Introducing dynamism to SA Forum cluster

G. Urbanics[1], A. Kovi[2], Z. Egel[2], A. Pataricza[2]

[1] OptXware LLC, 1137 Budapest, Katona J. U. 39., Hungary

[2] Budapest University of Technology and Economics, 1117 Budapest, Magyar Tudósok krt. 2., Hungary

Introduction

As mobile systems are gaining more and more importance in every aspect of life, be it business or everyday use, the need for highly available services appear there too. In fixed infrastructures, the computer clusters have been extensively used to provide such services. For example, a special type of computer clusters, the high availability clusters, are inherently designed to support fault tolerance for applications. This ability makes them look desired for the mobile environment too; however, the increased dynamicity and limited resources impose new challenges that did not exist before.

In this paper, we analyze how a service availability middleware and applications, based on the Application Interface Specification (AIS) of the Service Availability Forum (SA Forum,) can be adapted to tolerate relatively frequent configuration/environmental changes, and to support provision of services in these environments. These results were achieved in the HIDENETS IST-FP6-STREP project[1] funded by the European Union, where SA Forum specifications are used to provide the necessary services in the infrastructure domain, hence the consortium tried to take advantage of them in the ad-hoc (mobile) domain as well.

About Service Availability Forum and AIS

The Service Availability Forum^{TM} (SA Forum) is “a consortium of industry-leading communications and computing companies working together to develop and publish high availability and management software interface specifications. The SA Forum then promotes and facilitates specification adoption by the industry.” [2] The motivation of these specifications is to support rapid deployment of dependable and highly available applications with the use of broadly adopted open standards. These standards aim at delivering “highly available carrier-grade systems using off-the-shelf hardware platforms, middleware, and service applications.”

From our aspect the Application Interface Specification (AIS) is one of the more important parts of SA Forum’s specifications that “defines standard interfaces that allow application developers to write software that is portable across multiple platforms as long as the underlying middleware is compliant with the AIS” [3]. AIS offers interfaces of services that are designed to support highly available applications in a clustered environment. These services concentrate on various parts of application functionality, e.g. messaging, logging, management; and include the Application Management Framework (AMF) what is responsible for the high availability management of the application.

Context definition

Systems running in the fixed infrastructure are designed to satisfy all requirements that hosted applications impose. These requirements can be hardware specific (e.g. disk space, memory, special devices) and software specific (e.g. given network protocol stacks, availability of other applications, libraries.) These requirements are given for mobile (ad-hoc domain) systems as well but there may be no way to satisfy them fully because of limited resources (memory, storage) or simply because of the nature of a requirement. The latter is specific for mobile applications stemming from the fact that cooperation of different nodes is the usual way of carrying out a service.

In our research we expect the mobile applications to provide the following information about themselves: (i) provided services and installed components, (ii) required services and external components and constraints (trust, reliability, service quality…) This information is then stored in the system model and used when peering nodes are selected.

As an example, take the following simplified scenario. We have nodes A and B. A hosts an application that requires GPS services but node A lacks this functionality. Node B provides the GPS service. When A and B get close enough to start cooperation, A realizes that the GPS service is available through B, and thus the application on A can access it. This way, applications with requirements that could not be satisfied can still be provided on specific nodes.

The situation is more complex for highly available (HA) services. Services in the infrastructure domain are considered highly available if their availability is higher than 99.99% (4 nines.); however, defining HA services in the ad-hoc domain is not that simple. To shortcut the discussion, after long debates, we came up with the following definition: “an ad-hoc domain service is considered to be highly available if the service is considered highly available in the infrastructure domain sense whenever the application requirements can be satisfied.” The difference between a best effort and the HA service here is that while the best effort focuses only on the provision of the service, the HA tries to take care for backup possibilities as well.

Problem statement

The most important feature of mobile systems is the continuously changing environment. Mobile devices are moving around, thus, the distance, connection quality and most importantly the connectivity can change every second.

Contrarily current AIS implementations support mainly rather static environments. The dynamic modification and, consequently, the scalability of an AIS cluster significantly depend on the implementation. Additionally, current implementations work as best effort systems based on very simple policies. Best effort because the join of a node to the cluster is based only on whether the node has a valid configuration in the system or not. This kind of behavior does not conform to the needs of mobile environments and applications since there the set of available nodes is always changing and the system configuration cannot be determined in advance.

So the main deficiency from our perspective that AIS implementations have is the lack of ability to autonomously modify cluster configuration with a pretty high frequency. Autonomous in the sense that there is no need for manual adjustment or revision of the configuration but rather it is based on policies that may come from higher level requirements formulated in Service Level Agreements (SLAs) or Quality of Service (QoS) agreements.

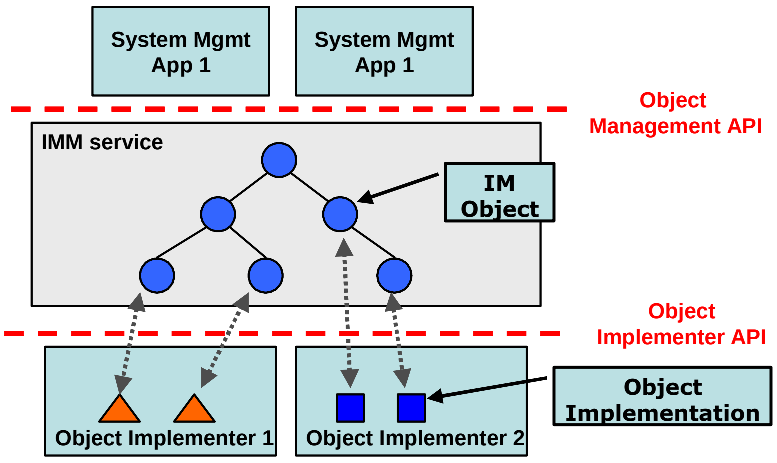

Figure 1: Schematic overview of the IMM.

For the description of an application or a system we use models. In AIS the system model is the Information Model [4]. This contains all the elements that are required to describe a multi node system and component based applications deployed on it. Using elements of the Information Model, both fixed infrastructure and mobile applications can be described in a similar way.

Approach

Considering all the specifics of HA mobile services described above, we found the following features to be important for a system that provides them:

- The nodes have to be self-descriptive. Each node has to be able to tell what services it provides and what others it requires.
- The services have to be uniquely identifiable.
- The node configuration adjustment has to rely on an algorithm that uses the application requirements to set the optimal configuration.
- To ensure high availability, the services have to support role assignments on behalf of different other services.

The SA Forum AIS specifications provide the basic functionality for implementing these features. For the management of the system, the Availability Management Framework (AMF) and the Information Management Service (IMM) should be used. While the AMF controls the state of the services, the IMM handles the configuration management. Using the elements defined in the Information Model, the system model with the applications/services and the nodes can be defined. Then it has to be extended with additional parameters to enable the requirements definition for applications.

Modeling the applications and handling configuration. The AMF is built on the notion of a cluster. The applications run in the cluster and the redundant resources are managed according to predefined policies. The applications are described by their services and the components that provide those. In the forthcoming, these kinds of configuration elements will be called physical elements, and physical configuration, since these are physically available on the node.

In order to meet our needs, we had to extend the configuration handling of the basic AMF in our systems, and besides the physical, we handle the configurations on a virtual level as well. The difference between the physical and the virtual configuration is that the physical one contains only elements that are available in the local node and configurable elements (references) for elements that are expected from other nodes, while the virtual configuration contains the configured elements. Each node has a physical and a virtual configuration, and the virtual configuration belongs to the cluster membership. Consequently, one node may have several virtual configurations if it is a member of several clusters. In our experiments, however, we allowed on one cluster membership per node to simplify the problem.

AMF has different duties on the two levels. On the physical level, AMF manages the availability of components: restarts, stops, performs healthchecks; while on the virtual level it only collects component state information for the configured component references from the other nodes and tries to maintain the required availability of services.

Components are aggregated into service units. The service unit is the unit of redundancy from the perspective of the AMF, which means, it is the smallest unit which can be instantiated in a redundant manner, that is, more than once. Then, service units are aggregated into service groups which handle the comprised service units according to a given redundancy model (e.g. 2N (failover pair), N+M (N active, M standby service units)).

As a consequence of the previous definitions of configuration, the following types of service units will be available:

- Local service unit (Local SU) – services hosted on the local node.
- SU Reference – a logical entity that can be assigned to an available service unit (which can be locally or externally hosted, as well).

- Configured SU Reference – an SU Reference assigned to a service unit.

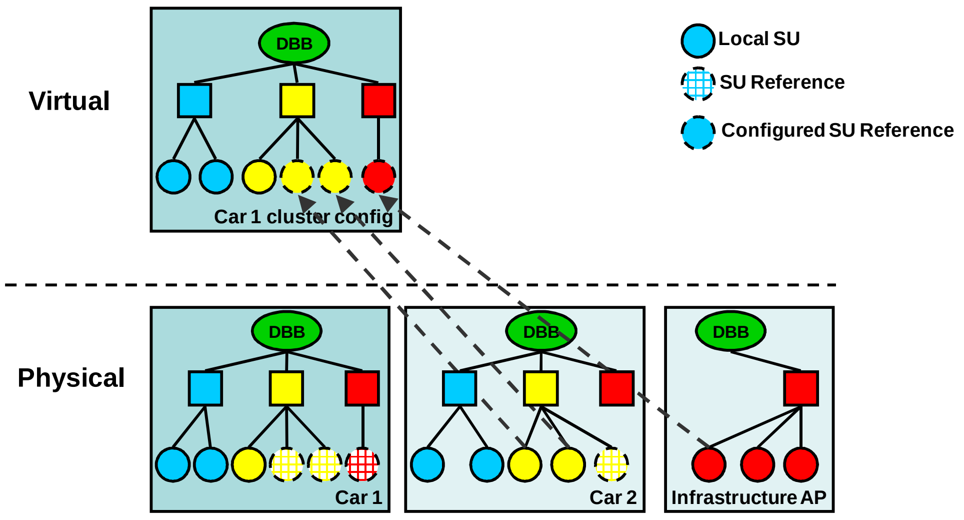

Figure 2: Configuration handling in ad-hoc domain systems

As an example, let us take the Distributed Black Box (DBB) application [1], which aims at creating a low cost, dependable data storage service that can be used for saving important data in mobile environments. Every mobile node contains its storage and offers it to other nodes and they use the services of those vice versa. Whenever the node comes in contact with the infrastructure, it saves all its data there too.

In our example the DBB application contains three types of functionality: (i) the data acquisition, (ii) the ad-hoc data store, (iii) and the infrastructure data store. During operation the configuration is handled both on the physical and virtual levels and Figure 2 shows the connection between those. In the figure we suppose that the cluster of Car 1 contains the node represented by Car 2 and an infrastructure access point. Each of these nodes run the Distributed black box application but with different physical configurations that are combined on the virtual level.The configurations are as follows:

Car 1:

Data acquisition and distribution service (2 service units)
Ad-hoc data store service (1 local and 2 non-configured reference service units)
Infrastructure data store service (1 configurable service unit)

Car 2:

Data acquisition and distribution service (2 service units)
Ad-hoc data store service (2 local and 1 non-configured reference service unit)
Infrastructure data store service (not available)

Infrastructure access point:

Data acquisition and distribution service (not exists)
Ad-hoc data store service (not exists)
Infrastructure data store service (3 local service units)

The figure shows a possible virtual configuration of Car 1 as well. It can be seen that the two reference SUs in the ad-hoc data store service group are configured to use the two local SUs of Car 2 in the same service group and that one of the SUs provided by the infrastructure is assigned to the reference SU in the infrastructure service group.

The assignments of the local service units to reference service units are handled by the node configuration algorithm. This algorithm is something that is not defined in the specifications and highly depends on the middleware implementations. To meet our needs, the selection algorithm should consider many factors for selecting the optimal configuration: application preferences, local and remote resource capacities, security measures, trustworthiness, etc. Finally, most importantly, the algorithm has to observe the availability of provided services and has to pursue configurations where services are provided in a redundant way, thus, failure or miss of one component does not affect the availability of the service.

Experiments

To show that the theoretical results described here are realizable, we created a demonstrator. Owing to the fact that the proposed system is quite complex, we decided to implement only a part of it.

Case study. Our case study involves an AIS cluster in which the nodes are able to move and to join or leave the cluster. Each node can supply information about its provided and required services in a form corresponding to AMF system model, which also includes the unique identification of these services.

In the case of joining, the new node registers itself at a central node of the cluster, the controller node, and they can exchange information about the provided and required services. After a successful joining the controller node notifies every former cluster member about the changes in the configuration. If any of the members requests it, the controller can associate the corresponding nodes.

Technology base for the implementaiton. As the first step of the work, we had to analyze solutions available today. We found that an AIS cluster is considered as a statically configured cluster rather than a dynamic one in terms of the aforementioned aspects. Although minor modifications of the cluster configuration are supported, there are no means of seamlessly adjusting it based on higher level rules or predefined policies. Autonomous and built-in mechanisms aiming at this functionality are completely missing from the specifications and the available implementations, too.

The Availability Management Framework (AMF) is the software entity in an AIS cluster which is responsible for managing the availability of the applications deployed in the cluster through coordination of available hardware and/or software resources. AMF provides a view of one logical cluster using a complex metamodel (defining the logical entities which AMF can operate on, and the operations that can be performed on them) to describe its components and the relationships among them. All these are typically pre-configured and stored in a configuration repository. According to the current release of the specifications “the access and modification of the configuration repository is provided by the Object Management interface” and SNMP MIBs. (For a more exhaustive description see [5].)

Today there exist various possible ways to make configuration adjustments in an AIS cluster. An AIS Service, the Information Model Management (IMM) was defined to be in charge of creating, accessing and managing the objects representing the whole system. The system model can also be modified by means of the Software Management Framework (SMF). It aims at controlling and executing the “migration from one configuration to another” [6]. But the implementation of both IMM and the SMF is very limited due to the fact that they are pretty “young” specifications.

The third way of configuration management is the Simple Network Management Protocol (SNMP). SNMP collects and stores management information about the Managed Objects. The specific information of interest is defined by the Management Information Base (MIB) which is considered to be the information model of SNMP. The MIBs representing the information about an AIS cluster are defined in [7]. They describe the same logical entities as the Information Model, and the relationships amongst them, i.e. the entire system model in a per service manner: each AIS Service has a separate MIB module which contains all the information pertaining to the given service.

Implementation and experience. In the demonstrator we tried to simulate the mobile environment with preconfigured nodes joining and leaving the cluster and assigning workloads to components according to current needs. We used SNMP interfaces to carry out the administrative operations and laptop computers with wireless connection to simulate the nodes.

Our experiments showed that the join and leave of nodes to and from the cluster and the assignment of workloads to different components at runtime are possible with today's implementations; however, since these systems were designed with much less dynamicity in mind, the frequency of these changes had to be held at a quite low rate. The low rate of changes meant that we could simulate only a few, 2-3 changes per minute, otherwise the cluster became instable. In a real mobile environment this number can easily achieve the 20 or more depending on the type of traffic that the device is in.

We did not consider any trust and security measures which may further complicate the implementation; however, it is obvious that these attributes are of absolutely high importance in a mobile environment. Two Hidenets services, the authentication and the trust and cooperation [10] may address some of the occurring problems in this field, but due to the complexity of this area, we decided to not consider these in our experiments.

Conclusion

In this work, we presented our research on the introduction of dynamic behavior to SA Forum AIS based clusters. We defined the problem context through a real life like application and identified the most important aspects that current solutions fail to serve. Then described our approach to solve the problems and finally demonstrated our results.

As this work only intends to give a high level overview of this research topic, there are plenty of details that have to be elaborated in the future. For example:

- Identification of services. A method has to be worked out to support the efficient fast identification of services since this is essential for deciding whether a node is highly demanded in the cluster or if it should rather be kept from joining.
- The algorithm that modifies the cluster configuration based on application requirements needs to be elaborated.
- Security aspects (role changes, trust) has to be taken into consideration in node-node or application interactions.
- The approach described in this paper shows certain similarities with virtualizations technologies, thus, it may be useful to look at our results from this perspective too.

References

[1] M. Radimisch et al., "D1.1 Use case scenarios and preliminary reference model", 2006. Sep, http://hidenets.aau.dk/GetAsset.action?contentId=843068&assetId=1380202

[2] Service Availability Forum^{TM} Homepage - http://www.saforum.org/

[3] Service Availability Forum^{TM}
Service Availability Interface Overview SAIOverview-B.04.01.

[4] Service Availability Forum^{TM} - Information Model Management Service, SAI-AISB.01.02, February 2006.

[5] Service Availability Forum^{TM} - Application Interface Specification Availability Management Framework SAI-AIS-AMF-B.03.01.

[6] Service Availability Forum^{TM} - Application Interface Specification Software Management Framework SAI-AIS-SMF-A.01.01.

[7] Service Availability Forum^{TM} - Distributed Systems Management for AIS-SNMP SAI-AIS-SNMP-A.01.01, 2005.

[9] A. Casimiro, et al., “D2.3 Service level resilience solutions for the ad-hoc domain”, 2008 June, http://hidenets.aau.dk/GetAsset.action?contentId=843068&assetId=3718208

[10] J. Arlat, M. Kaaniche, et al., “D1.2. Revised reference model”, 2007 June, http://hidenets.aau.dk/GetAsset.action?contentId=843068&assetId=2334506