

2. Related work

The research efforts in this domain are concentrated on three directions: developing computational resources with autonomic features, autonomic adaptation of computational systems to the context and defining context models with autonomic properties.

The existing results (briefly described below) propose systems with autonomy limited to only some of the four **CHOP** (**C**onfiguring, **H**ealing, **O**ptimizing and **P**rotecting) properties.

In the domain of **self-configuration**, the research efforts concentrate on developing models for managing the automatic discovery, installation and configuration of complex computational systems and their components. The goal is to obtain self-configuring systems that can automatically react to new components installation, discover the needed services or adapt and learn the behavior of the new components. The main research problem is to specify and represent the configuration, discovery and integration of system requests [2], [3]. In [3] Bahati proposes a self-configuration model based on directives. All the configuring directives for a component represent the self-configuration policy of that component. The self-configuration policies are stored in a repository attached to the system, which is queried when a new component is added. The main disadvantage of this approach is the rigid and static way (design time) of representing the directives, thus failing to provide the ability to modify or add directives during run-time. Also, the model lacks the automatic learning of the newly added component's characteristics and the dynamic creation of new policies.

The research related to **self-optimization** focus on obtaining models that allow systems to work 24/7 at maximum capacity and which optimize the resources usage by (i) establishing metrics for evaluating the system efficiency, (ii) specifying QoS models which allow the selection and configuration of components for optimal execution and (iii) monitoring the autonomic system life cycle for replacing the inefficient components. In [4] the authors proposes a model based on the analysis of the autonomic system parameters and on performing automatic updates for maintaining the system in the optimal state. The system components properties and parameters are monitored and the obtained values are registered at fixed periods of time (checkpoints). The obtained information is used for the analysis and detection of optimal values and for the updating the system.

The researches for **self-healing** focus upon specifying and developing autonomic system models that should: (i) identify the causes of system failures or crashes, (ii) identify possible errors in the system life cycle and (iii) perform diagnosis and offer solutions [5]. The proposed models have limited self-healing properties, anticipated at design-time. The models are based on anticipating problems that can appear at the execution for avoiding them and on identifying measures to restore the system to a previous functional state in case of failure.

The researches related to **self-protection** are based on designing models which allow the protection of autonomic system against attacks and to keep the information secure [6]. The current approaches use complex methods to secure system's components and also use the analysis of security problems for identifying the causes that lead to such problems in order to prevent their appearance in the future [8].

Research efforts are made to create new models and algorithms that allow computational systems to decide or execute some specific actions according to the context or situation in which they execute. The objective is to associate a certain degree of intelligence to the computational systems for **context adaptation**. In [7] the authors propose a context adaptive platform based on the *closed loop control principle*. The unique element of this proposal consists of defining and using the concept of *application-context description* to represent system knowledge about the context. Using this knowledge representation, the system becomes aware of its relation with the context (application-context relation). This description is frequently updated and used for the system control allowing the system to reconfigure and take adapting decisions. [9], [10] and [11] propose a context adaptation model based on defining the system behavior in a certain contextual situation using a set of *context adapting rules*. A rule is composed of a context condition and an associated action. The main disadvantage of these approaches consists of statically specifying (design time) the context adapting rules (new rules cannot be inferred at run time).

Another vision for context adaptation consists of using context information described at design time as high level policies for adapting the system to the situation that could appear at run-time [12]. This model is used for web service orchestration at run-time, according to the context in which they will be executed. In [13] the authors propose a model for interpreting context situations using the *CIBFR fuzzy* *logic (A Context Interpreter Based on Fuzzy Rule)*. By applying the fuzzy logic, functions that allow the classification of ambiguity degree of a concept are defined within the model.

A new research direction is to define **context models with autonomic properties**. In [14] O'Connor proposes a self-adapting context model based on the construction of a system situation space and representing the system execution context as a situations group within this space. A situation consists of the sampled values at a certain moment given by the set of sensors associated to the system. Based on this context definition, a function can be defined on the set of situations that form the context, having as output values from the action set that the system executes. It is noticed that this function has an interesting behavior: the situations that determine the system to execute the same action (i.e. have the same meaning) tend to form groups in the situations space. Using machine learning algorithms, the system can predict which action to be executed when a new situation appears, by placing it in the situations space group. The main disadvantage of this approach is given by the growth of the problem complexity with the increasing of the number of sensors that provide context information to the system.

3. The RAP context model

The **RAP context model** is defined as a triple **C = <R, A, P>,** where **R** is a set of context resources, **A** is a set of actors which interact with context resources, **P** is a set of context related policies [17].

In our context model a **context resource** is a physical / virtual entity which generates and / or processes context information. A context resource has a unique identity, can be annotated with semantic information and features *Resource Properties*, *Resource Services* and *Resource Influence Zone*. The **Resource Properties** consists of the set of relevant context information that a resource can provide. In the context model the resource functionality is specified by **Resource Services** (i.e. a service that locates / updates an object). The **Resource Influence Zone**, **Z(r)**, is the 3D physical space (a 3D spatial bubble) in which a resource captures / provides context information or in which the resource presence can be sensed (it becomes visible for an actor or for another resource).

An **actor** represents a physical or virtual entity that interacts directly with the context or uses the context resources to fulfill its needs.

A **policy** represents a set of rules that must be followed by actors or resources present in the context. The consequence for broking a policy determines context elimination for the actor or resource that has committed the fault.

The context model features complex processes for management, administration and adaptation. The RAP model management is achieved by using BDI agents as processing units due to their reactivity, collaboration, inference and adaptability properties. The agents are defined using the concepts provided by the SOUPA (Standard Ontology for Ubiquitous and Pervasive Applications) [15] BDI agents' sub-ontology [16]. We define and formalize four kind of generic agents which are instantiated for every specific context model: *Context Model Administering Agent, Context Interpreting Agent, Request Processing Agent, Execution and Monitoring Agent*. The **Context Model Administering Agent, CMAA,** is the manager of a specific context model and has as it's main task the synchronization of the specific context model with the real context. The **Context Interpreting Agent, CIA,** is used for evaluating the context information from a context instance so that the instance can be correctly represented into a semantic states space. The **Request Processing Agent, RPA,** is used for processing the requests coming from actors in a specific context model. This agent also identifies and / or generates action plans that must be executed for serving a request. The **Execution and Monitoring Agent, EMA,** processes the plan received from the RPA agent and executes every plan action using the resource available services.

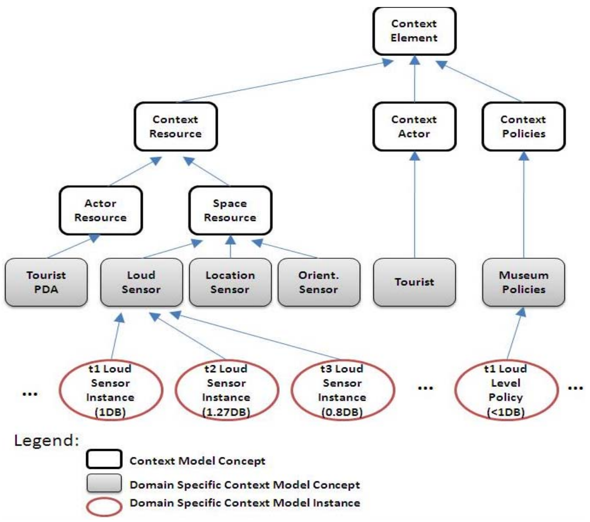

Figure 2. The context model ontology

The context model elements and there is-a relationships represent the context ontology core (figure 2).

The context model will be mapped onto different real contexts by populating the model defined sets with real context specific elements. The mapping result is a **specific context model** defined as: $\mathbf{C_S = <R_S, A_S, P_S>}$. A specific context model accurately reflects the real context and should be permanently kept consistent with the real context by the administering agents. The specific context model concepts are represented as sub trees of the context model core ontology (figure 2).

An **instance of the specific context model** represented as a tuple $\mathbf{CI_a^t = <R_a^t, P_a^t>}$, is used for describing the actor – context interaction. An instance contains the set of resources with which an actor can interact together with their values at a specific moment of time *t* (the context specific ontological concepts instances).

4. Towards an Autonomic Context Model

In order to enhance the RAP basic model with autonomic capabilities we introduce the following three new concepts: *isotropic context space*, *context granule* and *context model entropy*.

4.1. Isotropic context space

A context sub-space (part of the whole context space) is **isotropic** if and only if the set of sub-space attached resources R_S is invariant to the movements of all actors in the context sub-space. In other words, in an isotropic context sub-space, the R_S set is the same for all the actors that are physically located in the influence zone of the R_S resources. It should be noted that if Card (R_S) = 1, the context sub-space is isotropic. From now on, the context sub-space will be also considered and referred as a context space.

Given a non-isotropic context space, the variation degree of the space isotropy $\Delta_{\mathbf{IZ}}$ is defined as the variation of the $\mathbf{R_s}$ set, while the actor moves in the context space.

4.2. Context granule

Usually, a context space is non-isotropic but it can be split in a set of disjunctive isotropic context space volumes. We define the **Context Granule (GC)** as the maximum volume of a context space where the space isotropy degree variation is zero: $\Delta_{IZ\ GC\text{-}GC} = \mathbf{Ø}$.

In a given moment of time, an actor can be physically located in a single context granule. As a result, $\Delta_{\mathbf{IZ}}$ is non-zero only when an actor moves between context granules.

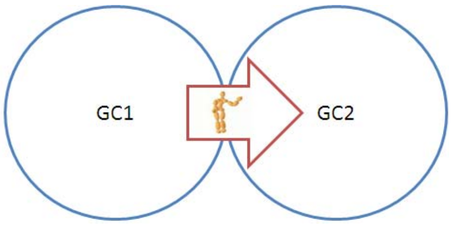

Figure 3. A context space with three context granules

Figure 3 shows two context granules GC1 and GC2 and an actor. When the actor moves from GC1 to GC2 the space isotropy degree variation Δ_{IZ} is determined as:

$$\Delta_{\mathbf{IZ\ GC1\text{-}GC2}} = \{\mathbf{R_{GC1} \setminus R_{GC2}}\}\ \mathbf{U}\ \{\mathbf{R_{GC2} \setminus R_{GC1}}\} \quad (1)$$

If $\Delta_{\mathbf{IZ\ GC1\text{-}GC2}}$ equals **Ø** the actor remains in the same context granule.

$$\Delta_{\mathbf{IZ\ GC1\text{-}GC2}} = \mathbf{Ø} \Rightarrow \mathbf{R_{GC1} = R_{GC2}} \quad (2)$$

Let's consider the example illustrated in figure 3 and the following presumptions: (i) the context granule GC1 contains a temperature sensor and (ii) the GC2 context granule contains both a temperature and a humidity sensor. We can identify two scenarios for actor evolution within the context:

1. The actor leaves GC1 for GC2.

$\Delta_{\mathbf{IZ\ GC1\text{-}GC2}}$ = {$R_{GC1} \setminus R_{GC2}$} U {$R_{GC2} \setminus R_{GC1}$} =
[{ tempSensor } \ {tempSensor, humiditySensor}] U
[{tempSensor, humiditySensor} \ {tempSensor}] =
Ø U { humiditySensor } = **{ humiditySensor }**

2. The actor moves only within GC1.

$\Delta_{\mathbf{IZ\ GC1\text{-}GC1}}$ = {$R_{GC1} \setminus R_{GC1}$} U {$R_{GC1} \setminus R_{GC1}$} =
[{tempSensor} \ {tempSensor}] U
[[{tempSensor} \ {tempSensor}] = {Ø} U {Ø}
= **{Ø}**

4.3. Context model entropy

In the RAP context model [17], the context policies are specified by the **P** set as $\mathbf{P = \{pol_i \mid i>0\}}$. Each context policy consists of a set of predefined rules $\mathbf{pol_i=\{rls_i \mid i>0\}}$.

We define **Rf**, as a function over the model's policy rules that evaluates if a certain rule is broken or not:

$$\mathbf{Rf}: \mathbf{rls_i} \rightarrow \{0,1\},$$

$$\mathbf{Rf(rls_i)} = \begin{cases} \mathbf{0, rls_i \text{ rule is not broken}} \\ \mathbf{1, rls_i \text{ rule is broken}} \end{cases} \quad (3)$$

The overall policy evaluation function **Pf,** measures the policy fulfilling degree:

$$\mathbf{Pf}: \mathbf{pol_j} \rightarrow \mathbb{Z},$$

$$\mathbf{Pf(pol_j)} = \textstyle\sum_{i>0} \mathbf{Rf(rls_i)} = \begin{cases} \mathbf{0, \text{policy is fulfilled}} \\ \mathbf{>0, \text{policy is not fulfilled}} \end{cases} \quad (4)$$

We define **E(C$_S$)** as the specific context model entropy (the level of disorder) reflecting the degree of fulfilling the context policies (all context policies are respected when **E** = **0**).

$$\mathbf{E}: \mathbf{C_S} \rightarrow \mathbb{Z},$$
$$\mathbf{E(C_S)} = \textstyle\sum_{i>0} \mathbf{Pf(pol_i)} \quad (5)$$

The entropy **E** is used to globally determine the autonomic capabilities of the specific context model as follows:

$$\mathbf{E(C_S)} = \begin{cases} \mathbf{0, C_S \text{ is in an autonomic state}} \\ \mathbf{> 0, C_S \text{ is not in an autonomic state}} \end{cases} \quad (6)$$

To ease the entropy evaluation process, the set of policies is split into four sub-sets, each corresponding to one of the four Self* autonomic paradigms. Presuming that the P set is split into four disjunctive sets, the entropy of the specific context model can be evaluated as a sum of each sub set entropy as shown below:

$$\mathbf{E(C_S)} = \mathbf{E(P_{SelfConfiguring} \cup P_{SelfHealing} \cup P_{SelfOptimizing} \cup P_{Self\ Protection})} = \mathbf{E(P_{SelfConfiguring}) + E(P_{SelfHealing}) + E(P_{SelfOptimizing}) + E(P_{Self\ Protection})} \quad (7)$$

A specific context model **C$_S$** features autonomic behavior if the autonomy invariant (8) is always true.

$$\mathbf{E^t(C_S) * E^{t+1}(C_S) = 0} \quad (8)$$

5. Adding self-configuring capabilities to the RAP context model

The property of **self-configuring** or context adaptation is obtained by detecting and configuring that context resources / actors that determine real context variations. When a variation is detected, the CMAA agent performs a negotiation stage that has as the outcome the configuration of a new resource / actor according to the context policies. The context model self-configuring process always ends by creating a new specific context model adapted to the new real context (i.e. adding / eliminating a context resource / actor form the specific context model).

The self-configuring property of the context model is enabled only if the autonomy invariant (8) holds for $C_S = \langle R, A, P_{SelfConfiguring}\rangle$.

The problem that arises is to evaluate the real context variation Δ_C and determine when the self-configuration process must be started.

It was shown that in the RAP context model, the **Context** abstraction is given by the set of all context properties in terms of the relevant information provided by its resources:

$$\mathbf{Context \equiv K = K_A \cup K_s} \quad (9)$$

The context variation (10) is obtained by adding the variation generated by the physical space and variation generated by actors:

$$\Delta_C = \Delta_{CS} \cup \Delta_{CA} \quad (10)$$

5.1. Evaluating the physical space variation

The physical space context variation is generated (i) by the space isotropy degree variation Δ_{IZ} and (ii) by attaching / detaching a space context resource. The space isotropy degree variation is generated by the actor mobility as a result of migrating from a context granule to another context granule. On the other hand, attaching / detaching a space resource generates a real context variation and therefore, specific context model adaptation is necessary. In this case the specific context model self-configures itself by adding / eliminating the resource.

As a result, the physical space variation is calculated as follows:

$$\Delta_{CS} = \Delta_{IZ} \cup \{\mathbf{R_S^{t+1} \setminus R_S^t}\} \quad (11)$$

The space isotropy degree is zero only if the context actors are moving inside a context granule. In this case in the relation (11) $\Delta_{\mathbf{IZ}} = 0$.

The specific context model self-configuration process should start when Card($R_S^{t+1} \setminus R_S^t$) >= 1 which means that a new resource has been added to the context or an existing resource has been removed from the context.

When Card(Δ_{IZ}) >= 1 a new context specific instance must be generated.

5.2. Evaluating the context variation generated by actors

The context variation generated by an actor is given by the context resources attached to the actor (i.e. the resources used in the actor-context interaction process). In a given context, an actor is characterized by a large number of actor – context interaction patterns. Only two of these patterns determine the specific context model modification: (i) the actor enters the context and (ii) the actor leaves the context.

If $A^t = \{ a_k \mid k>0 \}$ is the set of all actors physically located in the context influence zone at a given moment t and $\mathbf{R_A}$ is the set of context resources attached to the actors then:

$$\boldsymbol{\Delta_{CA} = \{ A^{t+1} \setminus A^t \} \cup \{ R_A^{t+1} \setminus R_A^t \}} \qquad (12)$$

The specific context model self-configuration process should start when Card($A^{t+1} \setminus A^t$) >= 1 and should have as a result the specific context model modification by adding / removing an actor and its resources form the specific context model.

6. Adding self-healing capabilities to the context model

The context model has the self-healing property if for any moment of time t with $E_t(C_S) > 0$, the context model is capable to identify the set of broken rules attached to a policy and to generate a new context specific model at $t+1$ with $E_{t+1}(C_S) = 0$.

The self-healing property of the context model is enabled only if the autonomy invariant (8) holds for $C_S = <R, A, P_{\text{SelfHealing}}>$.

In the RAP model, the self-healing property is enforced by the Model Administering Agent (CMAA), the Context Interpreting Agent (CIA) and the Execution and Monitoring Agent (EMA).

The **CMAA** agent continuously monitors the real context for detecting the broken context policies and executing compensating actions. It also reconfigures the resource or the actor that breaks the identified context policy. If the reconfiguration stage fails, the resource / actor is removed from the context.

The **EMA** agent implements the self-healing property by monitoring the execution of the actions plan and taking compensation actions in order to keep the context in a consistent state when the actions plan execution fails.

The **CIA** agent implements self-healing properties in order to achieve (i) the semantic space construction and (ii) the evaluation of the context instances for determining and placing the associated semantically value in the semantic space. The Semantic States Space is a semantic hyper-space which dimension is equal with the number of context resources from the context model.

The semantic space and the hyper-space semantic zones are constructed by the CIA agent using context policies, context ontology and reasoning / learning algorithms. The semantic value of a context instance is a unique hyper-point in the hyper-space determined by positioning all the resources values from the context instance on the semantic axes. The semantic values attached to the context instances that will determine the execution of the same actions form groups in the semantic space. Figure 4 presents a semantic space example obtained from two context resources: humidity and temperature sensors.

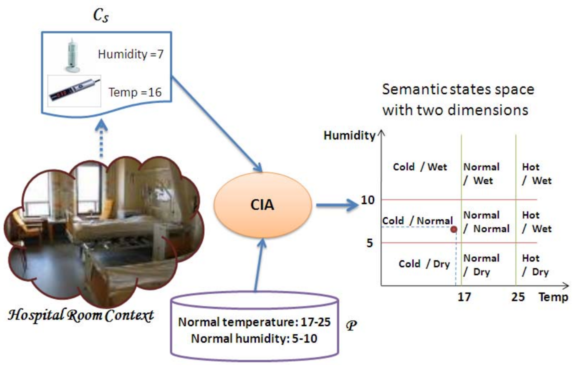

Figure 4. Representing a semantic space for a two resource context

The self-healing property is obtained by periodically evaluating the groups of the semantic space, targeting to avoid the forming of isolated semantically values.

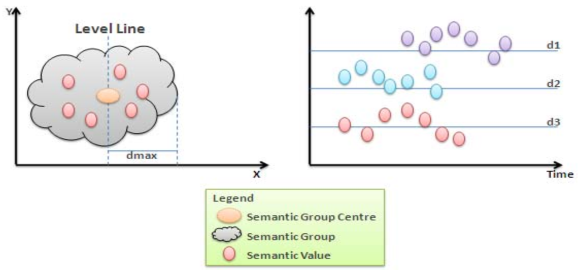

Figure 5. The semantic values polarization in time

In a semantic space, the time footmark of the semantic values belonging to a group follows a polarization line (level line) that is parallel with the time axis (figure 5). The self-healing property should be implemented by permanently checking the semantically values polarization degree and identify the values that don't follow a level line. For these values, associated plan must be identified and executed and then, the context state must be verified by the EMA agent for finding inconsistencies and for executing compensating actions, even if the plan execution does not fail.

7. Conclusions and future work

In this paper the concepts of isotropic context space, the context granule and the context model entropy are introduced and defined as the basic features to formally describe and evaluate the RAP context model autonomic characteristics. Using these concepts we have formalized for the RAP model two of the four self-* main autonomic computing paradigms: self-configuring and self-healing. The self-configuring paradigm or context adaptation is obtained by detecting / configuring / integrating the new context resources / actors in the context specific model. The context model self-healing properties are obtained by continuously monitoring the real context for detecting the context broken policies and executing compensating actions.

For the future development, we intend to enhance the RAP context model with self-protecting and self-optimizing features and provide a general formalism that will include all four self* paradigms in the RAP context model.

Acknowledgements

This work was supported by the Food Trace project within the framework of the "Research of Excellence" program initiated by the Romanian Ministry of Education and Research.

8. References

[1] Jeffrey O. Kephart, David M. Chess, *The Vision of Autonomic Computing*, IBM Thomas Watson Research Center; IEEE Computer Society, pages 41-50, January 2003

[2] Radu Calinescu, *Model-Driven Autonomic Architecture* Fourth International Conference on Autonomic Computing (ICAC'07), Florida, USA, June 11-15, 2007.

[3] Raphael M. Bahati, Michael A. Bauer, *Using Policies to Drive Autonomic Management*, 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'06), Niagara-Falls, Buffalo-NY, USA, 26-29 June, 2006.

[4] Eleni Patouni, Nancy Alonistioti, *A Framework for the Deployment of Self-Managing and Self-Configuring Components in Autonomic Environments*, International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'06), Niagara-Falls, Buffalo-NY, USA, 26-29 June, 2006.

[5] Salomie, I. Moga Al., Dinsoreanu, M., Soos, *Enhancing Web Service Composition with Self-healing Facilities*, WSEAS Transactions on Information Science and Applications , Issue 1, Volume 4, January 2007, ISSN 1709-0832, pag. 42-50.

[6] Mirco Musolesi and Cecilia Mascolo, *Evaluating Context Information Predictability for Autonomic Communication,* International Symposium on a World of Wireless, Mobile and Multimedia Networks WoWMoM'06

[7] Marcel Cremene, Michel Riveill, Christian Martel, *Autonomic Adaptation based on Service-Context Adequacy Determination*, Electronic Notes in Theoretical Computer Science, Elsevier, 2007

[8] Yingping Huang, Gregory, *Autonomic Web-based Simulation* 38th Annual Simulation Symposium (ANSS'05), San Diego, USA, April 3-7, 2005.

[9] Brecht Desmet, Jorge Vallejos, Pascal Constanza, Robert Hirschfel, *Layered design approch for context-aware systems*, First International Workshop on Variability Modeling of Software – Intensive Systems, Limerick, Ireland, 2007

[10] Luiz Olavo Bonino da Silva Santos, Fano Ramparany, Patricia Dockhorn Costa, Peter Vink, *A Service Architecture for Context Awareness and Reaction Provisioning*, 2007 IEEE Congress on Services

[11] Frederick Seyler, Chantal Taconet, Guy Bernard, *Context Aware Orchestration Meta-Model*, Third

International Conference on Autonomic and Autonomous Systems (ICAS'07)
[12] Yu Chen Zhou, Xin Peng Liu, Eduardo Kahan, Xi Ning Wang, Liang Xue, Ke Xin Zhou, *Context Aware Service Policy Orchestration*, 2007 IEEE International Conference on Web Services (ICWS 2007)
[13] Qin Huaifeng, Zhou Xingshe, *Integrating Context Aware with Sensornet*, Proceedings of the First International Conference on Semantics, Knowledge, and Grid 2006
[14] Neil O'Connor, Raymond Cunningham, Vinny Cahill, *Self-Adapting Context Definition*, First International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2007)
[15] Harry Chen, Tim Finin, Anupam Joshi, *The SOUPA Ontology for Pervasive Computing*, Ontologies for Agents: Theory and Experiences, July 2005
[16] Feruzan Ay, *Context Modeling and Reasoning using Ontologies,* University of Technology Berlin*,* July 2007
[17] ***, *Another paper submitted by the authors*