Autonomic Networking: Prototype Implementation of the Policy Continuum

S. van der Meer[1], A. Davy[1], S. Davy[1], R. Carroll[1], B. Jennings[1], J. Strassner[2]

[1] Waterford Institute of Technology, Telecommunications Software & Systems Group, Cork Road, Waterford, Ireland
{vdmeer, adavy, sdavy, rcarroll, bjennings}@tssg.org
http://www.tssg.org
[2] Motorola Labs, Schaumburg, IL USA
john.strassner@motorola.com

Abstract. The focus of new service development continues to move from network services towards application services. This is driven by many factors, such as converged networks, ambient networks and seamless mobility. These and other efforts enable customers to move from one service provider to another based on the availability of desired application services, as well as metrics concerning their performance (e.g., reliability and cost). In order to enable rapid deployment of these new services, policy becomes paramount. This paper presents first experiences with the implementation of the policy continuum, which represents one of the cornerstones of Autonomic Networking. It allows for the transformation of high-level (e.g. business) policies towards low-level (e.g. device) policies and enables seamless mobility and automated network and resource configuration. Based on a novel architecture, this paper shows our first proof of concept implementation of the policy continuum with results from a simulated network.

1 Introduction

Policy-based network management has been the focus of research and development over the past several years. Much of this research has focused on the development of models and languages for the representation and specification of policy. While this is important, it is not sufficient to move policy-based network management (PBNM) out of the laboratory and research environments into widespread commercial deployment.

Widespread commercial deployment of PBNM is dependant on the integration of a consistent and coherent approach to policy that extends from the business support systems through the network to the subscriber units at the very edges of the network. In order to enable the pervasive deployment and utilization of policies, three requirements are identified: 1) Definition of a Business policy language for the Policy Continuum; 2) Translation of business policies into network device policies using the policy continuum; and 3) Automatic network node configuration according to business policies.

These problems must be solved in concert, as opposed to separately, in order to develop a PBM architecture that can scale to meet the needs of a Service Provider or large Enterprise. Then, we can consider future applications of PBM, such as for autonomic computing [12] and seamless mobility [13]. For example, how can policy control the reconfiguration of a network element in the face of changing functionality of the network element, changing demands of its users, and changing environmental conditions?

The rest of this paper is organized as follows. First, we briefly introduce the Policy Continuum (section 2) and the general architecture we are using (section 3). Section 4 shows how policies are processed automatically in order to simulate (for this phase of the project) the Policy Continuum. Section 5 defines the scenario we are using to simulate and test the implementation. Section 6 shows how phase one of our implementation realizes the Policy Continuum, explains the simulation environment and shows graphs and figures from a simplified trial within the given scenario. Finally, sections 7 and 8 discuss conclusions and provide references, respectively.

2 The Policy Continuum

The policy continuum was developed in order to enable multiple constituencies, which have different concepts and terminologies, to co-define and -develop policies. This approach was presented in [3] and [2] (cf. Figure 1). Most systems define the notion of a policy as a single entity. This is incorrect. For example, there are policies to represent business rules, policies to control customer rebates, and even policies to represent configuring a feature of a device. There is little in common with these policies, because they use different grammars to express their function, and are used by different constituencies. However, they are often in reality related to each other in that they provide a different view of tasks that may be part of a some overall business function or goal.. This is the theory behind the policy continuum, an example of which is shown in Figure 1.

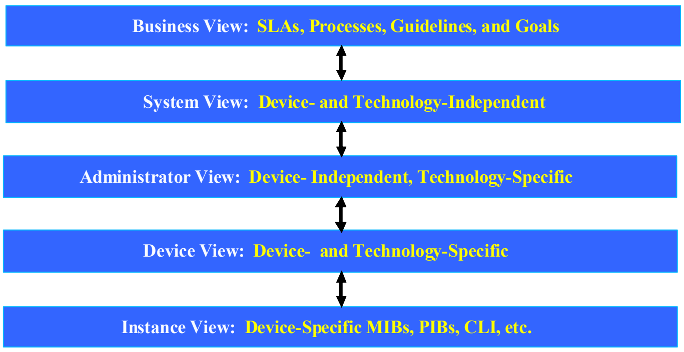

Fig. 1. The Policy Continuum with Example

Each level of the policy continuum is optimized for a different type of constituency that needs and/or uses information of a specific nature. For example, the business user wants Service Level Agreement (SLA) information, and isn't interested in the type of queuing that will be used to forward traffic. Conversely, the administrator of the network may want to develop specific commands to program the device, and may need to have a completely different representation of the policy. DEN-ng [2] [6] defines a set of views to support the needs of different constituencies. This is similar to the RM-ODP concept of viewpoints [4], except that DEN-ng defines a set of strongly related views. This enables the needs of different constituencies to be associated with each other. For instance, a business rule can be translated into command changes to govern changes in a device configuration. The four DEN-ng views are identical to the NGOSS views [5], which enables NGOSS to be used to help attain our goals.

The DEN-ng information model is not a single model, but rather a set of models, one per view. Each view has its own grammar and terminology, which enables each view to express the needs of a particular set of constituencies through specific semantics and structure. This enables policy to be treated as a continuum, where policies in different views are related to each other through model mappings [2] of the DEN-ng information model views. A model mapping is a translation from one type of model to another type of model. A model mapping changes the representation and/or level of abstraction used in one model to another representation and/or level of abstraction in another model. This provides a layered set of policies with different levels of abstractions, and model mappings to translate between them.

3 Autonomic Network Reference Architecture

Our autonomic network reference architecture (ANRA) uses four distinct architectural constructs: shared information, virtual software, infrastructure, and policy [1].

Shared Information – Policy definition and structure is based on DEN-ng [6], which is the foundation of TMF's Shared Information and Data (SID) model [2] [7]. DEN-ng is based on UML, and uses abstraction mechanisms and software patterns to provide an extensible structure that can integrate other models (whether or not they are UML compliant) and technologies as appropriate. Object-oriented information modeling provides a common language for representing the characteristics and behavior of entities in the managed environment in a single, extensible lingua franca. This extensibility is provided through patterns [8], roles [9], and various other abstraction mechanisms. DEN-ng is a backbone framework that is used to translate and refine high level business policies to low level network element policies, and supports the formation of the context model for the virtual software layer.

Virtual Software – The purpose of the virtual software is to support autonomic functionality for different heterogeneous networks and components. Autonomic Networks achieve governance through (intelligent) decision-making. Hence, this software contains a model of the desired behavior as well as the deduced current behavior of the system or component being managed. One solution towards achieving governance and intelligent decision making is through ontology [11]. An ontology can be defined as a formal, explicit specification of a concept, along with its taxonomy, a set of interrelationships, and a set of rules that govern how the concept is used and reacts with other concepts. An ontology can enable inferences to be made about the knowledge that they contain. Note that most information models do not contain the semantics and constructs needed to do this.

Infrastructure – The infrastructure includes network elements and other computing devices as well as the software that manages them. Since network infrastructure is composed of heterogeneous devices, the infrastructure layer supports a model mapping layer that enables different devices to communicate with each other. At the same time, this layer also support control for different types of protocols and adaptability mechanisms depending on the current context as well as the type of behavior that is being orchestrated.

Policy – Policies formalize the concept of decision making. This element of the architecture is based on the DEN-ng policy model [2]. The different levels of abstraction of the Policy Continuum are refined using a combination of information modeling, ontological engineering, and knowledge-based reasoning for each level of the continuum. The information model is used for two distinct purposes: (1) to represent policies at each level using the same set of building blocks, and (2) to explicitly define which set of managed entities are the subject and target of the policy.

4 Automated Processing of Policies

The implementation architecture of the prototype is illustrated in Figure 2. The architecture is organized into three separate modules, which includes a policy wizard graphical front end (later called GUI), a policy processing unit, and an OPNET backend. This backend will be replaced in future iterations with a testbed. The core of the policy processing unit is the Policy Continuum (in Figure 2 represented by all 5 layers) and the XML Transformation Logic, which allows the generation of XML so that each level can be easily interfaced to a particular constituency.

The policy wizard GUI module provides a simple business user interface specifying objects through a drop down menu. These objects include, (i) the different customers and their associated policies, (ii) the available subscription package types for the customers (Gold, Silver, or Bronze), (iii) subscribed content service list for the selected package (VoIP, VoD, Web data), and (iv) various grade metrics for each type of content service (e.g. grade 1 - 3).

Following the selection of the different components, the objects are captured and translated into a restricted natural language that provides a business-friendly language presented on the screen through the policy definition panel. This processing task is performed by the Natural Language Processing component, and as shown in Fig. 2 is passed through a feedback loop to the Policy Wizard GUI for presentation. An example of an output presented in a policy definition panel includes:

```
Start Policy Definition
    John has purchased Gold Service package type
    John services and minimum response time selection
    includes:
        VoIP with grade 1
        VoD with grade 2
        Web data with grade 3
End Policy Definition
```

Each level of the policy continuum has an associated set of objects, e.g. sub-sets of the policy language expressed by one specific dialect per level. The Policy processing component is also responsible of capturing objects selected in the GUI and requesting that appropriate XML is generated for each level of the policy continuum. The mapping process from the continuum to XML is performed by XML pipelining, where each layer of the policy continuum will be represented by a dialect of the policy language. The policy transformations between the layers are performed by corresponding XSLT documents. As shown in Figure 2, the XML file is transformed to a suitable representation that can be passed to the OPNET backend. This is performed through the OPNET interface.

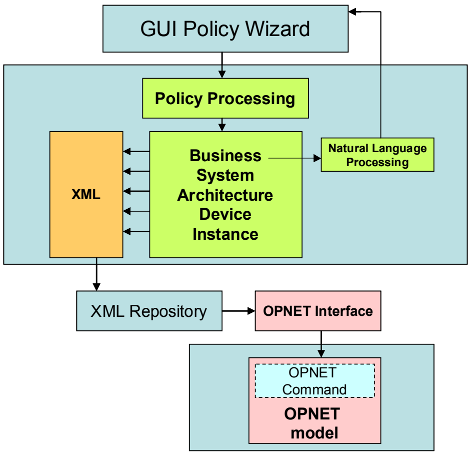

Fig. 2. Implementation Architecture

5 Scenario

The scenario depicted for the prototype implementation presents the ability to synthesize different requirements from multiple constituencies that are all working on the same solution. This captures the unique requirements of each constituency in a particular policy continuum that is represented by a unique interface and grammar; the grammar corresponds directly to both interface components as well as DEN-ng model objects. All constituencies are connected by the different levels in the continua. In this particular case, the scenario demonstrates the changes at the business level, which will be transformed into suitable low level network configuration commands. The advantage of this approach is that it maintains the concepts and terms that users in each level of the continuum are used to, while providing a way to integrate their requirements into a single coherent solution.

The main contribution that this scenario demonstrates is the definition of a business interface to the policy system, the translation of business level to network device level configuration policy, and the automatic configuration of network entities according to business policy.

The scenario that is depicted for the demonstration is an Internet Service Provider (ISP) that controls the resource management for different customers that are subscribed with the service provider. The ISP has one particular customer by the name of John who has subscribes to a particular package with the service provider and at the same time also has subscription to a number of content services (e.g., VoIP, VoD, and broadband), where John also has the ability to specify the priority grade for each application service (e.g. for VoIP, John has selected grade 1 (Grade 1 – 3, where grade 1 is the best). Currently, John's VoIP is selected as grade 1, VoD is selected as grade 2, and web application is selected as grade 3. However, John is a big soccer fan, and thus is keen to watch the upcoming premiere league match between Manchester

United and Chelsea. Therefore, for the VoD service, John has decided to select grade 1 for an hour before and after the match (in order to be able to listen to all the pre- and post-analysis of the match) in HD (High Definition) video mode. In order to do this, John contacts his ISP who uses the Policy Wizard to change the grade level for VoD.

The scenario will demonstrate the changes that are made at the network configuration due to the changes that John's ISP has made at the business level. Since the network that John is currently subscribed to conditions its traffic using Diffserv QoS, the network configuration changes will illustrate changes in resource management at the Edge and Core DiffServ routers. The changes include the improved level of QoS for VoD and also degraded service for VoIP and web services. We assume here that each customer has a set amount of bandwidth allocated by the ISP.

6 Implementation

The preliminary prototype implementation for the policy continuum includes a Policy Wizard GUI, and a simple policy continuum that maps between the business language and the configuration language. The Policy Wizard evaluates the objects selected from the GUI and processes the objects into natural language. The captured objects are then passed to the Policy language represented in XML and transforms to a representation suitable for OPNET simulator through the OPNET interface. This process demonstrates business level policy refinement to network device level configuration policy.

6.1 The Policy Continuum

The business view is supported by a business policy language that supports the description of business goals, e.g. "Offer gold, silver and bronze service packages". This will be mapped towards the system view, which is able to express that "customer has selected gold service package, with 1 to 3 content services and a service grade attached to each content service". This information is further mapped to the relevant network architecture (e.g., "need to provide Diffserv configuration for 2 Mbps, with grade one service 50% bandwidth, service grade 2 of 30% bandwidth, and service grade 3 of 20% bandwidth"). The device view shows how this information is used to reconfigure the Core and Edge routers of the Diffserv network. The instance view shows the specific configuration commands on a per-device and –vendor basis. The policy continuum will be shown by the following means:

- The policy wizard operates as the interface to the customer, allowing her/him to specify business-level configuration changes.
- Information from the wizard is transformed into the business policy language (by a language recognizer/parser).
- All steps of the policy continuum are realized by cascaded XML transformations using the apache cocoon framework.
- The instance configuration and the final testbed are realized with an XML file that represents the configuration information. This XML file will be further transformed by the OPNET interface in order to simulate its operation.

Package				AppSvc and Grade				Configuration		
Gold	==	2 Mbps	=>	VoIP (1)	VoD (2)	Web (3)	=>	VoIP (50%)	VoD (30%)	Web (20%)
			=>	VoIP (1)	VoD (3)	We (2)	=>	VoIP (50%)	VoD (20%)	Web (30%)
			=>	VoIP (2)	VoD (1)	Web (3)	=>	VoIP (30%)	VoD (50%)	Web (20%)
			=>	VoIP (2)	VoD (3)	Web (1)	=>	VoIP (30%)	VoD (20%)	Web (50%)
			=>	VoIP (3)	VoD (1)	Web (2)	=>	VoIP (20%)	VoD (50%)	Web (30%)
			=>	VoIP (3)	VoD (2)	Web (1)	=>	VoIP (20%)	VoD (30%)	Web (50%)
Silver	==	1.5 Mbps	=>	VoIP (1)	VoD (2)	Web (3)	=>	VoIP (50%)	VoD (30%)	Web (20%)
			=>	VoIP (1)	VoD (3)	Web (2)	=>	VoIP (50%)	VoD (20%)	Web (30%)
			=>	VoIP (2)	VoD (1)	Web (3)	=>	VoIP (30%)	VoD (50%)	Web (20%)
			=>	VoIP (2)	VoD (3)	Web (1)	=>	VoIP (30%)	VoD (20%)	Web (50%)
			=>	VoIP (3)	VoD (1)	Web (2)	=>	VoIP (20%)	VoD (50%)	Web (30%)
			=>	VoIP (3)	VoD (2)	Web (1)	=>	VoIP (20%)	VoD (30%)	Web (50%)
Bronze	==	1 Mbps	=>	VoIP (1)	VoD (2)	Web (3)	=>	VoIP (50%)	VoD (30%)	Web (20%)
			=>	VoIP (1)	VoD (3)	Web (2)	=>	VoIP (50%)	VoD (20%)	Web (30%)
			=>	VoIP (2)	VoD (1)	Web (3)	=>	VoIP (30%)	VoD (50%)	Web (20%)
			=>	VoIP (2)	VoD (3)	Web (1)	=>	VoIP (30%)	VoD (20%)	Web (50%)
			=>	VoIP (3)	VoD (1)	Web (2)	=>	VoIP (20%)	VoD (50%)	Web (30%)
			=>	VoIP (3)	VoD (2)	Web (1)	=>	VoIP (20%)	VoD (30%)	Web (50%)

Table 1. Content service specification and business to system mapping

The assumptions made for this mapping process includes (i) the mapping process is based on a ratio that is fixed and does not change between different packages and (ii) the ratio between the different packages for the three types of grades are identical in order to minimize complexity.

The basis of mapping between the levels is based on the translation table shown in Table 1. The table represents the translation from the business view to the system view in the policy continuum. It shows the possible combinations between the different objects that a customer can select from the GUI. For example, if John selects Gold class package with VoIP as Grade 1, VoD as Grade 2, and web as Grade 3 will result in the VoIP service throughput limited to 50%, VoD limited to 30% and Web limited to 20% of the total amount of bandwidth allocated to John, which is 2 Mbps.

6.2 Simulation Environment

The simulation environment is shown in Figure 3. The developed network uses Diffserv to manage Quality of Service. Marking and policing of traffic is performed at the edge routers. Each service flow per end user is identified at the edge router by an Access Control List made up of Source and Destination IP addressing and port numbering, plus a DSCP (DiffServ Code Point) to identify the required traffic conditioning.

The service packets are marked with the appropriate DSCP code, to identify Per Hop Behavior (PHB) routines within the core routers. Each service flow is subject to a rate limiting policer policy. Once traffic is identified to be within a particular flow, the appropriate policer policy limits the throughput of that traffic to match the relevant high-level policy settings the end user has set for that particular service. For each user there are access control lists defined on the edge routers to identify each flow of traffic for each service.

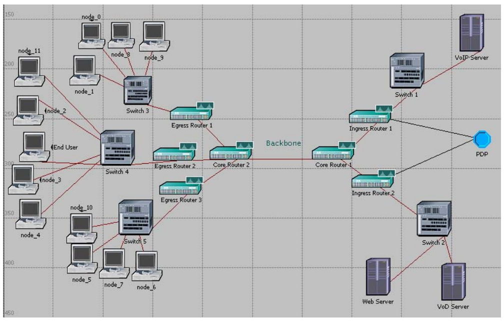

Fig. 3. Simulation Environment

The rate limit is calculated by mapping values within the deployed policy to throughput limits for each service flow of a particular user. For example, John is subscribed to a Gold service package. His VoIP service is assigned Grade 1, VoD service is assigned Grade 2, and Web service is assigned Grade 3. This will map to 2 Mbps (Gold) bandwidth being divided into three portions namely: Grade 1 VoIP getting 1 Mbps (50% of 2 Mbps), Grade 2 VoD getting 600 Kbps (30% of 2 Mbps), and Grade 3 Web getting 400 Kbps (20% of 6Mbps).

6.3 Simulation

Once the policy has been deployed, a PDP process will invoke the required changes on edge routers within the network. Customer Flows are identified by access control lists on the edge routers. Policer policies for identified flows are updated with the new rate limit values, calculated from the high-level policies and translation algorithms. The end user begins to use the service around 1m 50s into the simulation. The figures 4 to 7 show the users' throughput in bps for each service after 2m 20s:

- Figure 4: John's Package is set to Gold; thus, his maximum bandwidth available is 2Mbps. His VoIP service is set up as grade 1. Grade 1 is 50% of the overall package. This figure depicts an average throughput of 1Mbps, showing that the network is configured properly for John's VoIP service.
- Figure 5: John's Video on Demand service is set to Grade 2. As grade 2 is 30% of the overall package bandwidth, John should receive a max throughput of 600 Kbps.
- Figure 6: John's Web service is set to Grade 3. As grade 3 is 20% of the overall package bandwidth, John should be receiving a maximum throughput of 400 Kbps. Figure 6 shows that the average throughput of the Web service is less than 400 Kbps, thus showing that this grade is suitable the current generation of web traffic.
- Figure 7: The total amount of bandwidth used by the user between the three services averages just under the 2Mbps mark.

Once a policy is modified and deployed (PM – Policy Modification), the PDP within the OPNET simulated network runs through the same process as the initial phase, updating any QoS parameters that need to be set. The end user deploys a modified policy at around 2m 20s into the simulation. The figures 8 to 11 depict the users' throughput (bps) for each service after the modified policy deployment at 1m 20s to the end of the simulation:

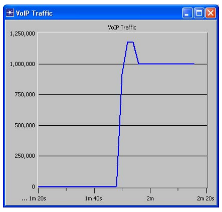

Fig. 4. VoIP Traffic

Fig. 5. VoD Traffic

Fig. 6. Web Traffic

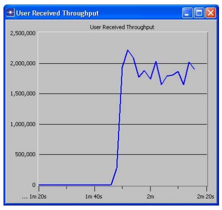

Fig. 7. User Throughput

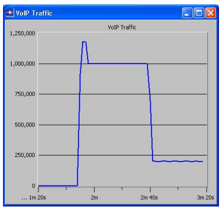

Fig. 8. VoIP Traffic

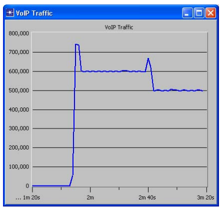

Fig. 9. VoD Traffic

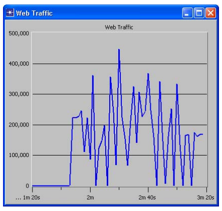

Fig. 10. Web Traffic

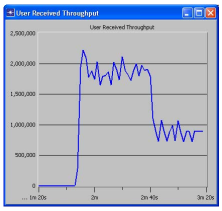

Fig. 11. User Throughput

- Figure 11: John modified his package to Bronze within the policy editor. We know from Table 1 that the maximum throughput of Bronze Package is 1Mbps. We can see that this policy has been invoked within the network, as the average throughput received by John drops from 2Mbps to around 1Mbps.
- Figure 8: John modified his VoIP service to Grade 3. As grade 3 is 20% of total package throughput. VoIP service is reduced to 20% of 1Mbps, which is 200 Kbps.
- Figure 9: John modifies VoD traffic from Grade 2, to Grade 1. Traffic throughput decreases from 30% of Gold (600 Kbps) to 50% of Bronze (500 Kbps).
- Figure 10: John modifies Web traffic from Grade 3 to Grade 2. This means that allowable traffic throughput for Johns Web service is reduced from (Gold @ 20% = 400 Kbps) to (Bronze @ 30% = 300 Kbps). The graph shows that when the modified policy was deployed, average web traffic was slightly reduced.

7 Conclusion

This paper has presented an implementation and simulation for a novel policy-based architecture for end-to-end network management. The architecture and the implementation are based on a set of key principles, which include: (i) the use of a policy continuum to enable different constituencies to express their needs and define goals and/or rules that govern the system, (ii) the use of an object-oriented information model (DEN-ng) that provides an extensible representation of information of interest; key is its ability to work using patterns and abstractions (e.g., capabilities, constraints and context) to manage the inherent complexity of a network (only partly shown in this paper), (iii) the use of a set of ontologies that *augment* the knowledge contained in the information model to better define semantics and meaning of data and events (background processes), (iv) the use of an extensible policy model that is independent of content to express goals and rules for *all* constituencies, (v) The focus of modeling *behavior*, not just *current state*, to control the life cycle of a managed element (future extension for the policy language). Future work will concentrate on stressing the above architecture, as well as incorporating additional reasoning capabilities (in the form of reinforcement and concept learning, as well as abductive and inductive reasoning algorithms) to give the autonomic system greater *understanding* of its environment. We also intend to move to a test bed environment where we can measure performance and scalability. As this future work grows, we expect the role of policy to be even more pronounced.

References

1. Davy, S., Barrett, K., Balasubramaniam, S., van der Meer, S., Jennings, B., Strassner, J.: Policy-Based Architecture to Enable Autonomic Communications - A Position Paper. IEEE CCNC 2006, Special Session on Autonomic Communication, Las Vegas, Nevada, Jan 7-10, (2006)
2. Strassner, J.: Policy-based Network Management: Solutions for the Next Generation. Morgan-Kaufman Publishers. ISBN 1-55860-859-1, Sep (2003)
3. TMF: TMF 053: The NGOSS™ Technology Neutral Architecture. plus Addenda, 2000-(2005)
4. Open Distributed Processing Reference Model – Foundations, ISO/IEC 10746-2, (1996)
5. TeleManagement Forum: GB927: The NGOSS™ Lifecycle and Methodology. Nov (2004)
6. DEN-ng forms the foundation of the TeleManagement Forum's Shared Information and Data Model, and is partially incorporated in the following specifications: GB922 main, GB922 Addenda 1A, 1BI, 1P, 1-POL, 1R, 4SO, 4S-QoS, 5RO, 5PR, and 5LR.
7. Strassner, J. How Policy Empowers Business-Driven Device Management. in proceedings of the Third International Workshop on Policies for Distributed Systems and Networks (Policy' 02), Monterey, California, USA, June (2002).
8. See, for example, M. Fowler: Analysis Patterns – Reusable Object Models.
9. In particular, variations of the role object pattern are used – please see http://www.riehle.org/computer-scienceresearch/1997/plop-1997-role-object.pdf (2006)
10. Strassner, J. Autonomic Networking – Theory and Practice. IM 2005 Tutorial, May (2005)
11. van der Meer, S., O'Sullivan, D., Lewis, D., Agoulmine, N. Ontology Based Policy Mobility for Pervasive Computing. 9th IFIP/IEEE International Symposium on Integrated Network Management, IM 2005 Nice, France, May 15-19, (2005)
12. Kephart, J.O. and Chess, D.M., The Vision of Autonomic Computing. Jan 2003, www.research.ibm.com/autonomic/research/papers/AC_Vision_Computer_Jan_2003.pdf (2006)
13. Motorola, Seamless Mobility page: http://www.motorola.com/content/0,,2364-4393,00 (2006)