Autonomic Communications: Exploiting Advanced and Game Theoretical Techniques for RAT Selection and Protocol Reconfiguration

Eleni Patouni[1], Sophie Gault[2], Markus Muck[2], Nancy Alonistioti[1], and Konstantina Kominaki[1]

[1]Communication Networks Laboratory, Department of Informatics & Telecommunications, University of Athens, Athens, Greece
{elenip, nancy, kominaki}@di.uoa.gr
[2]Motorola Labs, Paris, France,
{sophie.gault, markus.muck}@motorola.com

Abstract. The Autonomic Communications concept emerges as one of the most promising solutions for future heterogeneous systems networking. This notion implies the introduction of advanced mechanisms for autonomic decision making and self-configuration. To this end, this paper proposes an integrated framework that facilitates autonomic features to capture the needs for RAT selection and device reconfiguration in a Composite Radio Environment. Specifically, a game theoretical approach targeted to the definition of appropriate policies for distributed equipment elements is presented. Thus, the user terminals are able to exploit context information in order to i) identify an optimum trade-off for (multiple) Radio Access Technology (RAT) selection and ii) adapt the protocol stack and respective protocol functionality using a proposed component based framework for transparent protocol component replacement. Simulation and performance results finally show that the proposed mechanisms lead to efficient resource management, minimizing the complexity on the network and terminal side as well as keeping the required signaling overhead as low as possible.

Keywords: autonomic networking, cognitive networks, reconfiguration

1 Introduction

Future beyond 3rd Generation (B3G) systems are expected to exploit the full benefits of the diversity within the radio eco-space, composed of wide range of systems such as cellular, fixed, wireless local area and broadcast. In this framework, it is important to provide suitable means on the network and terminal side serving as an enabler for this vision. Such vision is captured by the notion of autonomic communications which provides the grounds for the deployment of advanced concepts, including a device agnostic and protocol independent approach for an hierarchy of systems with self-managing, self-configuring and self-governance features. Beyond the conceptual merits of such an approach, the following key issues need to be addressed in a practical context: i) Manage the complexity on the network and user terminal side and provide policy

communication means, ii) Minimize required signaling overhead and iii) Provide means for the device dynamic adaptation following the decision for RAT selection.
The first of these items typically motivates a distributed system concept as analyzed in [1] in the context of autonomic communications where self-managing devices with behavior controlled by policies are introduced. Furthermore, we assume the introduction of a suitable cognitive channel which covers, besides policy related information, future context data helping the devices to perform decisions. Item ii) relates to the policies themselves, leading to the observation that *simple, global policies* (applicable to all users) should be preferred to user specific rules in order to assure a minimum signaling overhead. In addition item iii) is related to the introduction of a framework incorporating the necessary mechanisms that enable the dynamic adaptation/reconfiguration of the protocol stack.
In the context of this paper, all these principles are highlighted; the rest of this contribution is organized as follows: Section 2 defines the general study framework portraying the problem that is examined in this contribution. A RAT selection analysis for a simple two-user context, based on game theoretic tools is presented in section 3. A framework for the dynamic protocol reconfiguration over heterogeneous RATs is analyzed in section 4. Finally, related work and conclusion remarks as well as directions for future research are highlighted in section 5 and 6 respectively.

2 Problem Statement

In this analysis, a composite radio environment, in terms of a distributed network of heterogeneous Radio Access Technologies (RATs), is considered, as illustrated below (Fig. 1). A multitude of users is assumed to compete for access to one or several RATs and one or several distinct communication channels (in terms of spectrum usage) in parallel. An efficient operation requires suitable RAT/channel selection algorithms: in heterogeneous and reconfigurable wireless systems, terminals and network equipments should incorporate enhanced capabilities for adapting to the drastically changing environment.
Towards this direction, this paper analyzes an integrated framework that facilitates autonomic features to capture the needs for RAT selection and device reconfiguration in a Composite Radio Environment. At first, the process of selecting a RAT targeted to the optimum adaptation of users is addressed. Following the RAT selection, the dynamic device adaptation to the new RAT should take place, to cope with application and QoS requirements. For example, after a change in the RAT, an update in a protocol component/codec may be triggered (either network initiated or device initiated) for various reasons: i) to compensate for QoS degradation, ii) to provide a protocol patch update to fix a software bug iii) to provide a new version of an existing component with enhanced capabilities. In this sense, a generic framework is provided that handles the necessary mechanisms for downloading, installation and on-the-fly activation of missing protocol-related RAT components. The following subsections highlight the focus and design assumptions in each of the previously mentioned reconfiguration phases.

2.1 RAT Selection Context

The RAT selection phase addresses an efficient attribution of corresponding resources to a specific user (different RATs such as WiMAX, WiFi (IEEE802.11a/b/g/n, etc.), 3GPP, DVB-T or DAB, different bands, etc.) in a distributed system, minimizing the required complexity in the network and user side as well as the signaling overhead. The focus is laid on techniques that are fully compatible with legacy technologies; the proposed approaches are also applicable to future air-interfaces, following the trend for the deployment of a (physical or virtual) cognitive channel as a single new element to be exploited for finding optimum resource usage strategy. These approaches are meant to be transparent to the physical user – any reconfiguration process is handled automatically by the equipment devices.
In addition, each terminal/user can apply several strategies in order to get the best service requested by the user. Multi-mode and reconfigurable terminals have the capability to connect simultaneously to several wireless network resources and also to reconfigure themselves in order to connect to new radio access technologies available in a cell. Given that multi-mode and reconfigurable network equipments inherently provide enhanced capabilities (by either dynamically adapting a specific radio access resource, or by reconfiguring some nodes to dynamically provide higher system capacity, depending on demands in a given area), consequently, the terminals should automatically adapt to the new scenario.

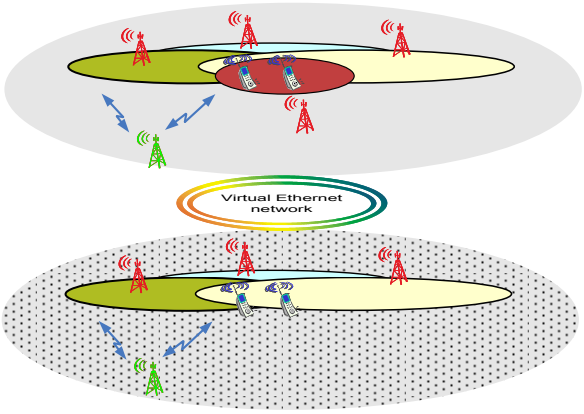

Fig. 1: A distributed network approach in a multicell context with different Cellular Access Points (CAPs).

Moreover, it is assumed that the system is organized in an entirely distributed way: the network propagates "policies" (e.g., via the Cognitive Channel) which define generic behavioral rules to be applied by any network and user equipment. Consequently, the network/user equipment is NOT parameterized by any central controller, but adapts autonomously (typically applying "Autonomic Networking" principles) to the constantly changing environment. This finally leads to a distributed optimization of the resource use. In the same example, a possible environmental change triggering user adaptation is

illustrated: a RAT terminates its services and the remaining resources (RATs, bands, etc.) must thus be split among the active users.
The problem addressed within this paper concerns the optimum adaptation of users to a changing context/environment using autonomic networking and policy-based self-governance principles. A mechanism is proposed that enables users to adapt their resource use autonomously (applying autonomic networking approaches and relying on policy based self-management), such that a suitable compromise is found that is near-optimum from the perspective of a specific user ("*get maximum data rate, even if I penalize other users*") and from the network perspective ("*maximize total network throughput and split resources fairly among all users*").

2.2 Protocol Reconfiguration

Following the RAT selection, the protocol reconfiguration phase is aimed to address generic mechanisms for the deployment of transparent plug-in of protocol components in equipments. The presented solution is aligned with a set of assumptions regarding the design aspects of the proposed architecture and mechanisms:

- a protocol stack is composed of discrete protocol layers. The communication between them is established either using standard defined interfaces, i.e. Service Access Points (SAPs) or queue-based communication schemes. This design also facilitates the maintenance of cross layer optimization issues in the protocol stack. In addition, this design provides the capability of specifying a protocol stack according to application needs, QoS requirements as well as the specific RATs.
- A protocol layer is composed of protocol components. Each protocol component may specify specific protocol functionality (i.e, if we consider a TCP protocol, a TCP component may realize the congestion control algorithms) or a combination of different functionalities (i.e, a TCP component that realizes both congestion control and flow control algorithms).

The introduced framework based on the above considerations is aimed to cope with the following protocol reconfiguration aspects: the dynamic binding of component services into a fully fledged protocol service as well as the runtime replacement of protocol functionality. Specifically, this solution extends the typical Manager-centric architectures for the establishment of component bindings introducing a distributed model. Such model apportions the above mentioned functionality to the protocol components. The latter is based on a semantic-layer of information which describes static characteristics of the components as well as dynamic characteristics to capture the environment configuration.
The above analyzed mechanisms are incorporated into a generic management and control architecture enabling dynamic protocol reconfiguration via self-configuring protocol components (Fig. 2). In particular, the following key elements are identified:

- The Download Manager module which caters for the software download in the system, as well as for authorization procedures and integrity checks.
- The Installation Manager, which is responsible for post-download steps as well as the software installation to the system.

- The Decision Manager module which specifies concrete decision concerning reconfiguration actions, based upon a set of policy rules and contextual information. In the scope of this paper, such module is responsible for the protocol stack configuration, in terms of specifying the different protocol layers and components to be used, as well as for triggering a protocol stack update.
- The Autonomic Manager module, which is responsible for the overall monitoring and control of the software operation, i.e., it instantiates the various components/triggers the component replacement process.

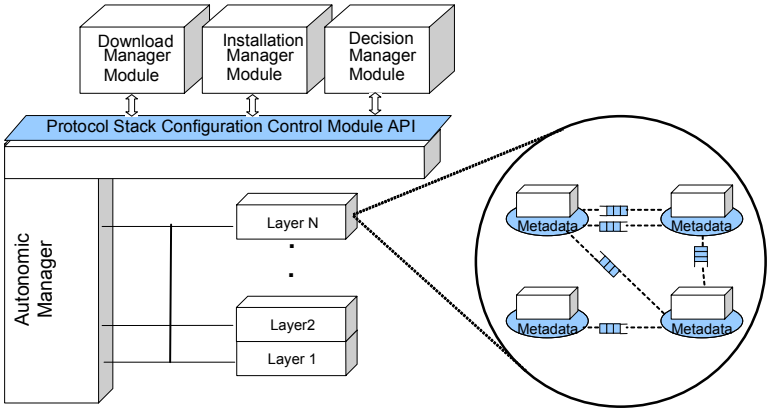

Fig. 2: A Management and Control Architecture Enabling Self-Configuring Protocols

3 Analysis of Game Theory based RAT Selection in a Simple Two-User Context

Considering a simple two-user scenario, this section illustrates the application of a game-theoretic analysis [2] in order to derive suitable policy rules directing the user behavior. It is shown that global policies, applicable to all users, reduce the RAT selection convergence time considerably. Moreover, a *global* policy assures a minimum signaling overhead, since the user terminals are not addressed independently as it is the case of a centralized approach. The main aspects presented below can be extended to more complex scenarios consisting of a multitude of heterogeneous RATs and a multitude of users at the cost of an increased complexity for the RAT selection and search for suitable policies. This generalization, however, is out of the scope of this paper and will be discussed in future contributions.

3.1 Scenario Definition

The following scenario is considered in this analysis: An operator controls four IEEE802.11n Access Points (APs), each operating in a distinct 20MHz band and at a

distinct carrier frequency. There are two Mobile Terminals (MTs) communicating over 1, 2, 3 or all of the available bands. The operator decides to switch off one AP, and indicates this information by propagating a corresponding message to the MTs. The MTs then need to redefine their spectrum / AP use autonomously. Each MT has the choice among seven possible spectrum allocation strategies denoted from S_1 to S_7:

1) S_1: use band #1;
2) S_2: use band #2;
3) S_3: use band #3;
4) S_4: use bands #1 and #2;
5) S_5: use bands #2 and #3;
6) S_6: use bands #1 and #3;
7) S_7: use bands #1, #2 and #3.

A simplified throughput computation model is used, assuming a throughput per band (or channel) equal to D bit/s. When a given channel is reserved to only one MT, the total throughput D is available for the MT. In case it is split among two MTs, the total throughput decreases due to collisions: D' = D*d where 0<d<1 is a kind of penalty factor, and each MT gets a throughput of D'/2 = D*d/2 with 0<d<1. In the examples below, we choose "d=0.9" for illustration purposes. The issue addressed is to find the best combination of strategies for both MTs such that maximal throughput is achieved for both.

3.2 Performance Analysis

The analysis is carried through the 2D game table presented below; the rows and the columns correspond to the strategies of MT1 and MT2 respectively. In addition, the table elements correspond to pairs of throughput values (MT1 throughput, MT2 throughput), obtained when MT1 and MT2 are using a given combination of strategies.

Table 1. Overall game table (1st column: User 1 strategies, 1st line: User 2 strategies)

	S1	S2	S3	S4	S5	S6	S7
S1	(0.45, 0.45)	(1,1)	(1,1)	(0.45, 1.45)	(1,2)	(0.45, 1.45)	(0.45, 2.45)
S2	(1,1)	(0.45, 0.45)	(1,1)	(0.45, 1.45)	(0.45, 1.45)	(1,2)	(0.45, 2.45)
S3	(1,1)	(1,1)	(0.45, 0.45)	(1, 2)	(0.45, 1.45)	(0.45, 1.45)	(0.45, 2.45)
S4	(1.45, 0.45)	(1.45, 0.45)	(2, 1)	(0.9, 0.9)	(1.45, 1.45)	(1.45, 1.45)	(0.9, 1.9)
S5	(2, 1)	(1.45, 0.45)	(1.45, 0.45)	(1.45, 1.45)	(0.9, 0.9)	(1.45, 1.45)	(0.9, 1.9)
S6	(1.45, 0.45)	(2, 1)	(1.45, 0.45)	(1.45, 1.45)	(1.45, 1.45)	(0.9, 0.9)	(0.9, 1.9)
S7	(2.45, 0.45)	(2.45, 0.45)	(2.45, 0.45)	(1.9, 0.9)	(1.9, 0.9)	(1.9, 0.9)	(1.35, 1.35)

To give an example: in the first cell on the upper left corner, user 1 chooses "strategy S1" and user 2 equally chooses "strategy S1"; in conclusion, both users are sharing a single channel where collisions may occur and the throughput per user is ½*d = 0.45. After analyzing the game table, the existence of a unique Nash equilibrium when both users choose strategy S_7 (red cell) is apparent. In fact, this forms a stable state which no user would find it interesting to deviate from. However, it is not Pareto efficient since better couples of throughputs are obtained with other combinations (yellow cells).

If a given user follows the simple rule of always targeting the maximal throughput, no matter what are the consequences on the other user, he will choose strategy S_7 and reach

the states corresponding to the green cells; this situation results in an operating point which is suboptimal, in spite of being a Nash equilibrium.
For instance, suppose users play in turn, as represented with the orange arrows in Table 1. If users are in an initial state such that both users pick up strategy S_1 (the normalized throughput they both achieve equals to 0.45) and if user 2 is the first to play, he will try to achieve the maximal throughput and therefore chooses strategy S_5 (he achieves throughput equal to 2 instead of 0.45). Then given the new strategy of user 2, user 1 will try to maximize its throughput in turn and chooses strategy S_7 (the normalized throughput he achieves equals to 1.9 instead of 1). Finally, user 2 responds by selecting strategy S_7 and the equilibrium is reached, since both users achieve throughput equal to 1.35 and no one can improve its throughput by modifying only its own strategy.

3.3 Derivation of suitable policies

The idea is to establish controlled competition so as to get the fairest split of resources and reach the states corresponding to the yellow cells. This is achieved by the use of simple policies propagated by the operator, e.g. "do not use strategy S_7".
The operating point search is made on the following suitable where strategy S_7 has been removed for both users. If the game is played based on this table (Table 2) and users still follow the simple rule of always seeking for the maximal throughput (no matter what are the consequences on the other user), the states corresponding to the yellow cells will systematically be reached.
For example, suppose again that users play in turn, following the orange arrows represented on Table 2. If users are in the same initial state as previously (both users select strategy S_1 and achieve normalized throughput equal to 0.45) and if user 2 is the first to play, he will choose strategy S_5 (he achieves a maximal throughput equal to 2 instead of 0.45). Then given the new strategy of user 2, user 1 will try to maximize its throughput in turn and chooses indifferently strategy S_4 or S_6 to get 1.45 instead of 1. Since the resulting throughput of user 2 is also maximized (he cannot achieve better throughput than 1.45), this new configuration is an equilibrium, which is clearly more efficient than the previous equilibrium where users both picked up strategy S_7.

Table 2. Modified game table (1st column: User 1 strategies, 1st line: User 2 strategies)

	S1	S2	S3	S4	S5	S6
S1	**(0.45, 0.45)**	**(1,1)**	**(1,1)**	**(0.45, 1.45)**	**(1,2)**	**(0.45, 1.45)**
S2	**(1,1)**	**(0.45, 0.45)**	**(1,1)**	**(0.45, 1.45)**	**(0.45, 1.45)**	**(1,2)**
S3	**(1,1)**	**(1,1)**	**(0.45, 0.45)**	**(1, 2)**	**(0.45, 1.45)**	**(0.45, 1.45)**
S4	**(1.45, 0.45)**	**(1.45, 0.45)**	**(2, 1)**	**(0.9, 0.9)**	**(1.45, 1.45)**	**(1.45, 1.45)**
S5	**(2, 1)**	**(1.45, 0.45)**	**(1.45, 0.45)**	**(1.45, 1.45)**	**(0.9, 0.9)**	**(1.45, 1.45)**
S6	**(1.45, 0.45)**	**(2, 1)**	**(1.45, 0.45)**	**(1.45, 1.45)**	**(1.45, 1.45)**	**(0.9, 0.9)**

At this point it should be pointed out that the use of a very fundamental policy rule expressing a constraint on the strategy selection ("do not use strategy S_7"), makes it possible to avoid sub-optimal Nash equilibrium.

3.4 Analysis of impact of policy introduction

In the following figures, the game evolution in four different scenarios is illustrated:

- Scenario 1: users play simultaneously and do not consider policy rules;
- Scenario 2: users play simultaneously and respect the previously mentioned policy rule;
- Scenario 3: users play in turn (i.e. one after each other) and do not consider policy rule;
- Scenario 4: users play in turn and respect the previously mentioned policy rule.

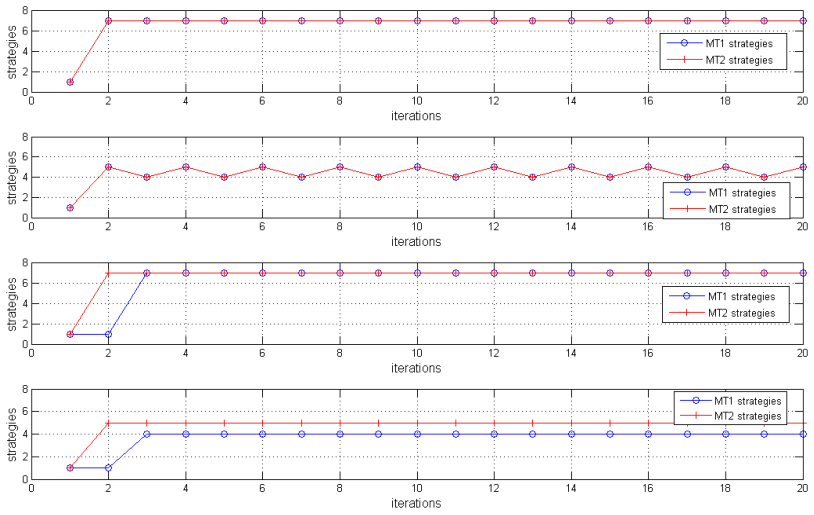

Fig. 3: Convergence of strategies.

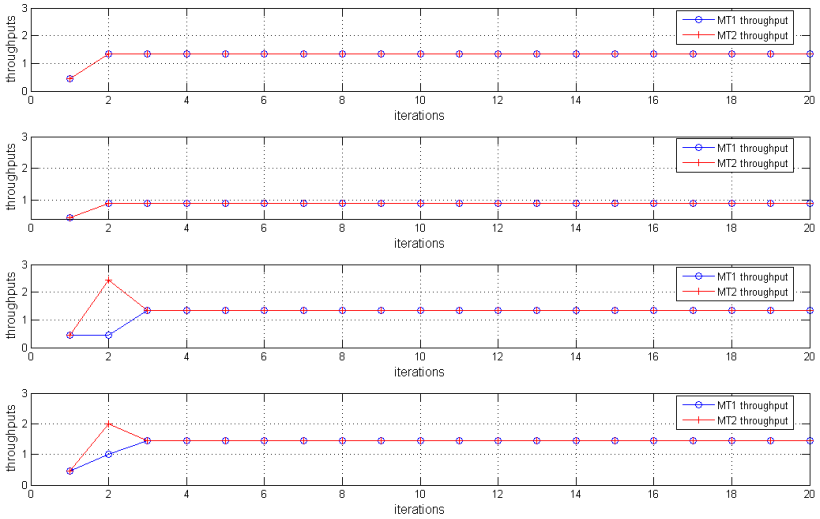

Fig. 4: Data Rates

The first series of four graphs (Fig. 3) represents the evolution of the users' choice of strategy for the set of four scenarios, while the second (Fig. 4) represents the users' throughput evolution for the same four scenarios. It should be noted that a configuration where users play in turn converge to the equilibrium faster than when users play simultaneously. Moreover, the curves confirm that the use of an appropriate policy rule helps the system to converge towards an absolutely efficient equilibrium.

4 A Framework for Dynamic Protocol Reconfiguration over Heterogeneous RATs

This section highlights the procedure of protocol reconfiguration, describing the proposed framework and mechanisms through a reconfiguration scenario over heterogeneous RATs.

4.1 Protocol Reconfiguration

The section analyzes the fundamental phases of the protocol reconfiguration procedure, Firstly, the mechanisms involved in the protocol stack bootstrap are considered. In addition, a simple case that the decision mechanisms embedded in the terminal dictate that a protocol reconfiguration should take place is presented. This decision concerns the downloading, installation and on-the-fly activation of a protocol component.

Control Signaling for Protocol Stack Bootstrap
During the protocol stack bootstrap, a configuration of the protocol stack is selected by the autonomic decision making functionality; such configuration specifies the protocol layers that should form the protocol stack as well as the components that should be used within each protocol layer. Thereafter, the Decision Manager informs the Autonomic Manager about the protocol layer configuration and the protocol component configuration. After acknowledging the reception of this information, the Autonomic Manager proceeds with the instantiation of the protocol components selected for each protocol. Considering that the binding between the protocol layers is realized via the standard defined Service Access Points (SAPs), the focus is on the procedures related to the protocol component binding and replacement. Therefore, the reconfiguration signalling depicted in Figure 1 illustrates only the instantiation of two protocol components that form a protocol layer (Component TestA and Component TestB) [16].

Semantic-Based Dynamic Binding of Protocol Components
Next, the semantic-based dynamic binding of protocol components is performed. Specifically, the components evaluate the dynamic characteristics of their metadata and identify the components there are composed with. Finally they establish the bindings to the components they are composed with.
The details of this procedure are illustrated with an example; a protocol layer comprised of two autonomic protocol components (CompA and CompB) is considered and the

dynamic binding of these components is analyzed. In this case study a unidirectional communication pattern between the protocol components is assumed, in the sense that CompA sends data to CompB. The XML representation for the metadata profiles of CompA and CompB is illustrated in Fig. 5 (a) and (b) respectively.

The metadata profiles include static characteristics specifying the component identification (name, version, path to source code), whereas the dynamic characteristics depict their current configuration in the system and are dynamically updated according to the different protocol stacks stratification (dynamic characteristics include arrays for input and output components and their static characteristics). For example, as depicted in Fig. 5 (a), the protocol component CompA does not provide any input interface, whereas it provides an output interface to CompB.

Based on the interpretation of the metadata profile, the component composition is realized. During this phase at first, CompA checks the input array in its metadata, which does not include any components. Thereafter, it checks its output array, which includes CompB. Next, it should verify the composition between CompA and CompB by checking that the input array in CompB metadata includes CompA. The same procedure is applied by CompB. At this point it should be clarified that this procedure also applies for all the protocol components regardless of the number of components they are composed with.

```
<?xml version="1.0"
encoding="ISO-8859-1" ?>

<component>
    <id> CompA</id>
    <version>Version 1</version>
    <path>/CompA</path>

    <inputs>
            <input No="0" />
    </inputs>

    <outputs>
        <output No="1">
        <id>CompB</id>
        <version>Version 1</version>
        <path>/CompB</path>
        </output>
    </outputs>

</component>
```

(a)

```
<?xml version="1.0" encoding="ISO-
8859-1" ?>

<component>
    <id> CompB</id>
    <version>Version 1</version>
    <path>/CompB</path>

    <inputs>
        <input No="1" />
        <id>CompA</id>
        <version>Version 1</version>
         <path>/CompA</path>
    </inputs>

    <outputs>
        <output No="0" />
    </outputs>

</component>
```

(b)

Fig. 5: Metadata profiles for protocol components CompA and CompB

After the validation and verification of the composition, the component communication establishment is realized. Specifically, each component establishes a communication link for each component it is composed with by creating a FIFO queue. The latter is realized with the use of a unique key. Such key is generated by a conversion function that produces a global unique output based only on a unique parameter, the concatenated String of the ID of the specified component and the ID of the component it is composed

with. In addition, it should be noted that with the use of this ID, the specified component creates or accesses a FIFO queue, depending if it already exists in the system. The composition verification and establishment procedures are repeated for all the protocol components that are bound to the specified component.

Control Signaling for Protocol Reconfiguration

This phase comprises the necessary steps for the protocol reconfiguration process. Such procedure may be triggered by a change in the environment (new RAT/cellular system, handover procedures), or the QoS requirements posed by applications or user preferences. Next, the decision module specifies the new protocol stack configuration. Based on contextual information about the protocol stack, it identifies whether the specified components exist in the system. In case of missing components, as in the scenario in Fig. 6 the Decision Manager should select the appropriate protocol component from the repository available in the network. Thereafter, the Autonomic Manager requests the Download Manager to perform the component downloading. After the successful realization of this procedure, the component installation in the system is performed by the Installation Manager. Finally, the Decision Manager informs the Autonomic Manager about the new system configuration.

Regarding the selection of the most appropriate protocol component from the network, a theoretical approach is introduced. At first the interface compatibility between the stationary and available components is checked; then the compatible components are compared in order to select the "best", based on the QoS it may provide and the delay introduced for its downloading in the system.

This analysis is based on a modular system with the following formulation [8]. Let us assume there are i protocol components, with input-output interfaces. In this sense, compatible components have the same interfaces with the same components. The methodology to find the compatible components is based on the Graph Theory. Every group of compatible components belongs to a module s_i . Further, $I = \{s_1, s_2, s_3\}$ is the set of these modules, s_i module comprises all the compatible components. Each module is associated with: 1) a vector of output variables $P_{s_i} \in P_i$, which describes the output interfaces of the protocol components. Each module has an initial estimate P_i^0 of its vector, which defines the output interface of the initial protocol components connection. 2) a set of input interfaces vectors from it's neighbour modules $P'_{s_i} \in P'_i$ and N_i which specifies the modules with which modules belonging to I need to interact. 3) An objective function Q_i: that is a measure of how well the output vector P_i of the specified module satisfies the task of the module, given its inputs from all its surrounding modules:

$Q_{s_i}(P_{s1}, P_{s2}, P_{s3}) = Q(P_{s_i})$, $Q_{s_i}(P_{s1}, P_{s2}, P_{s3}) = q_i(P_i) + \frac{1}{\lambda_i g_i(P_i)}$, where

q_i is the quality of service function of each protocol component, g_i is the delay function of each protocol component and λ_i depends on the number of the replaced components.
The decision procedure, in terms of replacing a compatible component with the one can be modelled as a game. Specifically, a game with $I = \{s_1, s_2, s_3\}$ players is considered; in our example the game has three players s_1 , s_2 and s_3. Each player is associated with a decision vector (An individual decision vector P_{s_1}, P_{s_2} and P_{s_3}) and a payoff function. During the course of the game, which is a sequence of stages and moves, each player chooses a specific decision vector. The payoff function Q evaluates the performance of the player based on its decision P_{s_i} and the decisions from the other players that influence the decision of player i. The I player game starts; each player wants to find the decision that optimises its payoff. In particular, a solution requires: 1) a concept of what is meant to be optimality 2) decision making models that allow for the computation of this equilibrium. There exist no cooperation among the players and each player makes its own decision independently. The game theoretic integration framework is a particular solution to the integration problem where the decision making model is defined in the context of no cooperative games.

Dynamic Replacement of Protocol Components
After evaluating the new protocol configuration, the Autonomic Manager should perform the replacement of the old protocol component with the new one. At first, the Autonomic Manager pauses the functionality of the component under replacement and retrieves its execution state (Fig. 6). Next, the Autonomic Manager instantiates the new component and dispatches to it the retrieved state information. Based on the acquired state information and its metadata, the new component realizes its dynamic composition with existing software components (by accessing the FIFO queues that correspond to existing communication links). The above analyzed on the fly replacement process also allows the reliable operation of the software under configuration, since it applies state management models to ensure the transparent switching from the old to the new component.

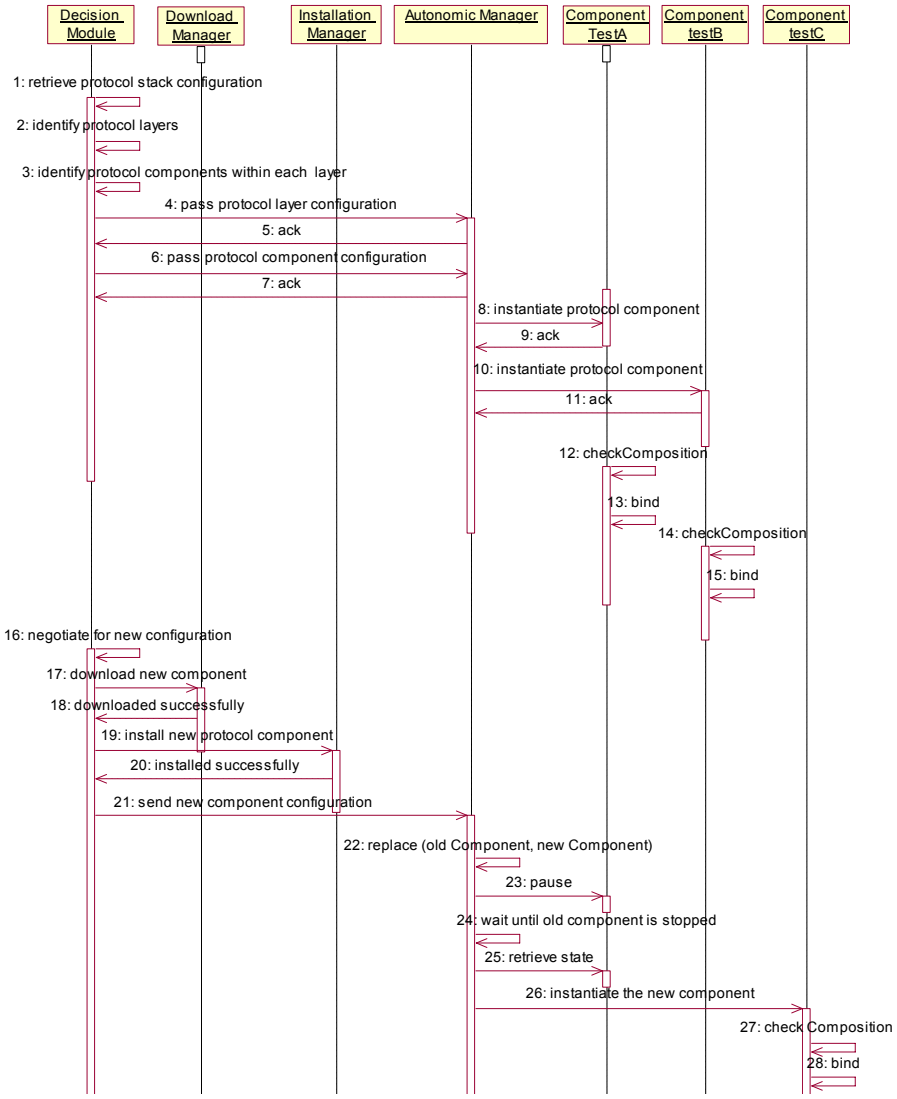

Fig. 6: Signaling for Protocol Stack Bootstrap and Protocol Self-Configuration

4.2 Validation and Performance Assessment

Targeted to verify and validate the introduced framework, a proof-of-concept prototype is implemented, concerning the dynamic binding and reconfiguration capabilities of test protocol components. Specifically, this prototype concerns the deployment of the following functionality: a) the Autonomic Manager, b) a test protocol which comprises of two protocol components (CompA and CompB presented in the example) with

unidirectional communication as well as the component metadata and c) the presented reconfiguration framework upon its application in the test protocol.

Exploiting the mechanisms and procedures presented in the previous sections, the implemented Autonomic Manager initiates these two protocol components; thereafter the protocol components discover and access their metadata files and automatically establish their bindings. In addition, a simple reconfiguration scenario was realized, in terms of replacing CompB with another component with similar functionality (CompC), during runtime operation of this test protocol. Following the scenario presented in Fig. 6, the dynamic replacement of CompB was achieved; in addition the new component incorporated itself seamlessly by taking into account the current system composition and configuration. Furthermore the reliable operation of the test protocol was preserved during the replacement process, in terms of achieving no loss of protocol data or existing connections. Therefore the process of autonomic component self-configuration (dynamic component binding and replacement) was successfully validated under the current prototype, thus proving that the protocol components, when reassembled, form the initially specified protocol functionality.

Furthermore, the performance assessment of the proposed system was considered taking into account that in communication devices, the protocol stack subsystem must meet strict performance requirements. Another aspect that should be taken into consideration is the overhead introduced by the incorporation of autonomic features in the new framework, in terms of performance metrics. In order to address these issues and validate the feasibility of the autonomic approach, performance evaluation studies were executed considering the following factors:

- the calculation of the overhead that is created for the dynamic establishment of bindings between the protocol components, due to the introduced of autonomic capabilities,
- the calculation of the time that is necessary for the specified protocol reconfiguration to take place

The delay associated to the reconfiguration of the protocol was defined as the time interval from the time instance the existing protocol component is being signalled by the Autonomic Manager to stop its operation until the time instance the new protocol component is successfully incorporated in the system. The time interval for the original framework includes the following operations:

- Autonomic Manager waiting in idle state until the old component stops (so as the old component is in a safe state at the time of replacement),
- Instantiation/initialization of the new protocol component by the Autonomic Manager,
- Self-configuration of the new protocol component (the new protocol component identifies and establishes its bindings with the existing protocol components on its own)

In particular, Fig. 7 (a) illustrates the binding delay for each component within a set of 100 samples, considering the underlying hardware capabilities (Pentium III 800MHz PC with 512MB RAM) and operating system (Debian Linux 3.0 R4). The mean value for this binding delay is 787,46 μsec for CompA and 462,89 μsec for CompB. The minimum value of the binding delay is found to be 505 μsec for CompA and 415 μsec for CompB, whereas the maximum value is 2039 μsec and 694 μsec for CompA and CompB respectively (it should be noted that the delay is greater for CompA compared to

CompB, since the Autonomic Manager firstly initiates CompA; this way CompA creates the communication queue, whereas CompB simply accesses it).
Moreover, Fig. 7 (b) illustrates the delay that was introduced for the protocol reconfiguration, for a set of 100 samples, considering the aforementioned experiment setup capabilities. The mean value for this delay is 2381 μsec (the minimum and maximum values are 2284 and 2456 μsec correspondingly).
The performance evaluation studies proved that the deployment of the proposed framework and the introduction of autonomic capabilities in the protocol components have minimum performance impact in the system, thus increasing its flexibility.

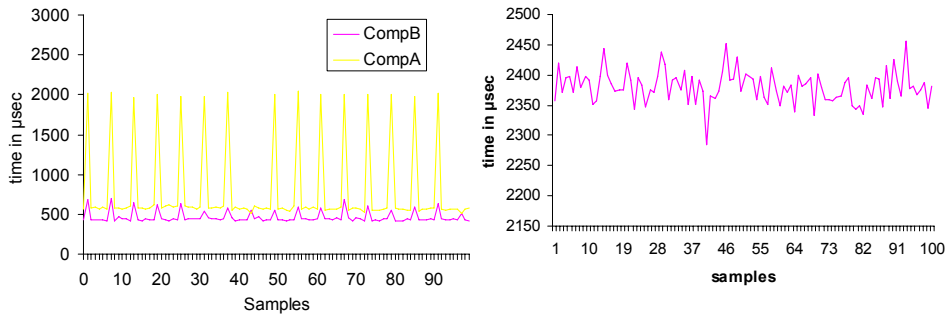

Fig. 7: (a) Binding Delay for the autonomic components within a set of 100 samples, (b) Total time for the replacement of the old component with the new one

5 Related Work

The vision for autonomic computing and communications was the basis for several research activities in the past years in both industry and academia. These activities spawn across the definition, design and deployment of self-* features in emerging communication systems and devices [6]. Following the model proposed by IBM [5], an autonomic system should at least incorporate four attributes: self-configuring, self-healing, self-optimizing and self-protecting, known as self-CHOP features. On the other side, additional features for autonomic communications systems are addressed in [7], including self-awareness, self-adaptation, self-implementation and self-description.
In addition, several work was performed in the context of deploying the above presented self-* features. Since one of the basic capabilities addressed in autonomic communication environments is the autonomic decision making, previous work in policy based management was considered. In particular, the approach presented in [9] should be mentioned, which addresses the policy management issue from a new perspective through posing it as a problem of learning from current system behavior, while creating new policies at runtime in response to changing requirements. A hierarchical policy model is used to capture users and administrators' higher level goals into network level objectives.
Moreover, regarding the introduction dynamic configuration or autonomic capabilities in software subsystems, several other approaches extend the conventional software design.

The above concept was initially applied to X-kernel, Cactus and Appia frameworks, which deal with protocol composition based on microprotocol objects so as to fulfill the application QoS requirements [13][14] . In addition, the CORBA Component Model specifies components for distributed software systems, However it is not appropriate for use in autonomic environments due to its limitations regarding standard defined interfaces and limited extension of object functionality [10][11]. Moreover, the Accord Programming Framework enables the development of autonomic components, but gives limited information on the design blueprints, pertaining to the establishment of component composition and replacement [12].
Furthermore, a different vision on protocol stack design for autonomic communication based on the POEM model is analysed in [15]. This cross-layer design focuses on the advantages of layering and the benefits of holistic and systematic cross-layer optimization is at the core of this work. Finally, a Java-based protocol suite that supports protocol subsystems is analyzed in [17]. This system introduces great performance overhead (10:1) and does not specify the mechanisms required to achieve on-the-fly protocol reconfiguration.
Regarding the game theoretical analysis, it should be noted that it has recently attracted much attention in the general context of resource allocation in wireless networks. Moreover, it can be seen as an appropriate optimization tool for a fully distributed and scalable implementation (with a complexity transferred and shared out on the terminal side), where traditional centralized decision becomes computationally infeasible as the number of terminals in a cell, or the number of carriers in a multicarrier setting grows. Game theory has been applied to many wireless network problems, related to the physical, medium access or higher layers. In those problems, users are generally competing for a limited resource while having a limited knowledge of their environment, and therefore the strategic non-cooperative game model is often used, where each player selfishly tries to maximize its own utility regardless of the consequences its choice may have. For example, [3] deals with a power control game for wideband (e.g. CDMA-based) systems. Random access protocols, in particular ALOHA, are addressed in [4].

6 Conclusion

This paper presented a framework to cope with RAT selection and the requirement for transparent plug-in of protocol-related RAT components in heterogeneous systems. The study results illustrate that a suitable introduction of policies applicable by user terminals impact the system behaviour considerably. The simple example in a two-user context proved, that the interdiction of one strategy (applicable to all users) avoids sub-optimum equilibria and leads to a quasi-immediate convergence in terms of RAT selection strategy. As a follow-up of this work, the corresponding behaviour needs to be analyzed in more complex scenarios consisting of an increased number of users and RAT selection choices.
In addition, a protocol reconfiguration framework was analyzed, which provides the necessary mechanisms for component-based protocol reconfiguration. The signaling for the four basic phases of this procedure was specified as well as the mechanisms that enable the transparent protocol component self-configuration. Finally the validation of

the proposed framework proved its feasibility, thus the introduction of minimum performance overhead in the system.

Acknowledgments. This work has been performed in the framework of the IST project IST-2003-507995 E2R II [18], which is partly funded by the European Union. The authors would like to acknowledge the contributions of their colleagues.

References

1. J. Strassner, "Autonomic networking – theory and practice", In Proc. *9th IFIP/IEEE International Symposium on Network Management* (*IM'2005*), Nice, France, May 2005.
2. D. Fudenberg and J. Tirole, "Game Theory", MIT Press, ISBN 0-262-06141-4, USA, 1991.
3. C.U. Saraydar, N.B. Mandayam and D.J. Goodman, "Efficient Power Control via Pricing in Wireless Data Networks," *IEEE Trans. on Communications*, Feb. 2002.
4. A. B. MacKenzie and S.B. Wicker, "Selfish users in Aloha: a game theoretic approach," In Proc. *Vehicular Technology Conference (VTC)*, vol. 3, Oct. 2001.
5. Jeffrey O. Kephart, David M. Chess. "The Vision of Autonomic Computing," Computer, vol. 36, n° 1, pp. 41-50, Jan. 2003
6. Murch, R., Autonomic Computing, Prentice Hall, 2004.
7. M.Smirnov, "Research Agenda for a New Communication Paradigm", Fraunhofer FOKUS White Paper, Nov. 2004, http://www.autonomic-communication.org/publications/doc/WP_v02.pdf
8. H. Isil Bozma and James S. Duncan, "A game Theoretic Approach to Integration of Modules", IEEE Trans. Pattern Anal. Machine Intell., vol. 16, no. 11, November 1994
9. N. Samaan and A. Karmouch, "An Automated Policy-Based Management Framework for Wired/Wireless Differentiated Communication Systems" in special issue of JSAC on Autonomic Communication systems, December 2005, Volume 23, Number 12
10. Wang, N., Schmidt, D.C., O'Ryan, C. Overview of the CORBA Component Model. In Component-Based Software Engineering: Putting the Pieces Together, Addison Wesley, Editors G.T. Heineman and W.T. Council. 2001.
11. OMG CORBA Components, version 3.0, June (2002), http://www.omg.org/
12. H. Liu and M. Parashar. A component based programming framework for autonomic applications. In Proceedings of the International Conference on Autonomic Computing, New York, NY, 2004.
13. Norman C. Hutchinson and Larry L.Peterson: The x-kernel: An architecture for implementing network protocols. IEEE Transactions on Software Engineering, 17(1):64-76, January 1991.
14. Sergio Mena, Xavier Cuvellier, Christophe Gregoire, Andre Schiper: Appia vs. Cactus: Comparing Protocol Composition Frameworks. 22nd International Symposium on Reliable Distributed Systems (SRDS'03), Florence, Italy, October 2003.
15. X. Gu, X. Fu, H. Tschofeni, L. Wolf, "Towards Self-Optimizing Protocol Stack for Autonomic Communications: Initial Experience", in the Proceedings of the 2nd IFIP International Workshop on Autonomic Communication (WAC'05), Athens, Greece, October 2005.
16. E.Patouni, N.Alonistioti "A Framework for the Deployment of Self-Managing and Self-Configuring Components in Autonomic Environments", in the Proceedings of the International IEEE WoWMoM Workshop on Autonomic Communications and Computing (ACC 06) Niagara-Falls, Buffalo-NY, 26-29 June 2006.
17. B.Krupczak, K.L.Calvert, M.A.Ammar, Implementing Communication Protocols in Java, IEEE Communications Magazine, October 1998.
18. End-to-End Reconfigurability (E2R II), IST-2003-507995 E2R II, http://www.e2r2.motlabs.com