Agent-Based Distributed ARC Programming

R. K. Joshi, Harikrishnan. C. R.
Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Email: {rkj, harikcr}@cse.iitb.ac.in
T. Vamsi Kalyan.
Veritas Software India Pvt. Ltd.
Email: tvamsika@vxindia.veritas.com

May 24, 2004

Abstract

Anonymous Remote Computing (ARC) for C# over .NET supports development of distributed and parallel programs in presence of object mobility. This paper presents deployment view of ARC over LAN, explains development of agent-based applications on ARC framework with simple example programs. Observations on writing agent-based programs using ARC are discussed, it compares round trip time between two hosts of an object for ARC framework with IBM's Aglet system. Evaluations have shown that code motion component of ARC makes it to down perform than Aglets. Paper concludes by mentioning directions on future work.

1 Introduction

Agent-based programming follows a new approach for devising solutions to numerous distributed applications of interest. It extends object-oriented programming by treating objects as autonomous entities, which may be active in a separate thread of execution. These objects can hop between multiple hosts in a network. Agent-based solutions address different kinds of problems [5] with single approach and offer various individual advantages [1] than their alternative solutions. This paper takes common set of all definitions of *mobile agent* and calls a migrating object as mobile agent, which carries state of the object and its associated code while migrating.

ARC framework for C# over .NET [7] is a service oriented framework built on .NET technology [6]. It follows ARC suite, built using C based RPC services over Linux workstations [9]. It offers various services to programmers to develop distributed or parallel programs in presence of heterogeneity, load and failures. Services include, object migration and object retraction. These migratable objects supported in ARC remain addressable to originator application as well as to remote contexts while they roam on the network. ARC supports multi-hopping of an object over a network of machines.

2 Related Work

Much work on mobile agent platforms and agent-based programming has been done over the past ten years. Many research and commercial agent platforms like IBM's Aglets [4] and Voyager [10]

support agent-based programming. Aglets and Voyager are popular commercial platforms used for developing applications involving agents. These are 100% Java based platforms.

Sl No.	Service	Aglets	Voyager	ARC
1.	Creation	yes	yes	yes
2.	Migration	yes	yes	yes
3.	Retraction	yes	no	yes
4.	Activation and Deactivation	yes	yes	yes
5.	Messaging	yes	yes	yes
6.	Load Characterization	no	yes	yes
7.	Multi-hopping	yes	yes	yes
8.	Asynchronous Return of Object	no	no	yes
9.	Disposal	yes	yes	no
10.	Object Arrival Intimation Service	yes	yes	yes
11.	Multi-casting	no	yes	no
12.	Remote Agent Creation	yes	yes	no
13.	Security	yes	yes	no
14.	Proxy Update on Migration	yes	yes	yes

Table 1: List of Commonly Used Services in Agent-Based Programming

Agent-based ARC programming uses .NET technology exclusively. It uses .NET Remoting [8] as means for communicating with ARC services. Table 1 compares ARC framework with Aglets and Voyager in terms of services that are commonly provided for agent-based programming, it is not a complete list of services.

3 ARC Framework Deployment View

Figure 1 shows deployment view of ARC software over the LAN. A node in the LAN may join ARC network to give its computational resources to other nodes present in ARC network. Joined nodes form a logical ring to handle fail stop failures in ARC network. Nodes in the ARC network contain ARC software components that provide different ARC services accessible to local as well as to remote contexts through .NET remoting infrastructure over TCP.

Distributed applications reside in top most layer of ARC framework architecture [7]. These applications written on top of ARC platform may use provided interfaces of ARC service components at a node, whose proxies can be obtained using .NET remoting. These applications may make use of one or more systems present in ARC network.

4 Simple Agent-Based Example Programs

Application development process over ARC is discussed in [7]. It has mainly following three steps; rest of this section explains two simple example programs by describing these three steps

1. Specification of ARC object interface.
2. Implementation of ARC object interface and method, *Trigger()*.

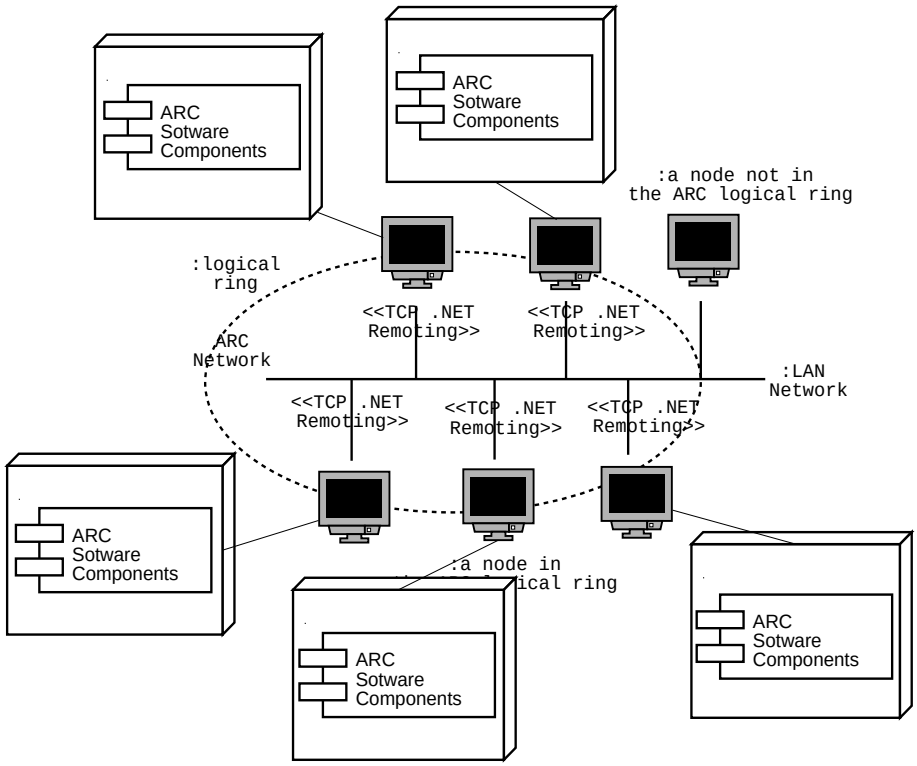

Figure 1: Deployment View of ARC Software

3. Using the ARC object implementation in an application.

4.1 Distributed Copy Paste Application

Distributed copy paste application is useful in copying text at one machine and pasting it at another machine. It is especially very useful in a cluster, where all nodes are connected using a CPU switch.

An ARC object stores text in it and acts as messenger between originator and remote machine. Application uses Object Arrival Intimation Service implemented in component, *NameServer* of ARC framework to get an intimation when an ARC object, containing text arrives as local host. It registers a request with *NameServer* at startup for an intimation of arrival of *copy paste* ARC object.

Application uses ARC framework to transmit text by wrapping it in *copy paste* ARC object. The object migrates to a remote machine and calls method, *RegisterForRemoteAccess* to register a proxy of it to enable access to its interface from remote environments. It also registers a proxy with *NameServer* to allow applications, which have registered a request at *NameServer* access its interface. *Copy paste* application running at remote machine extracts the text from arrived object upon receiving an intimation from *NameServer* and displays the text. Originator machine uses *connect* construct to subsequently read and modify the copied text from migrated object. Various steps of the application development are described below.

4.1.1 Interface Specification

Following interface is used in copy paste application to create an ARC object. It consists of public member functions that *copy paste* ARC object exports to applications for use.

```
public interface ICP{
  Object getData();
  void setData(Object data);
  void setDataAndNotify(Object data);
  void RegisterApp(IUI app);
}
```

4.1.2 Implementation of the Interface and Trigger

Interfaces *ICP* and *ITrigger* are implemented in class, *Real*. A proxy to the ARC object is registered from method, *Trigger*. The proxy is registered at current local system to allow access to it from remote environments as well as from local context. *Trigger* method registers a proxy at the *NameServer* and blocks the thread. Following is the code snippet of class, *Real*.

```
   public class Real:PReal,CPObjectInterface.ICP,ITrigger {
   private Object Lock = new Object();
   public void Trigger(){
     RegisterForRemoteAccess();
     ARCObjectInterface_NamingService.IARCObject nameserver = null;
     if(this.Proxy != null){
         Object proxy = this.Proxy;
         nameserver = (ARCObjectInterface_NamingService.IARCObject)
                         Activator.GetObject(typeof(
                         ARCObjectInterface_NamingService.IARCObject),
                              "tcp://localhost:9123/NameServerClass");
         nameserver.Register("CopyPasteARCObject",proxy);
      }
      Monitor.Enter(Lock);
      Monitor.Wait(Lock);
      Monitor.Exit(Lock);
      UnRegisterForRemoteAccess();
      if(nameserver != null)
         nameserver.UnRegister("CopyPasteARCObject");
   } // end of Trigger method

   ... //implementation of interface, ICP

} //end of class definition
```

4.1.3 GUI of Distributed Copy Paste Application

Figure 2 shows the GUI of *copy paste* application. The application registers a request for *copy paste* ARC object with *NameServer* at startup. It gets the notification when ARC object registers a proxy with the *NameServer* from method, *Trigger*.

Figure 2 has string, "test string" in one of the Textboxes. When button labeled Copy/Modify located immediately next to "test string" is clicked for the first time, an ARC object is instantiated. String in the Textbox is stored into the *copy paste* ARC object and sent to destination machine.

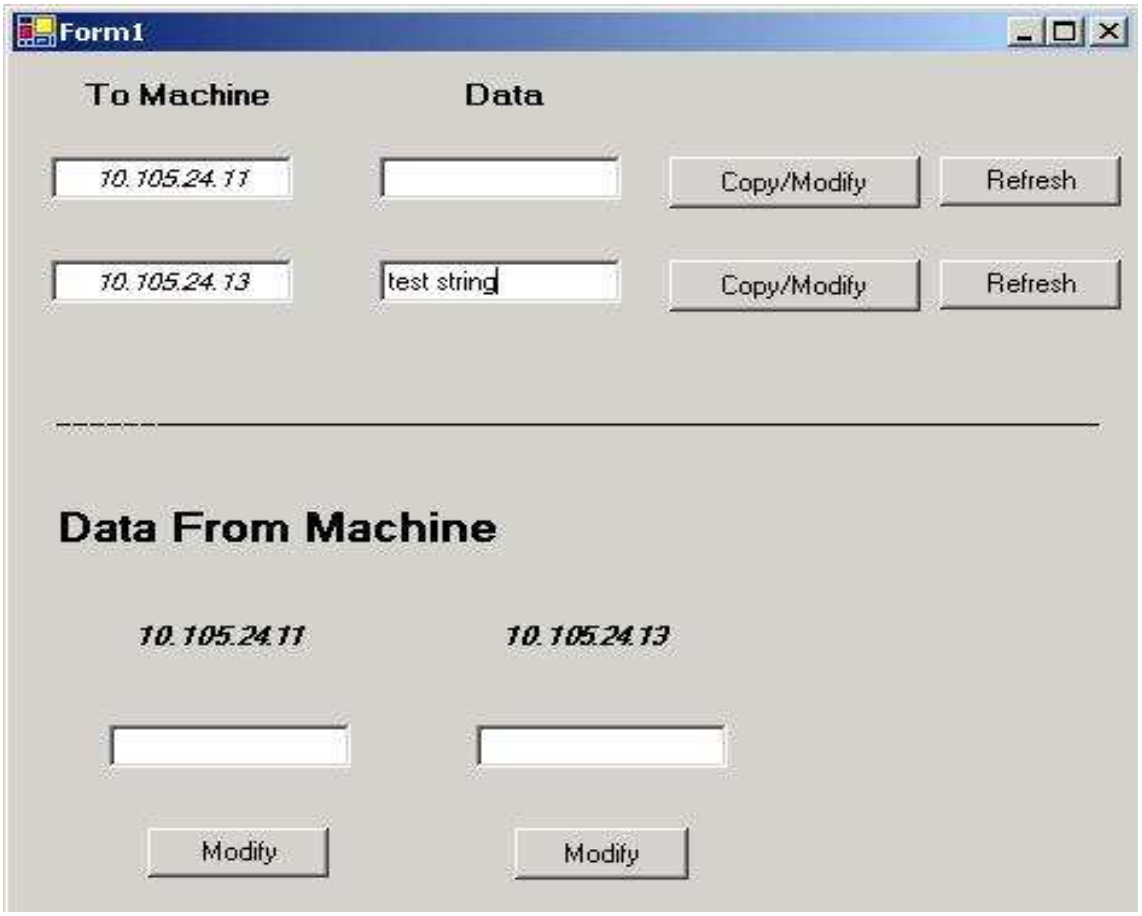

Figure 2: Copy Paste Application Interface

Remote *copy paste* application gets a notification from *NameServer* after object arrives at that node. Application extracts the string from ARC object by using ARC object interface and displays it. *Copy paste* application at sender machine may connect to previously sent ARC object by calling method, *connect* on ARC object. Clicking on *Refresh* button at originator machine connects to the sent ARC object and displays current value of string. Any modifications to the string data from original sender machine requires a click on *Copy/Modify* button and any modification to the string data from remote environment requires a click on *Modify* button.

4.2 Tic Tac Toe Game Using ARC

Tic Tac Toe game developed using ARC uses multiple machines. Here, players can play the game from different machines in a network. Game window can be sent to a machine from where the player is playing the game. ARC object is used to carry state of the game from one machine to the other. Figure 3 shows working of the game with screenshots taken when played from two machines in a network. Game window has provision for entering machine address to where it should go for next move in the game. Using this feature, same game window can be migrated among more than two machines. It demonstrates multi-hopping of an ARC object. Rest of this section describes various steps involved in developing the application.

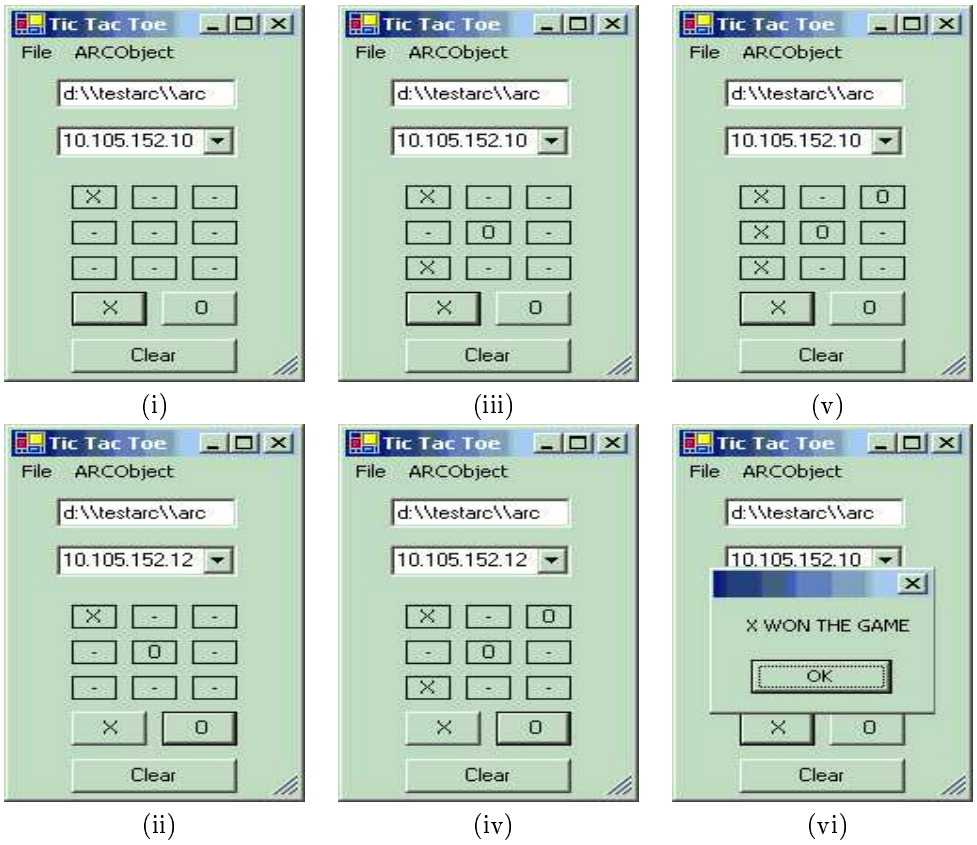

Figure 3: Working of Tic Tac Toe Using ARC

4.2.1 Interface Specification

Following interface is used in Tic Tac Toe application. It contains a single method, *startGame* to start the game.

```
public interface IT3{
 void startGame();
}
```

4.2.2 Implementation of the Interface

Implementation of method *Trigger* consists a statement to show the game window. Player enters next move, selects a machine address to send the game window to it and selects push operation from menu, ARCObject. *Trigger* then acquires HPF value on selected machine and pushes the object to selected machine for next move in the game.

```
public class Real:PReal,T3object.IT3,ITrigger
 public void Trigger(){
   this.ShowScreen();
   UserInterface_HPFServer.IUser hpfserver = (UserInterface_HPFServer.IUser)
                    Activator.GetObject(typeof(UserInterface_HPFServer.IUser),
                          ''tcp://localhost:9123/HPFServerClass'');
   UserInterface_HPFValue.IUser hpfv;
   hpfv = hpfserver.getHPFValue(machn);
   this.push(hpfv);
 }

 ... implementation of IT3 ...
}
```

4.2.3 Tic Tac Toe Application Development

Application code registers a channel to communicate with local ARC kernel. It then creates a new ARC object and invokes method, *startGame* on it. Following is the code snippet showing the application program.

```
using T3objectGeneratedInterfaces;
using T3objectNamespace;
public static void Main(){
  TcpChannel chan = new TcpChannel(8106);
  ChannelServices.RegisterChannel(chan);
  IT3GeneratedInterfaces.IContainer arcobject = Factory.New();
  arcobject.StartGame();
}
```

5 Observations

Many researchers have agreed till now that mobile agent technology does not solve any problem that otherwise cannot be solved using alternative techniques such as RPC [2], CORBA [3], .NET Remoting [8] *etc.* We used ARC framework to write agent-based solutions for different distributed/parallel

problems like, algorithms such as Parallel Quick Sort, useful applications such as Distributed Copy Paste, work flow implementations and games such as Tic Tac Toe. Our observation is that developing programs using ARC services solve many problems with a single approach.

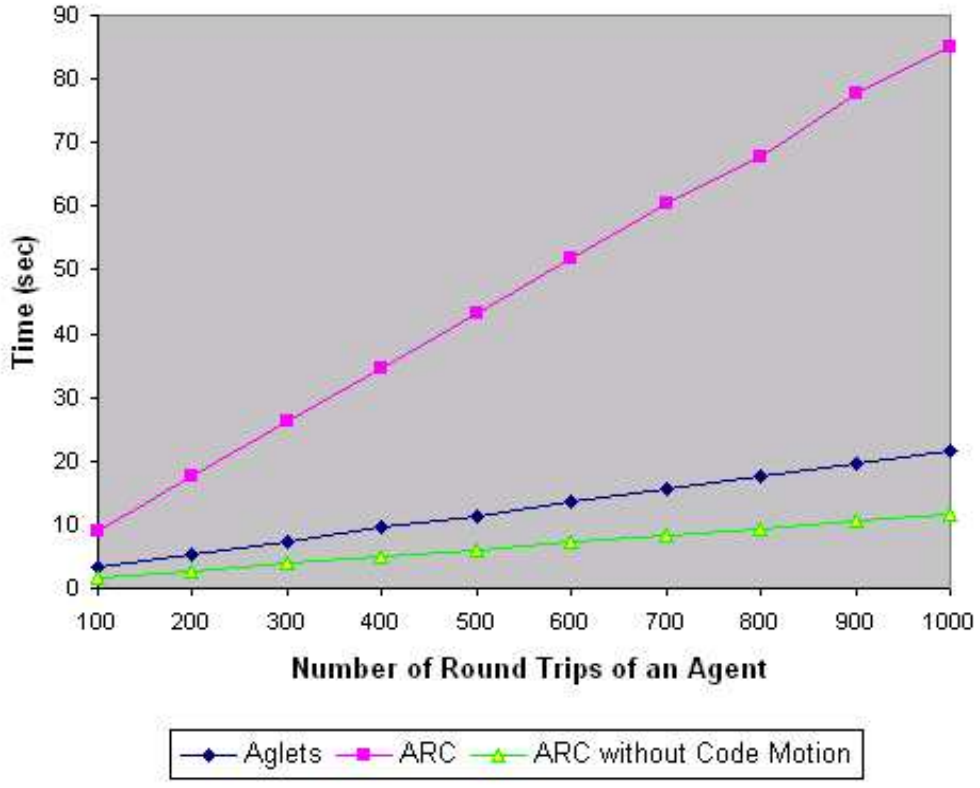

Figure 4: Communication costs of Aglets, ARC and ARC without Code Motion

Figure 4 compares performance of ARC with IBM's Aglets. Performance metric used for comparison is time taken to finish n round trips between two hosts. Test was done using two Windows based workstations connected through 100 Mbps switch. Results have shown that ARC takes more time than Aglets due to disk I/O involved in class mobility. When required classes are kept manually at remote machine and removed the use of *CodeMotionServer* component of ARC framework, which is used to transfer associated code of an object, ARC performed better than Aglets.

6 Conclusion

ARC deployment view was presented along with simple agent-based example programs written on top of ARC platform. Performance of ARC was compared with IBM's Aglet system. It was observed that devising solutions with ARC services solve many problems under the same approach. We continue our research in this direction to explore the possibility of integrating different agent-based applications in a cluster by using ARC services and communication between agents.

References

[1] C. G. Harrison, D. M. Chess, A. Kershenbaum. Mobile agents: Are they good idea? *Research Report, IBM Research Division*, March 1995.

[2] A. D. Birrel and B. J. Nelson. Implementing remote procedure calls. *ACM Transactions on Computer Systems*, 2(1):39–59, February 1984.

[3] Object Management Group. Common Object Request Broker Architecture Specification. *http://www.omg.org/*, 2002.

[4] Danny B. Lange and Mitsuru Oshima. Mobile Agents with Java: The Aglet API. *World Wide Web Journal*, 1998.

[5] Danny B. Lange, Mitsuru Oshima. Seven good reasons for mobile agents. *Communications of the ACM*, 42(3):88–89, March 1999.

[6] David S. Platt. *Introducing Microsoft .NET*. Microsoft Press, January 2003.

[7] T. Vamsi Kalyan, R. K. Joshi. Architecture of the object oriented anonymous remote computing framework for c# over .net. *International Workshop on Software Design and Architecture*, January 2004.

[8] Ingo Rammer. *Advanced .NET Remoting*. APress, April 2002.

[9] D. Janaki Ram, Rushikesh K Joshi. Anonymous Remote Computing, A paradigm for Parallel Programming on interconnected Workstations. *IEEE Transactions on Software Engineering*, pages 75–90, Jan/Feb 1999.

[10] Thomas Wheeler. Reducing development effort using the voyager orb, 2003. URL: http://www.recursionsw.com/products/voyager/whitepapers/Ease of developement.pdf.