Efficient Service Provisioning in a Multi-Domain Network Environment

Malamati Louta, Evangelos Loutas, Angelos Michalas, Stylianos Kraounakis, Dimitrios Zissopoulos

Abstract— Highly competitive and open environments should encompass mechanisms that will assist service providers in accounting for their interests, i.e., offering at a given period of time adequate quality services in a cost efficient manner which is highly associated to efficiently managing and fulfilling current user requests. In this paper, service task assignment problem is addressed from one of the possible theoretical perspectives, while new functionality is introduced in service architectures that run in open environments in order to support the proposed solution. The pertinent problem is concisely defined, mathematically formulated, computationally efficient solved and evaluated through simulation experiments.

Index Terms— **Intelligent Agents, Service Task Assignment, Service Nodes, Simulated Annealing.**

I. Introduction

The ongoing liberalisation and deregulation of the telecommunication market will introduce new actors [1][2]. In principle, the main role of all players in such a competitive environment will be to constantly monitor the user demand, and in response to create, promote and provide the desired services and service features. The following are some key factors for success. First, the efficiency with which services will be developed. Second, the quality level, in relation with the corresponding cost, of new services. Third, the efficiency with which the services will be operated, controlled, maintained, administered, etc.

The challenges outlined above have brought to the foreground several new important research areas. Some of them are the definition of new business models, the elaboration on e-business concepts [3][4], the specification of service architectures (SAs) [5][6][7], the development of advanced service creation environments (SCEs) [8][9] and service features (e.g. the personal mobility concept [5][10][2]), and the exploitation of advanced software technologies, (e.g. distributed object computing [11][12] and intelligent mobile agents [13][14]). The aim of this paper is, in accordance with the cost-effective QoS provision and the efficient service operation objectives, to propose enhancements to the sophistication of the functionality that can be offered by service architectures in open competitive communications environments.

According to a typical view of the future competitive telecommunications world five main different entities can be identified, namely, user, retailer, (third party) service (or content) provider, broker and connectivity provider. The role of the (third party) service (content) provider is to develop and offer services (content). The role of the retailer is to provide the means through which the users will be enabled to access the services (content) of (third party) service (content) providers. Limited by techno-economic reasons or considering administrative, management and resilience/redundancy purposes each service provider deploys service components realising service logic in different service nodes, residing in the same and/or different network domains. Moreover, it can be envisaged that a service will in general comprise a set of distinct service tasks, which could be executed by different service nodes. The broker assists business level entities in finding other business entities. Finally, the role of a connectivity provider is to offer the network connections needed for service provision.

Highly competitive and open environments should encompass mechanisms that will assist service providers in accounting for their interests, i.e., offering at a given period of time adequate quality services in a cost efficient manner which is highly associated to efficiently managing and fulfilling current user requests. Specifically, assuming that the service is constituted by a set of service tasks and various candidate service nodes may be involved in the service provisioning process then a problem that should be addressed is the assignment of service tasks to the most appropriate service nodes. In this paper, the pertinent problem is called *service task assignment*. The aim of this paper is to address the problem from one of the possible theoretical perspectives and to show how the solution can be incorporated in service architectures that run in the open environment. The presented framework can be applied to numerous distributed service provisioning systems, among others, for network management purposes, information retrieval, location based services, etc. Our reference service architecture bears resemblance with the one specified the PARLAY/OSA Framework [15] and the Telecommunications Information Networking Architecture Consortium (TINA-C) [6]. However, the presented practices can be applied to other models as well.

The approach in this paper is the following. The starting point (section II) is the general description of the service task assignment concept, through the presentation of a relevant business case. Section III and IV present a concise definition, mathematical formulation and solution to the service task assignment problem, respectively. Section V

Malamati Louta and Angelos Michalas are with the National Technical University of Athens, School of Electrical and Computer Engineering, Athens, Greece (telephone: 30-210-7721447, e-mail: louta@telecom.ntua.gr, michalas@central.ntua.gr.

Evangelos Loutas is with University of Thessaloniki, Department of Informatics, Thessaloniki, Greece (e-mail: eloutas@zeus.csd.auth.gr).

Stylianos Kraounakis is with City University, London, England

Dimitrios Zissopoulos is with Macedonia Research Center, Greece.

describes the service node selection algorithm applied in our framework. Finally, section VI gives future plans and concluding remarks.

II. General Presentation of the Service Task Assignment Concept

This section starts from the description of the business case, through which the role (and importance) of the service task assignment concept can be understood. *Sub-section A* provides the description in terms of business level entities (i.e., users and service providers), while in *sub-section B* the description is refined by introducing the role of the computational level components.

A. Description in terms of Business Entities

Assume that a user wishes to access a specific service offered by a service provider. The service is composed by a distinct set of service tasks. Each service task can be served by various candidate service nodes (CSNs). The choice of the most appropriate service node engagement for the completion of each service task (service task assignment process) requires the realisation of the three general phases depicted in Fig. 1.

The first general phase involves service independent features like user authentication, authorisation, etc. It involves the user and an entity that will be called *Default Retailer* (DR) residing in the Default Domain. In essence, at the end of this phase the user is enabled to request services. This phase will not be further addressed.

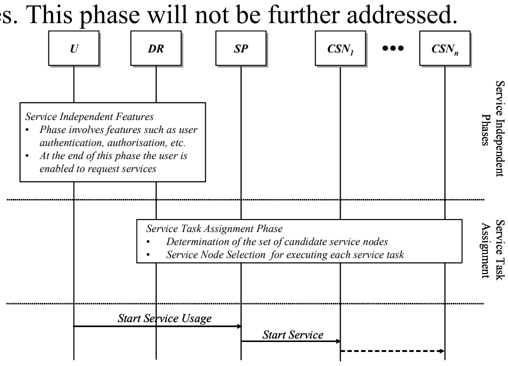

Fig. 1. Interactions among the business level entities during the service task assignment business case.

At the second phase, the service task assignment process is conducted by the respective *Service Provider* (SP) entity, which is specialised in the assistance of the service provider in the open competitive communication environment. The SP can accomplish this by providing, maintaining and hosting (essential parts of) the software that will conduct the service task assignment process. In this respect, the SP is assumed to play a co-ordinating role in the second general phase, which is the core of the service task assignment process. At this point the user has expressed the wish to access a given service. Involved in this phase will be the SP, the DR and the service provider's CSNs that could be deployed for the provision of the service. In general, service task assignment is founded on general and service specific user preferences and provider's specific service logic deployment. At this point it should be noted that the appropriate SP is determined by the DR at the end of the first general phase based on the user preferences/ requirements regarding the requested service.

In the third phase of the business case the result of the service task assignment is available, and hence the service usage can possibly start, in accordance to the specific task assignment obtained during the previous stage.

At this point, some concepts concerning the business case can be outlined. Specifically, the scenario presented accounts for both the user and the service provider. Specifically, service providers succeed in better managing their resources, while users implicitly exploit in a seamless and transparent way the otherwise unutilised power and capabilities of provider's network. Thus, the SP assists service providers in equitably and efficiently distribute their resources, in essence leading to a higher level QoS service provision to the users.

Based on the described business case we may provide the high level definition of the pertinent design problem. This means that we should define the cost function and specify the constraints that derive from the requirements of (primarily) the user and the provider's policies in conjunction with the respective capabilities and load conditions of the service nodes. The solution in our case should provide the minimum cost assignment of service tasks to service nodes.

The user requirements may be characterised in terms of service preferences. Service preferences yield the service tasks needed for the service provision, as well as the load that will originate from each service task, which may be expressed in terms of an associated with each service task, CPU time, memory and disk resources. In essence, these values correspond to the service node CPU time, memory and disk space required by the service task, so as it is adequately provided.

The cost function of the service task design problem may consist of the following factors. First, the cost of the service nodes that need to be deployed (involved in the solution). Second, the communication cost of components between the service node that has primarily undertaken the execution of the service task requested and the service nodes that may as well be involved in the accomplishment of the specific service task (e.g., one may consider the case of a service task requiring additional processing to data retrieved from a database server). Third, the management cost introduced due to the assignment of service tasks constituting a service to different service nodes. The constraints of the problem derive from the capabilities of the service nodes. These may be expressed in terms of their maximum resources (i.e., CPU time, memory and disk space), and probably, the maximum number of tasks they can control at the same time (i.e., number of parallel sessions).

Taking into account the aspects outlined above, a general problem statement may be the following. Given the set of candidate service nodes and their layout, the set of service tasks constituting the required service, the resource requirement of each service task in terms of CPU time, memory and disk space, the cost of deploying each service node, the current load conditions of each service node, find the minimum cost assignment of tasks to service nodes (in terms of the number of nodes that need to be deployed, the communication cost introduced during the execution of service tasks, and the management cost imposed by the arrangement) subject to a set of constraints, associated with the capabilities of the service nodes.

This paper (section III) includes a mathematical description of a general version of the problem. At a next step the problem version is formulated as a 0-1 linear programming problem [16], and a brief outline of computationally efficient solution algorithms is presented. The problem is solved by an heuristic algorithm following the simulated annealing technique.

B. Description in terms of Computational Entities

A computational level model of the business case is depicted in Fig. 2. Of interest to our study is the access session concept. In general, a session is defined as the temporary relationship among a group of objects that are assigned to collectively fulfill a task for a period of time. The access session is a service independent concept, which can be seen as the gateway to any specific service usage. It comprises activities that allow user authentication, user profile control (inspection), and service invocation.

The Initial Agent (IA) is the component that enables the initial access to a domain. The User Agent (UA) component represents the user beyond the terminal e.g., in the default domain. Its role is to intercept and process user requests. The UA maintains user profile related information e.g. preferences, requirements and constraints regarding certain services, service subscriptions, etc. The User Application Agent (UAP) models the entity (user interface) with which the user is confronted in access session mode. The UA invokes the Service Factory (SF) for initiating a service.

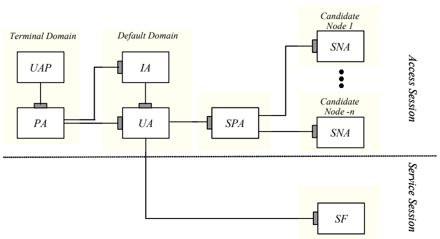

Fig. 2. TINA-like computational model for the service task assignment business case.

As a first step, service task assignment requires a computational component that will act on behalf of the user. Its role will be to capture the user preferences, requirements and constraints regarding the requested service and to deliver them in a suitable form to the appropriate service provider entity. As a second step, service task assignment process requires an entity that will act on behalf of the service provider. Each role will be to intercept user requests, acquire and evaluate the corresponding service node load conditions, and ultimately, to select the most appropriate service nodes for the realisation of the service. Furthermore, a monitoring module is required. Monitoring module consists of a distributed set of agents, which run on each service node of the service provider. Each agent is responsible for monitoring the load conditions and available resources of the service node and delivering them to the service provider related entity.

The following key extensions are made so as to cover the functionality that was identified above. First, the *Service Provider Agent* (SPA) is introduced and assigned with the role of selecting on behalf of the service provider the best service task assignment pattern. Second, the UA is extended by being assigned with the role of promoting the service request to the appropriate service provider agent. Finally, the *Service Node Agent* (SNA) is introduced and assigned with the role of promoting the current load conditions of a CSN. In essence, the distributed set of the SNAs forms the monitoring module. In other words, as depicted in Fig. 2, the SPA interacts with the UA in order to acquire the user preferences, requirements and constraints, analyses the user request in order to identify the service tasks constituting the service, identifies the set of CSNs and their respective capabilities, interacts with the SNAs of the candidate service nodes so as to obtain their current load conditions, and selects the most appropriate service task assignment pattern for the provision of the desired service.

The tasks outlined require a method that will enable service providers to process the user's request and generate a service task assigning scheme, satisfying user's preferences, requirements, and constraints, at the same time resulting in an efficient resource management scheme on the service provider's side.

III. Formal Problem Statement

User u wishes to use a given service s. A fundamental assumption at this point is that service s may be decomposed in a set of distinct service tasks, which will be denoted as $ST(s)$. Among these service features, of interest to the user are those designated in the user profile and will be denoted as $ST(u,s)$ ($ST(u,s) \subseteq ST(s)$).

Let's assume the existence of multiple service nodes for the provision of service s, denoted by $SN(s) = \{n_1,...,n_{|s|}\}$. Each service node-$n_j$ contains a collection of components, denoted as $A_{n_j}(i)$, which inter-work with other components that may reside in the same or in a different service node in order to accomplish each service task $i \in ST(s)$. Let A_{n_j} and C be the total set of components residing in the n_j service node and the various service nodes in total, respectively. Hence, the following relationship holds: $A_{n_j}(i) \subseteq A_{n_j} \subseteq C$. Each service task $i \in ST(s)$ may be executed on an associated set of possible candidate service nodes, represented by the set $SN(i)$, ($i \in ST(u,s)$). Thus, $SN(i) \subseteq SN(s)$. The service logic deployment pattern adopted by service providers determine each of these service node sets.

Task i, ($i \in ST(s)$) requires for its completion consumption of $r(i)$ resources of service node(s) n_j, ($n_j \in SN(i)$). A realistic assumption is that SPA being in charge of assisting the service providers in the competitive telecommunication market, has a solid interest in as accurately as possible identifying the resources $r(i)$ needed for the provisioning of service task i in terms of CPU time, memory and disk space. In this respect, the SPA can be the entity that configures these values based on the service feature characteristics, user preferences and requirements, exploiting also previous experience.

Let c_D denote the cost of involving service node

n_j ,$(n_j \in SN(i))$, in the service provision. For notation simplicity it is assumed that the cost of involving a service node in the solution is the same for all service nodes. As an alternative this cost could be taken variant (depending on the cost of acquiring and/or maintaining the node etc.). Notation may readily be extended.

The objective of our problem is to find a service task assignment pattern, i.e., an assignment $A_{ST}(s)$ of service tasks i ($i \in ST(u,s)$) to service nodes n_j ,$(n_j \in SN(i))$, that is optimal given the current load conditions and number of service tasks being served by each service node n_j, represented as $r_{pre}(n_j)$ and $k_{pre}(n_j)$, respectively. The assignment should minimise an objective function $f(s, A_{ST}(s))$ that models the overall cost introduced due to network resources consumption. Among the terms of this function there can be the overall cost due to the deployment of various service nodes to the service provisioning process, the communication cost introduced due to the interaction of the components A_{n_j} residing in n_j service node with the components A_{n_k} residing in service node n_k for the completion of each service task i, $(\forall i \in ST(s))$ (one possible communication cost model will be presented in *sub-section A*), as well as the management cost $c_M(i,i')$ introduced due to the assignment of $(i,i') \in ST^2(s)$ service tasks to different service nodes $(n_j, n_{j'}) \in SN^2(i)$.

The constraints of our problem are the following. First, each service task i ($i \in ST(u,s)$) should be assigned to only one service node n_j, $(n_j \in SN(i))$. Second, the capacity constraints of each service node should be preserved. Lets assume that $r_{sn}^{\max}$ and $k_{sn}^{\max}$ represent the maximum load (in terms of CPU-time, memory, disk space) and the maximum number of tasks that a service node may handle. For notation simplicity, these parameters are assumed to be the same for each service node n_j, $(n_j \in SN(i))$. Thus, the constraints are $r_{post}(n_j) \le r_{sn}^{\max}$ and $k_{post}(n_j) \le k_{sn}^{\max}$, $(\forall n_j \in SN(i))$, where $r_{post}(n_j)$ and $k_{post}(n_j)$ denote the potential load conditions of service node n_j, after the service task assignment process. Notation may readily be extended.

The overall problem can be formally stated as follows.

[Service Task Assignment Problem]. Given:

(a) a user u who wants to use a service s,

(b) the profile of user-u,

(c) the set of service tasks $ST(u,s)$ of service s that are of interest (relevant) to user u (this set is formed by the service specification, the user profile and the service provider's related capabilities),

(d) the set of service nodes $SN(s)$ and the set of candidate service nodes $SN(i)$ at which each service task i ($i \in ST(u,s)$) can be completed, according to the service specification, the service node capabilities and the preferences of user u,

(e) the communication cost introduced due to the interaction of the components A_{n_j} residing in n_j service node with the components A_{n_k} residing in service node n_k for the completion of each service task i, $(\forall i \in ST(s))$,

(f) the deployment cost c_D of each service node n_j involved in the service provision process, which derives from the assignment of service task i, ($i \in ST(u,s)$) to service node n_j ($n_j \in SN(i)$),

(g) the management cost $c_M(i,i')$ introduced due to the assignment of $(i,i') \in ST^2(s)$ service tasks to different service nodes $(n_j, n_{j'}) \in SN^2(i)$,

(h) the current load conditions $r_{pre}(n_j)$ and number of service tasks $k_{pre}(n_j)$ being executed on each service node n_j, $n_j \in SN(s)$,

(i) the capacity constraints of each service node $r_{sn}^{\max}$ and $k_{sn}^{\max}$,

(j) the resources $r(i)$ required for the completion of service task i, $(\forall i \in ST(s))$,

find the best service task configuration pattern, i.e., assignment of service tasks to service nodes $A_{ST}(s)$, that optimises an objective function $f(s, A_{ST}(s))$ that is related to the overall cost introduced by the assignment, under the constraints $r_{post}(n_j) \le r_{sn}^{\max}$ and $k_{post}(n_j) \le k_{sn}^{\max}$, and that each service task is assigned to exactly one service node.

A. Communication Cost Model

Following we will present a model for the overall communication cost introduced in case n_j service node has undertaken the responsibility for the execution of service task i ($i \in ST(u,s)$). In essence, the model covers the case in which the components of set $A_{n_j}(i)$ need to interact with the components of set $A_{n_k}(i)$ residing in service node n_k in order to provide service task i, $(i \in ST(s))$. Otherwise, in case only the components of set $A_{n_j}(i)$ are involved in the execution of the service task i, $(i \in ST(s))$, the communication cost is considered to be negligible and is taken equal to zero. At this point, a major assumption adopted in our study, is that part of A_{n_j} components are implemented as mobile agents, while the rest are supposed to be fixed service agent components. Let $A_{n_j}^M$ and $A_{n_j}^F$ be the subset of components of A_{n_j} that are implemented as mobile and fixed agents, respectively.

The volume of messages exchanged between each pair of components (e.g., dependent on the number of messages and size of each message) for the accomplishment of task i ($i \in ST(s)$) will be represented as $m_{wv}(i)$, $\forall (w,v) \in C^2$ and $\forall i \in ST(s)$. Let $cc(n_1, n_2)$ be the communication cost per unit message that is exchanged between service nodes n_1 and n_2, $\forall (n_1, n_2) \in SN(s)^2$. This factor may be proportional to the distance (e.g., number of hops) between the two service nodes. Another factor that should be taken into account is the cost associated with the migration of a component (implemented as a mobile agent) from one service node to another. In this respect, let $mc(w, n_1, n_2)$ be the migration cost of component-w from service node n_1 to service node n_2, $\forall w \in C$ and $\forall (n_1, n_2) \in SN(s)^2$.

Assuming that the components of set A_{n_j} are implemented as fixed agents, the overall cost for the completion of task i $(i \in ST(s))$ can be calculated by the following formula.

$$C_1(n_j \to n_k) = \sum_{w \in A_{n_j}} \sum_{v \in A_{n_k}} m_{wv}(i) \cdot cc(n_j, n_k),\ \forall n \in SN(s),\ \forall i \in ST(s) \quad (1)$$

The above cost is proportional to the messages (their number and size) that are exchanged between every pair of components (w,v) and the communication cost per unit message between different service nodes.

In case part of the components that reside in service node n_j are implemented as mobile agents the following approach is adopted. The overall cost for the completion of task i $(i \in ST(s))$ can be calculated by the following formula.

$$C_2(n_j \to n_k) = \sum_{w \in A^F_{n_j}} \sum_{v \in A_{n_k}} m_{wv}(i) \cdot cc(n_j, n_k) + \sum_{w \in A^M_{n_j}} mc(w, n_j, n_k) + \sum_{w \in A^M_{n_j}} \sum_{v \in A_{n_k}} m_{wv}(i) \cdot cc(n_k, n_k)\ \ \forall n \in SN(s),\ \forall i \in ST(s) \quad (2)$$

In the previous formulation three main factors are identified. The first one is related to the communication cost deriving from the fixed components and is analogous to equation (1). The second factor is associated with the migration cost of mobile agent components between two different service nodes. This factor is dependent on the path which the mobile agent will follow (i.e., number of hops) and the information encryption and code execution cost. The last factor is the communication cost within the same service node, which in practice may be negligible, as already assumed.

Apparently, the designation of the components that will be included in sets $A^M_{n_j}$ and $A^F_{n_j}$ by the service providers may be based on factors such as the overall communication and migration costs as well as estimation of the respective component invocations. In our study, the service logic deployment pattern (i.e., service components, service nodes) adopted by the service providers is known.

IV. Service Task Assignment Problem Solution

The general problem version presented in section III is open to various solution methods. Its generality partly lies in the fact that the objective and the constraint functions are open to alternate implementations. Thus, the problem statement can be distinguished from the specific solution approach adopted in this section. In *sub-section A* the service task assignment problem is optimally formulated, while in *sub-section B* computationally efficient solutions are discussed.

A. Optimal Formulation

In order to describe the allocation $A_{ST}(s)$ of service tasks to service nodes we introduce the decision variables $x_{ST}(i,j)$ ($i \in ST(u,s), n_j \in SN(i)$) that take the value 1(0) depending on whether service task i is (is not) executed by service node-n_j. The decision variables $y_{SN}(j)$ assume the value 1(0) depending on whether candidate service node n_j ($n_j \in SN(i)$) is (is not) deployed (involved in the solution). In addition, we define the set of variables $z_{ST}(i,i')$ ($\forall (i,i') \in ST(u,s)^2$) that take the value 1(0) depending on whether the service tasks i and i' are (are not) assigned to the same service node. The variables $z_{ST}(i,i')$ are related to variables $x_{ST}(i,j)$, $x_{ST}(i',j)$, through the relation $z_{ST}(i,i') = \sum_{j=1}^{|SN(i)|} x_{ST}(i,j) \cdot x_{ST}(i',j)$, which may be turned into a set of linear constraints through the technique of [9]. Allocation $A_{ST}(s)$ may be obtained by reduction to the following 0-1 linear programming problem.

Service Task Assignment Problem:

Minimise

$$f(s, A_{TN}(s)) = c_D \cdot \sum_{n_j \in SN(s)} y_{SN}(j) \cdot (1 + a \cdot r_{pre}(n_j)) + \sum_{i \in ST(s)} \sum_{n_j \in SN(i)} C(i, n_j) \cdot x_{ST}(i,j) + \sum_{i \in ST(s)} \sum_{i' \in ST(s)} c_M(i,i') \cdot (1 - z_{ST}(i,i')) \quad (3),$$

where,

$$C(i, n_j) = \sum_{\forall n_k \in SN(s)} \Big[\sum_{w \in A^F_{n_j}} \sum_{v \in A_{n_k}} m_{wv}(i) \cdot cc(n_j, n_k) + \sum_{w \in A^M_{n_j}} mc(w, n_j, n_k) + \sum_{w \in A^M_{n_j}} \sum_{v \in A_{n_k}} m_{wv}(i) \cdot cc(n_k, n_k) \Big] \quad (4),$$

subject to the constraints:

$$\sum_{n_j \in SN(i)} x_{ST}(i,j) = 1 \ \ \forall i \in ST(s) \quad (5),$$

$$r_{pre}(n_j) + \sum_{i \in ST(s)} r(i) \cdot x_{ST}(i,j) \le r^{\max}_{sn}(j) \cdot y_{SN}(j) \ \ \forall n_j \in SN(s) \quad (6),$$

$$k_{pre}(n_j) + \sum_{i \in ST(s)} x_{ST}(i,j) \le k^{\max}_{sn}(j) \cdot y_{SN}(j) \quad \forall n_j \in SN(s) \quad (7)$$

Cost function (3) penalises the aspects identified in section III (i.e., cost of the service node involved in the solution, communication cost introduced during the realisation of each service task (given by equation (3)), and management cost of service tasks that are assigned to different service nodes). In order for the service providers to better utilize their resources, the cost of each service node deployment introduced in cost function (3) takes also into account the node's current load conditions in order to obtain a load balancing solution. Parameter a, $(a < 1)$, denotes the relative significance of load balancing to the service provider. Equation (4) provides the communication cost introduced during the execution of service task i on service node n_j, where only the involved to the provisioning process components are taken into account. Constraints (5), guarantee that each service task will be assigned to one service node. Constraints (6) and (7) guarantee that each service node will not have to cope with more load and service tasks than those dictated by its pertinent capacity constraint.

B. Computationally Efficient Solution

Exhaustive search should be conducted in case the solution space is not prohibitively large. These algorithms apply a method for scanning all the solution space. The cost of the respective solutions is evaluated and finally, the best solution is maintained. The complexity of the search in our case is $\prod_{i \in ST(u,s)} SN(i)$, i.e., a function of the service tasks that are relevant to the service user requested and the service nodes at which these service tasks may be executed.

In general, there may be a significant amount of computations associated with the optimal solution of the service task assignment problem. In this respect, the design of computationally efficient algorithms that may provide good (near-optimal) solutions in reasonable time is required. Classical methods in this respect are simulated annealing [17], genetic algorithms [18], taboo search [19], greedy algorithms [16], etc. Hybrid or (user-defined) heuristic techniques may also be devised.

Simulated Annealing has been adopted in our framework. During each phase of the algorithm, a new solution is generated by minimally altering the currently best solution. If the new solution improves the cost function value (i.e., the difference between the cost function value of the new and the old solution, Δc , is negative) the new solution becomes the currently best solution. Solutions that increase the cost function value may also be accepted with a probability, which assists in escaping from local optima. Initial results acquired were satisfying compared to those of an exhaustive search, since a near optimal solution is reached quicker and by utilizing less computational power.

V. Service Node Selection Algorithm

In this section the resulting service node selection algorithm is described and the functionality of the involved computational level components is presented.

Step 1. The UA component is acquainted with the preferences, requirements and constraints of user u regarding service s. These are expressed by the following data. First, the set of the service tasks $ST(u,s)$ that are of interest (relevant) to the user. Second, the reference of the respective SPA.

Step 2. The SPA obtains from the UA the aforementioned information and retrieves from a database the set of candidate service nodes $SN(i)$ for the completion of each service task i, ($i \in ST(u,s)$), the deployment cost c_D of each service node n_j, ($n_j \in SN(i)$) and their respective capacity constraints $r_{sn}^{\max}$ and $k_{sn}^{\max}$, the communication cost $C(i,n_j)$ introduced during the execution of each service task i, ($i \in ST(u,s)$) and the management cost $c_M(i,i')$ ($(i,i') \in ST^2(s)$). Additionally, the SPA computes for each service task i ($i \in ST(u,s)$) the corresponding resources $r(i)$ required for its completion in terms of CPU time, memory and disk resources.

Step 3. The SPA component interacts with the SNAs in order to obtain the corresponding load conditions $r_{pre}(n_j)$ and $k_{pre}(n_j)$ of each CSN n_j, $n_j \in SN(s)$.

Step 4. The SPA solves the appropriate instance of the service task assignment problem.

Step 5. End.

VI. Conclusions

The highly competitive communications markets of the future should encompass mechanisms that will assist service providers in accounting for their interests, i.e., offering at a given period of time adequate quality services in a cost efficient manner which is highly associated to efficiently managing and fulfilling current user requests. This paper presented such mechanisms. Our starting point was the definition of a business case, through which the role of the service task assignment problem was explained. In the sequel, the problem was analysed, concisely defined, mathematically formulated and solved. The identified components of the service task assignment problem were targeted to acquiring the best service task configuration pattern, i.e., assignment of service tasks to service nodes for efficient service provisioning. A model for the communication cost between the components involved during the provision of a service task was also provided. At the final section the resulting service node selection algorithm was described.

Directions for future work include, but are not limited to the following. First, the realisation of further wide scale trials, so as to experiment with the applicability of the framework presented herewith. Second, the experimentation with alternate approaches for solving the service task assignment problem.

References

[1] H. Ungerer and N. Costello, *Telecommunications in Europe: Free choice for the user in Europe's 1992 market*, 2nd ed.. Office for Official Publications of the European Communities, Brussels, 1990.

[2] D. Prevedourou et al., "Use of agent technology in service and retailer selection in a personal mobility context", *Computer Networks Journal*, vol. 31, no. 19. pp. 2079-2098, 1999.

[3] S. Ghosh, "Making business sense of the Internet", *Harvard Business Review*, vol. 76, no. 2, pp. 127-135, 1998.

[4] R. Kalakota and A. Whinston, *Electronic commerce: A manager's guide*. Addison-Wesley Publishing Company, 1997.

[5] S. Trigila, K. Raatikainen, B. Wind, and P. Reynolds, "Mobility in long-term service architectures and distributed platforms", *IEEE Personal Commun.*, vol. 5, no. 4, pp. 44-55, 1998.

[6] T. Magedanz, "TINA-Architectural basis for future telecommunications services", *Computer Communications*, vol. 20, no. 4, pp. 233-245, 1997.

[7] A. Lazar, "Programming telecommunication networks", *IEEE Network*, vol. 11, no. 5, pp. 8-18, 1997.

[8] M. Tag, "Service creation environment engineering", *Proc. Interworking'96 Conference*, Japan, 1996.

[9] P. Demestichas, F. Westerhuis, M. Guyot, C. Loftus, E. Sherratt, and M. Anagnostou, "Service creation environment for open distributed service architectures", *Proc. IEEE International Conference on Intelligence in Networks*, France, 1998.

[10] P. Demestichas, E. Tzifa, M. Louta, N. Liossis, A. Kaltabani, and M. Anagnostou, "Personal Mobility Support in Future Service Architecture", *International Journal of Communication Systems*, vol. 14, pp. 837-856, 2001.

[11] S. Vinoski, "CORBA: Integrating diverse applications within distributed heterogeneous environments", *IEEE Commun. Mag.*, vol. 35, no. 2, pp. 46-55, 1997.

[12] Object Management Group (OMG) Web site, http://www.omg.org.

[13] "Mobile software agents for telecommunications", Feature topic in the *IEEE Commun. Mag.*, vol. 36, no. 7, 1998.

[14] P. Morreale, "Agents on the move", *IEEE Spectrum*, vol. 35, no. 4, pp. 34-41, 1998.

[15] Parlay Group, http://www.parlay.org/

[16] C. Papadimitriou and K. Steiglitz, *Combinatorial optimization: Algorithms and complexity*. Prentice Hall, Inc., 1982.

[17] E. Aarts and J. Korts, *Simulated annealing and the Boltzmann machines*. J. Wiley & Sons, New York, 1989.

[18] Z. Michalewicz, *Generic algorithms + Data structures = Evolution* programs. Springer-Verlag, Berlin, 1995.

[19] F. Glover, E. Taillard, and D. de Werra, "A User's Guide to Taboo Search", *Annals of Operations Research*, vol. 41, pp. 3-28, 1993.