Checking Service Instance Protection for AMF Configurations

P. Salehi, F. Khendek
ECE Department, Concordia University
Montréal, Canada
{pe_saleh, khendek}@ece.concordia.ca

M. Toeroe
Ericsson Canada Inc.
Montréal, Canada
Maria.Toeroe@ericsson.com

A. Hamou-Lhadj, A. Gherbi
ECE Department, Concordia University
Montréal, Canada
{abdelw, gherbi}@ece.concordia.ca

Abstract - An AMF configuration is a logical organization of resources, components and Service Units (SUs) into Service Groups (SGs), for providing and protecting services defined as Service Instances (SIs). The assignment of SIs to SUs is a runtime operation performed by a middleware implementing the AMF service. However, ensuring the capability of the provisioning and the protection of the SIs by the configured application is a configuration issue. In other words, a configuration is valid if and only if it is capable of providing and protecting the services as required and according to the specified redundancy model. Ensuring this may require the exploration of all possible SI-SU assignments and in some cases different combinations of SIs, a complex procedure in most redundancy models defined in the AMF standard specification. In this paper, we explore the problem of SI protection at configuration time; we investigate and discuss its complexity and identify some special and more tractable cases.

Keywords - Availability Management Framework, Configurations, Service Instance Protection, Validation, Complexity.

I. INTRODUTION

High availability is a requirement for services in several domains including banking and telecommunications. Traditionally, high-availability has been achieved with proprietary solutions. Service Availability Forum (SAF) [1] is defining and standardizing high availability solutions for systems and services. Among others, SAF has developed the Application Interface Specification (AIS), which includes the Availability Management Framework (AMF) [2]. The role of AMF is to manage the availability of the service provided by an application. This consists in managing the redundant components composing an application and in shifting dynamically the workload assigned to a faulty component to a redundant and healthy one when a fault is detected. In order for AMF to manage the availability of the services delivered by an application under its control, it requires a configuration.

An AMF configuration [2] for a given application is a logical organization of resources for providing and protecting services. An AMF configuration consists of components grouped into service units (SUs), which are grouped into services groups (SGs). An application may consist of one or several SGs to provide and protect services defined in terms of service instances (SIs) composed of component service instances (CSIs). At runtime, for each SI, the middleware implementing the AMF service assigns the active and standby high availability (HA) states to SUs, according to the redundancy models. The AMF specification defines five redundancy models [2]: the No-redundancy, the 2N, the N+M, the NWayActive and the NWay redundancy models.

The problem of ensuring SI protection does not occur when configurations are generated automatically to provide and protect the required services [5, 6], but when configurations are designed in ad hoc manner and the question of SI protection arises afterwards. In other situations, new or additional services may be required afterwards and the question of providing and protecting all the services with the current configuration may arise.

We have been developing an UML profile for AMF [3] and used the domain model for the validation of AMF configurations [3], for example ones that are created by designers using traditional tools. The issue of validating the capability of the provisioning and the protection of the services aroused in this case as well. Indeed a configuration is AMF compliant if and only if it meets all the AMF requirements including the protection of services as required. Validating the protection of services as requested may require the exploration of all possible SI combinations and SI-SU assignments. This is a combinatorial and complex problem in general. In this paper we explore and discuss this issue for the redundancy models defined in the AMF specification [2]. We discuss its complexity and identify some specific cases where the problem can be solved efficiently and complexity overcome.

The rest of this paper is organized as follows. In Section 2, we introduce some definitions and notations used throughout the paper. We discuss the 2N and the No-redundancy model and propose solutions for deciding about the SI protection for these two cases in Section 3. In Section 4, we explore the case of the N+M redundancy model and show the complexity of the SI protection problem in this case. In Section 5, we briefly discuss NWayActive and NWay redundancy models before identifying in Section 6 some special cases where complexity can be overcome. We conclude in Section 7.

II. DEFINITIONS AND NOTATIONS

Provided services from the provider perspective, or requested services from the requester perspective, can be defined in terms of component service types (CSTs) and the number of CSIs of each CST provided or requested, respectively. We therefore define a *capacity,* for a component, SU or an SI, as a set of pairs of elements *(cst,* *i)*, where *cst* and *i* define a CST and an integer referring to the capacity in terms of CSIs, respectively.

The capacity of an SU, for instance, is an aggregation of the capacities of the different components composing the SU. Some of these components may be overlapping in service provisioning, i.e. may support the same CST; similarly for an SI. Therefore the definition of the capacity of an SU or an SI is not a simple union of the capacities of the components or CSIs composing the SU or the SI, respectively, but it is defined as a cumulative union where capacities referring to the same CST are added together as defined in part 1 of Equation 1. For simplicity, the cumulative union is defined only for two sets of capacities in this equation, but applies for *n (n>2)* sets of capacities, where the cumulative union of the first *k-1* sets is cumulated with the set *k*, in a recursive manner.

In the following sections, we also need to compare capacities provided by SUs against capacities requested by SIs. A capacity *A* is higher than a capacity *B*, if and only if for every pair *(cst, b)* in *B* there exist a pair *(cst, a)* in A referring to the same CST with *a* higher than *b*. The comparison of capacities is formally defined in part 2 of Equation 1, while division between capacities is defined in part 3 of Equation 1.

$$A, B, C : capacity;$$

$$1. A = B \overset{CUM}{\bigcup} C \quad where$$

$$A = \{(a,b) \mid ((\exists\, (a,x) \in B \ \wedge \ \exists (a,y)) \ \in C) \rightarrow (b = x + y)) \ or$$
$$((\exists (a,b) \ \in B \ \wedge \neg \exists (a,w) \in \ C)) or$$
$$((\exists (a,b) \ \in C \ \wedge \neg \exists (a,v) \in \ B))\}$$

$$2. A \geq B \ iff \ \forall (cst,b) \in B \ \ \exists (cst,a) \in A | a \geq b$$

$$3. A = B \, div \, C \ iff \forall (x,y) \in B,$$
$$if \ \exists (x,z) \in C \longrightarrow (x, \lfloor y/z \rfloor) \in A$$
$$if \ \neg \exists (x,z) \in C \longrightarrow (x,y) \in A$$

Equation 1. Cumulative union, comparison and division of capacities.

In an AMF configuration, SUs are grouped into SGs. We refer to the set of SUs in an SG with *SUSet* and use *SISet* for the set of SIs protected by the SG. We denote by *ActiveCapacity(x)* for an SU *x* the total capacity the SU *x* can support in the active role. Similarly, *StandbyCapacity(x)* denotes the total capacity the SU *x* can support in the standby role. The active capacity required by an SI *y* is denoted by *RequiredActiveCapacity(y)*. The total active capacity required from an SU *x* in a given SU-SI assignment *A* is denoted by *RequiredActiveCapacityFrom(x)* and is defined by the cumulative union of all the required active capacities of the SIs associated to *x* through assignment *A*. Similarly, the standby capacity required by an SI *y* is denoted by *RequiredStandbyCapacity(y)* and the total standby capacity required from an SU *x* in a given SU-SI assignment *A* is denoted by *RequiredStandbyCapacityFrom(x)* and is defined by the cumulative union of all the required standby capacities of SIs associated to *x* through assignment *A*.

For purpose of clarity, we briefly remind, in Table 1, the reader of some mathematical notations used throughout this paper and where R is a relation between two sets X and Y.

TABLE 1. Some mathematical notations.

Notation	**Meaning**
Domain(R)	The set of elements in X that are in a pair in R, i.e. {x | ∃ (x, y) ∈ R)
Range(R)	The set of elements in Y that are in a pair in R, i.e. {y | ∃ (x, y) ∈ R)
|X|	The cardinality (the size) of set X.
∃!	There exists one and only one

We also remind the reader that the AMF specification [2] requires that any SU in an SG must be able to protect any of the SIs protected by the SG. Furthermore, we make the reasonable assumption that all SUs in an SG are identical, i.e. they have identical capacity with respect to the SIs.

III. SERVICE INSTANCE PROTECTION FOR THE 2N AND NO-REDUNDANCY MODELS

In an SG with the 2N redundancy model, at most one SU will have the active HA state for all SIs and is referred to as the active SU, and at most one SU will have the standby HA state for all SIs and is usually called the standby SU. Any SU should be capable of taking the active or the standby role for all SIs [2].

In order to capture unambiguously the meaning of the 2N redundancy model for an SG, we define it formally as shown in Equation 2. We consider any two different SUs in the SG, *x* and *y,* and define two relations; the first one is for the active assignment while the second one is for the standby assignment. *ActiveAssignment* and *StandbyAssignment* are defined as relations between one SU and the set *SISet* of SIs, with the following properties:

- *ActiveAssignment* relation is defined as a set of pairs with a range equal to the set *SISet*. Similarly, for *StandbyAssignment* relation. Therefore, each SI is taken care of once and only once, for both the active and the standby assignments.
- The capacity required, from an SU, does not exceed the SU capacity, for both the active and the standby assignments, as specified in *RequiredActiveCapacity(x)* ≤ *ActiveCapacity(x)* for the active part, and in

RequiredStandbyCapacity(y) ≤ *StandbyCapacity(y)* for the standby part.

- Only one SU, *x*, is assigned the active role for all SIs and only one SU, *y*, is assigned the standby role for all SIs, and they are different.

Having assumed that all SUs in the SG are identical, the properties specified by Equation 2 will be satisfied by a configuration, if and only if the SG consists of at least two SUs and anyone of these SUs is capable of taking the active or the standby role for all SIs. These necessary and sufficient conditions, summarized by Equation 3, can be checked easily.

$$\begin{aligned}
&\forall x, y \in SUSet, \text{such as } x \neq y,\\
&(\exists\ ActiveAssignment\ =\ \{(x,u)\ |\ u \in SISet\} \text{ with}\\
&Range(ActiveAssignment) = SISet\\
&and\\
&(RequiredActiveCapacityFrom(x) \leq\\
&ActiveCapacity(x)))\\
&and\\
&(\exists\ StandbyAssignment\ =\ \{(y,u)\ |\ u \in SISet\} \text{ with}\\
&Range(StandbyAssignment) = SISet\\
&and\\
&(RequiredStandbyCapacityFrom(y) \leq\\
&StandbyCapacity(y)))
\end{aligned}$$

Equation 2. Formal specification of the 2N redundancy model.

$$\begin{aligned}
&|\ SUSet\ | \geq 2, and\\
&\text{let } x \in SUSet,\\
&\bigcup^{CUM} \{RequiredActiveCapacity(si)\ |\ si \in SISet\} \leq\\
&ActiveCapacity(x)\\
&and\\
&\bigcup^{CUM} \{RequiredStandbyCapacity(si)\ |\ si \in SISet\} \leq\\
&StandbyCapacity(x)
\end{aligned}$$

Equation 3. Necessary and sufficient conditions for the 2N redundancy model.

The No-redundancy model is used for non-critical applications and components [2]. An SU is assigned the active HA state for at most one SI. An SI can be assigned to only one SU at a time. All SIs should be assigned if the number of SUs in service permits. An SU is never assigned the standby HA state for any SI. The No-redundancy model is formalized by Equation 4, where *ActiveAssignment* is simply a bijective relation between *SUSet* and *SISet*.

$$\begin{aligned}
&\exists\ ActiveAssignment\ =\ \{(x,y)\ |\ x \in SUSet,\ y \in SISet\ |\\
&(RequiredActiveCapacity(y) \leq ActiveCapacity(x))\\
&and\\
&(\forall\ z \in\ SISet,\ \exists!\ \ (k,z) \in ActiveAssignment)\\
&and\\
&(\forall\ k \in\ SUSet,\ \exists!\ \ (k,z) \in ActiveAssignment)\}
\end{aligned}$$

Equation 4. Formal specification of the No-redundancy model.

Knowing from [2] that any SU in the SG should be capable of protecting any SI that is protected by the SG and assuming this condition, modeled here with $RequiredActiveCapacity(y) \leq ActiveCapacity(x)$, is checked a priori, the only necessary and sufficient condition for an *ActiveAssignment* relation with the specified properties to exist is to have at least as many SUs in *SUSet* than SIs in *SISet*.

IV. SERVICE INSTANCE PROTECTION FOR THE N+M REDUNDANCY MODEL

An SG with the N+M redundancy model has N+M SUs. An SU can be active for all SIs assigned to it or standby for all SIs assigned to it. In other words, no SU can be simultaneously active for some SIs and standby for some other SIs [2]. On the service hand, for each SI there is one and only one SU that is assigned the active HA state and one and only one SU that is assigned the standby HA state.

A. Formal Definition of the N+M Redundancy Model

In order to capture the characteristics of the N+M redundancy model in a precise manner, a formal specification of an SG with the N+M redundancy model is given by Equation 5. As for the case of the 2N redundancy model, we can distinguish two parts for expressing separately the active and the standby assignments.

The 2N and the N+M redundancy models share several properties. In both cases, the SUs can only be either active or standby, and from the service side each SI should only have one active assignment and only one standby assignment. The difference is that for the N+M redundancy model, the number of SUs that are assigned the active HA state or the standby HA state is not limited to one for each. Consequently, in Equation 5, *ActiveAssignment* relation is a relation between a set of SUs and a set of SIs; similarly for *StandbyAssignment* relation. It is well known that the 2N redundancy model is a special case of the N+M redundancy model, i.e. the 2N redundancy model can be identified as the 1+1 redundancy model.

$$\begin{aligned}&\exists\ ActiveAssignment\ =\ \{(x,y)\ |\ x \in SUSet,\ y \in SISet\ |\\ &(\forall\ z \in\ SISet\ \exists!\ \ (k,z) \in ActiveAssignment)\\ &and\\ &(\forall\ w \in\ SUSet,\ RequiredActiveCapacityFrom(w) \leq\\ &ActiveCapacity(w))\}\\ &and\\ &\exists\ StandbyAssignment\ =\ \{(x,y)\ |\ x \in SUSet,\ y \in SISet\ |\\ &(\forall\ z \in\ SISet\ \exists!\ \ (k,z) \in StandbyAssignment)\\ &and\\ &(\forall\ w \in\ SUSet,\ RequiredStandbyCapacityFrom(w) \leq\\ &StandbyCapacity(w))\}\\ &and\\ &Domain(ActiveAssignment) \bigcap Domain(StandbyAssignment) = \varnothing\end{aligned}$$

Equation 5. Formal specification of the N+M redundancy model.

B. Checking SI Protection for an SG with the N+M Redundancy Model

In order to ensure SI protection at configuration when this is not achieved by design, we need to verify the configuration against the specification given in Equation 5. We need a procedure to check for the properties stated in this equation. Such as procedure may have to consider all the possible combinations of SIs to assign to the SUs, and obviously it will be a complex procedure in general. In the case of the 2N redundancy model, there was only one combination of SIs, i.e. SIs are assigned all together to one SU for the active role, and all together to another SU for the standby role.

The complexity of the problem for the case of N+M is due to the different possible combinations of SIs we may have to consider in order to find an *ActiveAssignment* or a *StandbyAssignment* relation that satisfies the properties.

We will here show that the SI protection problem for the N+M redundancy model is an NP-hard problem. Therefore there is no polynomial order algorithm to solve it [7]. According to the NP-hardness theory, a problem *H* is NP-hard if and only if there is an NP-complete problem *L* that is polynomial time Turing-reducible to an instance of *H* [7]. Therefore, in order to prove the NP-hardness of SI protection problem, we have to find an NP-complete problem and reduce it to an instance of the SI protection problem.

For the N+M redundancy model, there is N active SUs and M standby SUs. The active and standby capacities of SUs are independent of each others, since different SUs take these different roles. Consequently, without loss of generality, we will consider here the active part only. The proof for NP-hardness for the active part can be likewise applied for the standby part. If an NP-complete problem reduces to an instance of the SI protection problem in polynomial time, the NP-hardness of the SI protection problem will be established. For this purpose, let us consider the *Subset Sum* problem, which is known to be NP-complete [7].The *Subset Sum* problem can be defined as follows [7]: "*Given a set of positive integers (I) and a positive integer (t), does the sum of some non-empty subset equal exactly to t*?". To prove the NP-hardness of the SI protection problem, let us now consider a specific case in which the number of active SUs is 2 and each SU support only one CST. We refer to this problem as the (*2,1*)-assignment problem. We show the problem is NP-hard in this case; hence NP-hardness of the general SI protection problem. We hereafter, present a reduction of the *Subset Some* Problem to the (*2,1*)-assignment problem.

Theorem 1: The *Subset Sum* problem reduces to the (*2,1*)-assignment problem in polynomial time.

Consider an instance $(I = \{a_1,..........a_p\},t_1)$ of the *Subset sum* problem. Let α be the sum of members of *I*. Define $t_2 = \alpha - t_1$. Observe that for $t_2 < 0$, the answer to the problem is *No* and for $t_2 = 0$, the answer is *Yes*. These are trivial cases. Now, let t_2 be greater than *0* (positive). We need to define an instance of the (*2,1*)-assignment problem. So, we have only two active SUs and the capacity of protected SIs can be represented as positive integers (they can only support one specific CST). Let us define the capacity of SUs as $t_{max} = max(t_1,t_2)$. Also, let the SIs have weights $(a_1,......a_p, \beta)$ in which $\beta = t_{max} - min(t_1,t_2)$ (obviously they consist of CSIs of one CST).

Lemma 1: If the answer to the *Subset sum* problem is Yes, then the answer to the (*2,1*)-assignment problem is also Yes.

Lemma 2: If the answer to the (*2,1*)-assignment problem is Yes, then the answer to *Subset sum* problem is also Yes.

The proof for both lemmas is straightforward. With these lemmas and based on NP-hardness theory, the NP-hardness of (2,1)-assignment problem is proven. Therefore, the NP-hardness of SI protection problem for the N+M redundancy model is established.

V. THE NWAYACTIVE AND NWAY REDUNDANCY MODELS

An SG with the NWayActive redundancy model contains N SUs. Each SU has to be active for all SIs assigned to it. An SU is never assigned the standby HA state for any SI. From the service side, for each SI, one, or multiple SUs can be assigned the active HA state according to the preferred number of assignment configured for the SI.

For the NWay redundancy model, the SG also contains N SUs that protect multiple SIs. An SU can simultaneously be assigned active HA state for some SIs and standby HA state for some other SIs. At most, one SU may have the active HA state for an SI, but one, or multiple SUs may have standby HA state for the same SI. No SU is assigned active HA state and standby HA state for the same SI.

The issue of considering different combinations of SIs remains the same as for the N+M redundancy model. Moreover, the NwayActive and NWay redundancy models allow for multiple assignments of SIs to SUs. Therefore the SI protection problem for both of them is at least as complex as for the N+M redundancy model. Similar proof of NP-hardness can be conducted for these two redundancy models. Therefore, the SI protection problem for the NWayActive and NWay redundancy models is also NP-hard.

VI. OVERCOMING COMPLEXITY

We will explore here how to reduce the complexity of the SI protection problem in the case of the N+M redundancy model. Let us consider the case where *SISet* can be partitioned into subsets of identical SIs and the SIs of any pair of different subsets do not have any CST in common. We refer to this as the case of CST_Disjoint subsets of identical SIs. More precisely, *SISet* can be partitioned into $SISet_1$, $SISet_2$, …, and $SISet_n$, where each $SISet_i$ contains only identical SIs and $SISet_i$ and $SISet_j$ do not have any CST in common when $i \neq j$.

For the N+M redundancy model, any SU in the SG can either be assigned the active or standby HA state. From the service perspective, for each SI, we only have one active assignment and one standby assignment. Consequently, we can divide the set of SUs into two partitions: the active and standby partitions. Any SU in the active partition acts only as active and any SU in the standby partition acts only as standby.

We assumed that SUs in an SG are all identical, which means they all have the same number of components of the same component types. We have so far defined and discussed the capacity in terms of CSTs, we will here define another capacity for an SU with respect to SIs as the number of SIs that the SU can provide service for at the same time. In fact, each SU can have an active capacity and a standby capacity with respect to each SI. We determine the active and standby capacity of an SU with respect to each SI using the division operation introduced in Section 2 as given by Equation 6.

$$\boxed{\begin{array}{l}
\mathit{Integer\ c\ :ActiveCapacity(su{:}SU,si{:}SI)}\\
\mathit{Let\ DivisionSet{:}CSet\ =\ ActiveCapacity(su)\ \ div\ \ RequiredActiveCapacity(si)}\\
\mathit{Let\ Cap\ :\ Set(Integer)} \mid \forall \mathit{(x,y)} \in \mathit{DivisionSet},\ y \in \mathit{Cap\ and} \mid \mathit{Cap} \mid \leq \mid \mathit{DivisionSet} \mid\\
\mathit{c = Min(Cap)};\\
\mathit{Integer\ c\ :StandbyCapacity(su{:}SU,si{:}SI)}\\
\mathit{Let\ DivisionSet{:}CSet\ =\ StandbyCapacity(su)\ \ div\ \ RequiredStandbyCapacity(si)}\\
\mathit{Let\ Cap{:}Set(Integer)} \mid \forall \mathit{(x,y)} \in \mathit{DivisionSet},\ y \in \mathit{Cap\ and} \mid \mathit{Cap} \mid \leq \mid \mathit{DivisionSet} \mid\\
\mathit{c = Min(Cap)}
\end{array}}$$

Equation 6. Active/Standby capacity of an SU w.r.t. to an SI.

The set of protected SIs, *SISet*, is partitioned into CST_Disjoint subsets of identical SIs. By calculating the capacity of one SU for one of the SIs of each partition we will have the capacities of any SU in the SG regarding any SI in the *SISet*. We know that $SISet = SubSet_1 \bigcup SubSet_2 \bigcup ....SubSet_n$, and each $SubSet_i$ is CST_Disjoint with the other subsets. Consequently, we can define an ordered set of *n* integers for an SU in the SG: $\{ac_1, ac_2, ...ac_n\}$, in which ac_i represents the active capacity of the SU with respect to the SIs in $SubSet_i$. Similarly, we define a set of integers for each SU in the SG as $\{sc_1, sc_2, ...sc_n\}$, in which sc_i represents the standby capacity of the SU with respect to the SIs in $SubSet_i$. Now, we have all required information in order to check whether an SG with the N+M redundancy model is capable of protecting the set of SIs it is configured for, or not.

As mentioned earlier, in the N+M redundancy model, we have N SUs and M SUs that are taking the active assignments and standby assignments, respectively. From the service perspective, *SISet*, the set of protected SIs, is partitioned into *n* CST_Disjoint subsets of identical SIs. In this specific situation, the conditions specified in Equation 7 represent the necessary and sufficient conditions for the SG to protect the set of SIs it is configured for.

$$\boxed{\begin{array}{l}\forall 1 \le i \le n,\ \ ac_i \times N \ge \ |SubSet_i| \\ and \\ \forall 1 \le i \le n,\ \ sc_i \times M \ge \ |SubSet_i|\end{array}}$$

Equation 7. Necessary and sufficient conditions for the N+M redundancy model.

Intuitively, $\forall 1 \le i \le n,\ ac_i \times N \ge \ |SubSet_i|$, states that there is enough capacity in the SUs of the SG to protect all the SIs in $SubSet_i$, each SI once. Since the $SubSet_i$ are CST_Disjoint with each other, each SU will be able to provide service for all the subsets simultaneously. The same reasoning applies for the standby part. Moreover, the last property of the N+M redundancy model is satisfied, since we handle active and standby SUs separately. A simple procedure can be written for checking the conditions in Equation 7. Similar conditions and reasoning can be followed for the NWayActive and NWay redundancy models.

VII. CONCLUSION

In this paper, we discussed the complexity of the problem of deciding for SI protection for an AMF configuration. To the best of our knowledge, except for the traditional work on complexity [7], we are not aware of related work tackling SI protection and taking into account the specificities of the domain as specified in the AMF standard [2].

The work has been done under the reasonable assumption that all SUs in the SG are identical. In the case of the 2N redundancy model and the No-redundancy model, we have identified necessary and sufficient conditions that can be checked with simple algorithms. However, the problem is in general NP-hard for the N+M, the NWayActive, and the NWay redundancy models. For overcoming this complexity, due mainly to the consideration of all possible combinations of SIs to assign to SUs, we have characterized a special set of SIs, where necessary and sufficient conditions have been defined and can be checked with simple algorithms.

If we remove the assumption of identical SUs in the SG, the problem remains tractable for the 2N and the No-redundancy models because of their respective specificities, and it remains NP-hard for the other redundancy models. Moreover, the solution, discussed in Section 6, for the special set of SIs remains valid only in the case of the N+M redundancy model.

Checking for SI protection is important for the validation of configurations for compliance with the AMF specification. However, we believe that ensuring the compliance to the standard by construction is the best approach, as we have presented solutions in [5, 6].

ACKNOWLEDGEMENTS

This work has been partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and Ericsson Software Research.

REFERENCES

1. Service Availability Forum™, URL: http://www.saforum.org
2. Service Availability Forum, Application Interface Specification. Availability Management Framework SAI-AIS-AMF-B.04.01.
3. A. Gherbi, P. Salehi, F. Khendek, A. Hamou-Lhadj, "Towards a UML Profile for AMF Configurations Modeling and Analysis", Technical Report , ECE dept, Concordia U., Jan. 2009,
4. OMG, Unified Modeling Language (OMG UML), Infrastructure, V2.1.2, OMG Document Number: formal/2007-11-04, 2007.
5. A. Kanso, M. Toeroe, F. Khendek, A. Hamou-Lhadj, "Automatic Generation of AMF Compliant Configurations", Proceedings of the International Symposium on Service Availability (ISAS), Tokyo, Japan, May 19-21, 2008, Lecture Notes in Computer Science, Vol. 5017, pp.155-170.
6. A. Kanso, M. Toeroe, A. Hamou-Lhadj, F. Khendek, "Generating AMF Configurations from Software Vendor Constraints and User Requirements", Proceedings of the Fourth International Conference on Availability, Reliability and Security (ARES), Fukuoka, Japan, March 2009.
7. M. R. Garey and D. S. Johnson, "Computers and Intractability; A Guide to the Theory of NP-Completeness", W. H. Freeman & Co., New York, NY, USA, 1990.