Appendix A

PAIncrementalPlanningCapsule

This ANTS capsule is used by Panda to perform incremental planning. It is issued by the source node, visits the connection node, and returns to the source node, indicating that the incremental plan is deployed and the stream should be unlocked. This functionality is in the *evaluate()* method of the capsule class. The capsule is delivered to every PANDA node on its way to the destination. On its way back, it is automatically forwarded to the source node without visiting any PANDA nodes.

```
package FMG.Panda.architecture.ants.capsules;

import java.util.Vector;
import java.util.Enumeration;
import java.util.Date;

import ants.Capsule;
import ants.ByteArray;
import ants.Node;
import ants.Xdr;

import FMG.Panda.architecture.ants.PAAddress;
import FMG.Panda.architecture.ants.UserProfile;
import FMG.Panda.architecture.ants.capsules.PAAdaptor;
import FMG.Panda.architecture.ants.capsules.PADataCapsule;
import FMG.Panda.architecture.ants.capsules.PlanHeader;


public class PAIncrementalPlanningCapsule extends Capsule
{
    final private static byte[] MID =
            findMID("FMG.Panda.architecture.ants.capsules.
            PAIncrementalPlanningCapsule");

    protected byte[] mid() { return MID; }

    final private static byte[] PID =
            findPID("FMG.Panda.architecture.ants.capsules.
            PAIncrementalPlanningCapsule");
```

```
protected byte[] pid() { return PID; }

// Since we don't have access to UserFlowType (class) we'll mimic
// the FlowType identifiers here
public static final int UDP_FLOW_TYPE = 0;
public static final int TCP_FLOW_TYPE = 1;

public static final boolean PLANNING_PROTOCOL_PRINTOUT = true;

public int src;                // ANTS source
public int dest;               // ANTS destination
public short paPort;           // The Panda Application Port
public boolean planned;        // planned?
public ByteArray sid;          // SessionID (byte[] form)
public int ufType;             // UserFlow Type (int form)
public PlanHeader thePlan;     // The Plan
public UserProfile userProfile;// user profile
public int appType;            // application Type (1,3, or 4)

private ByteArray thePlanBA;
private ByteArray userProfileBA;
public Vector protocolList;
public boolean appFlagDelivery;
public long rtt;
public long planning_register_time;

public int length() {
  Enumeration e;
  int s = super.length();
  String str;

  s += Xdr.INT;                      // src
  s += Xdr.INT;                      // dest
  s += Xdr.SHORT;                    // paPort
  s += Xdr.BYTEARRAY(sid);           // sid
  s += Xdr.INT;                      // ufType
  s += Xdr.INT;                      // appType
  s += Xdr.BOOLEAN;                  // planned
  s += Xdr.BOOLEAN;                  // appFlagDelivery
  s += Xdr.LONG;                     // rtt
  s += Xdr.LONG;                     // planning_register_time
  userProfileBA = new ByteArray(userProfile.toBytes());
  s += Xdr.BYTEARRAY(userProfileBA); // user profile
  thePlanBA = new ByteArray(thePlan.toBytes());
  s += Xdr.BYTEARRAY(thePlanBA);

  // Determine the length of protocolList
    s += Xdr.SHORT;         // Number of elements in interceptList
    e = protocolList.elements();
    while (e.hasMoreElements()) {
        str = (String) e.nextElement();
        s += Xdr.STRING(str);
    }
```

```
    return s;
}

public Xdr encode() {
  Enumeration e;
  Xdr xdr = super.encode();

  xdr.PUT(src);
  xdr.PUT(dest);
  xdr.PUT(paPort);
  xdr.PUT(sid);
  xdr.PUT(ufType);
  xdr.PUT(appType);
  xdr.PUT(planned);
  xdr.PUT(appFlagDelivery);
  xdr.PUT(rtt);
  xdr.PUT(planning_register_time);
  xdr.PUT(userProfileBA);
  xdr.PUT(thePlanBA);

  // Encode the protocolList
    xdr.PUT((short) protocolList.size());
    e = protocolList.elements();
    while (e.hasMoreElements()) {
      String str=(String)e.nextElement();
        xdr.PUT(str);

      if(PLANNING_PROTOCOL_PRINTOUT)
              System.out.println("Adapters :
              "+str.substring(0,str.indexOf(":")));
    }



  return xdr;
}

public Xdr decode() {
  Enumeration e;
  int protocolSize;

  Xdr xdr = super.decode();

  src = xdr.INT();
  dest = xdr.INT();
  paPort = xdr.SHORT();
  sid = xdr.BYTEARRAY();
  ufType = xdr.INT();
  appType = xdr.INT();
  planned = xdr.BOOLEAN();
  appFlagDelivery = xdr.BOOLEAN();
  rtt = xdr.LONG();
  planning_register_time = xdr.LONG();
```

```
  userProfileBA = xdr.BYTEARRAY();
  userProfile = new UserProfile(userProfileBA);
  thePlanBA = xdr.BYTEARRAY();
  thePlan = new PlanHeader(thePlanBA);

  if(PLANNING_PROTOCOL_PRINTOUT)
      System.out.println("thePlan");

    protocolSize = xdr.SHORT();

  if(PLANNING_PROTOCOL_PRINTOUT)
      System.out.println("protocolSize "+protocolSize);

  while (protocolSize-- > 0) {
      if(PLANNING_PROTOCOL_PRINTOUT)
        System.out.println(protocolSize);
        String str = xdr.STRING();

        protocolList.addElement(str);

    }


    return xdr;
}




public boolean evaluate(Node n) {
   if(PLANNING_PROTOCOL_PRINTOUT)
     System.out.println("PAIncrementalPlanning.evaluate() on " +
                  n.getAddress()+" "+appFlagDelivery);

   if (!appFlagDelivery) {
     appFlagDelivery = true;
     if(PLANNING_PROTOCOL_PRINTOUT)
        System.out.println("PAIncrementalPlanning.evaluate():
        PAIC(not planned yet) is forwarded to"+getDst());

     return n.routeForNode(this, getDst());
   }

   if (getDst() != n.getAddress() && planned) {

     if(PLANNING_PROTOCOL_PRINTOUT)
         System.out.println("PAIC is forwarded to "+getDst());

     return n.routeForNode(this, getDst());
   } else {
     if(PLANNING_PROTOCOL_PRINTOUT)

        System.out.println("PAIncrementalPlanning.evaluate(): PAIC
        (destination "+getDst()+" ) arrived");
```

```
        return n.deliverToApp(this, paPort);
      }
    }

    public PAIncrementalPlanningCapsule(int src,
                                int dest,
                                short paPort,
                                int ufType,
                                byte[] sid,
                                UserProfile userProfile,
                                int appType) {
      this.src = src;
      this.dest = dest;
      setDst(dest);
      this.paPort = paPort;
      this.ufType = ufType;
      this.sid = new ByteArray(sid);
      this.thePlan = new PlanHeader();
      this.userProfile = userProfile;
      this.appType = appType;
      appFlagDelivery = true;
      rtt = (new Date()).getTime();
      planning_register_time = 0;

    }
    // Needed for ANTS to create this capsule
    public PAIncrementalPlanningCapsule() {

      protocolList = new Vector();
    }
}
```

Appendix B

PACentralPlanningCapsule

This ANTS capsule is used by Panda to perform central planning. It is issued by the source node, visits the connection nodes collection planning data, and forwarded by the destination node to the planning site, which is the source node in this case. This functionality is in the *evaluate()* method of the capsule class. The capsule is delivered to every PANDA node on its way to the destination. On its way back, it is automatically forwarded to the source node without visiting any PANDA nodes.

```
package FMG.Panda.architecture.ants.capsules;

//import java.util.Vector;
//import java.util.Enumeration;

import ants.Capsule;
import ants.ByteArray;
import ants.Node;
import ants.Xdr;

import FMG.Panda.architecture.ants.PAAddress;
import FMG.Panda.architecture.ants.capsules.PAAdaptor;
import FMG.Panda.architecture.ants.capsules.PADataCapsule;
import FMG.Panda.architecture.ants.PlanningData;
import FMG.Panda.architecture.ants.UserProfile;

public class PACentralPlanningCapsule extends Capsule
{
    final private static byte[] MID =
      findMID("FMG.Panda.architecture.ants.capsules.
                                    PACentralPlanningCapsule");
    protected byte[] mid() { return MID; }

    final private static byte[] PID =
      findPID("FMG.Panda.architecture.ants.capsules.
                                    PACentralPlanningCapsule");
    protected byte[] pid() { return PID; }

    // Since we don't have access to UserFlowType (class) we'll mimic
    // the FlowType identifiers here
```

```
public static final int UDP_FLOW_TYPE = 0;
public static final int TCP_FLOW_TYPE = 1;

public int src;                // ANTS source
public int dest;               // ANTS destination
public int plannerAddress;      // ANTS planner address
public int storedDestAddress;  // stored Source Address
public short paPort;           // The Panda Application Port
public boolean planned;        // planned?
public int way;                // way of PAC(0, 1,or 2)
public ByteArray sid;          // SessionID (byte[] form)
public int ufType;             // UserFlow Type (int form)
public PlanHeader thePlan;     // The Plan
public PlanningData planningData; // Planning data
public UserProfile userProfile;// user profile
public int appType;            // application type (1,3,or 4)
public long init_time;          // init time

private ByteArray thePlanBA;
private ByteArray planningDataBA;
private ByteArray userProfileBA;
public boolean appFlagDelivery;

public static final boolean PLANNING_PROTOCOL_PRINTOUT = true;

public int length() {

  int s = super.length();

  s += Xdr.INT;                   // src
  s += Xdr.INT;                   // dest
  s += Xdr.INT;                   // Planner Address
  s += Xdr.INT;                   // stored dest Address
  s += Xdr.SHORT;                 // paPort
  s += Xdr.BYTEARRAY(sid);        // sid
  s += Xdr.INT;                    // ufType
  s += Xdr.INT;                    // way (0,1, or 2)
  s += Xdr.INT;                    // way
  s += Xdr.BOOLEAN;                //appFlagDelivery
  s += Xdr.LONG;                   // init time

  userProfileBA = new ByteArray(userProfile.toBytes());
  s += Xdr.BYTEARRAY(userProfileBA); // user profile
  thePlanBA = new ByteArray(thePlan.toBytes());
  s += Xdr.BYTEARRAY(thePlanBA);
  planningDataBA = new ByteArray(planningData.toBytes());
  s += Xdr.BYTEARRAY(planningDataBA);


    return s;
}

public Xdr encode() {
```

```
  Xdr xdr = super.encode();

  xdr.PUT(src);
  xdr.PUT(dest);
  xdr.PUT(plannerAddress);
  xdr.PUT(storedDestAddress);
  xdr.PUT(paPort);
  xdr.PUT(sid);
  xdr.PUT(ufType);
  xdr.PUT(appType);
  xdr.PUT(way);
  xdr.PUT(appFlagDelivery);
  xdr.PUT(init_time);
  xdr.PUT(userProfileBA);
  xdr.PUT(thePlanBA);
  xdr.PUT(planningDataBA);

  if(PLANNING_PROTOCOL_PRINTOUT) {
      System.out.println("XDR encode:");
      System.out.println(src);
      System.out.println(dest);
      System.out.println(paPort);
  }

    return xdr;
}

public Xdr decode() {
  Xdr xdr = super.decode();

  src = xdr.INT();
  dest = xdr.INT();
  plannerAddress = xdr.INT();
  storedDestAddress = xdr.INT();
  paPort = xdr.SHORT();
  sid = xdr.BYTEARRAY();
  ufType = xdr.INT();
  appType = xdr.INT();
  way = xdr.INT();
  appFlagDelivery = xdr.BOOLEAN();
  init_time = xdr.LONG();
  userProfileBA = xdr.BYTEARRAY();
  userProfile = new UserProfile(userProfileBA);
  thePlanBA = xdr.BYTEARRAY();
  thePlan = new PlanHeader(thePlanBA);
  planningDataBA = xdr.BYTEARRAY();
  planningData = new PlanningData(planningDataBA);

  if(PLANNING_PROTOCOL_PRINTOUT) {
      System.out.println("XDR decode:");
      System.out.println(src);
      System.out.println(dest);
      System.out.println(paPort);
  }
```

```
    return xdr;
}

public boolean evaluate(Node n) {
   if(PLANNING_PROTOCOL_PRINTOUT)
     System.out.println("PACentralPlanning.evaulate() on " +
                    n.getAddress()+" "+appFlagDelivery);
   if (!appFlagDelivery) {
     appFlagDelivery = true;
     if(PLANNING_PROTOCOL_PRINTOUT)
        System.out.println("PACentralPlanning.evaluate(): PAC(not
        planned yet) is forwarded to"+getDst());
     return n.routeForNode(this, getDst());
   }

   if (getDst() != n.getAddress() && way == 1) {
     if(PLANNING_PROTOCOL_PRINTOUT)
         System.out.println("PAC is forwarded to "+getDst());
     return n.routeForNode(this, getDst());
   } else {
     if(PLANNING_PROTOCOL_PRINTOUT)
        System.out.println("PACentralPlanning.evaluate(): PAC
        (destination "+getDst()+" ) arrived");

     return n.deliverToApp(this, paPort);
   }
}

public void setPlannerAddress(int plannerAddress) {
  this.plannerAddress = plannerAddress;
}

public PACentralPlanningCapsule(int src,
                        int dest,
                        short paPort,
                        int ufType,
                        int appType,
                        byte[] sid,
                        UserProfile userProfile,
                        PlanningData planningData) {
  this.src = src;
  this.dest = dest;
  setDst(dest);
  this.paPort = paPort;
  this.ufType = ufType;
  this.appType = appType;
  this.sid = new ByteArray(sid);
  this.thePlan = new PlanHeader();
  this.userProfile = userProfile;

  if(planningData == null)
      this.planningData = new PlanningData();
  else
```

```
            this.planningData = planningData;
        appFlagDelivery = true;
        this.way = 0;
    }

    // Needed for ANTS to create this capsule
    public PACentralPlanningCapsule() {
    }
}
```

Appendix C

Adapter Data

This is a simplified example of the adapter description that is necessary for planning. There must be sufficient support for adapter consistency, plan feasibility, etc. Postconditions of the adapter are stored in a stack of adapter postconditions (SAP) that characterizes the data stream (described in Chapter 5).

In this implementation, we counted the resources required to run adapters in am amount equal to the number of adapters that can be run on a node. That is why CPU is always equal 1, which basically means "it costs one adapter to run this adapter." Memory and hard drive resources are not counted here. In reality, more sophisticated resource accountability systems are necessary.

```
#########################################
#Lempel Ziv Compressor
#
name = Compressor
args = 100
problemCode = Throughput
solutionCode = ZEV
binary = true
UNDOname = de_Compressor
UNDOargs = 200
efficiency = 0.5
efficiencyThroughput = 1.0
dataPreservation = 1.0
#
# Required Resource to run the compressor
#
requiredResources: CPU = 1
requiredResources: Memory = 0
requiredResources: HD = 0
#requiredResources: Monetary Cost = 0
#
# Preconditions
#
```

```
preCondition: CompressabilityZEV = 1
#
# Postconditions
#
postCondition: CompressabilityZIV = 0
postCondition: FormatCompressabilityMode = compressedZIV
#
# Resources required to run decompression
#
UNDORequiredResources: CPU = 1
UNDORequiredResources: Memory = 0
UNDORequiredResources: HD = 0
UNDORequiredResources: Monetary Cost = 0
#
# Preconditions to run decompressor
#
UNDOPreCondition: CompressabilityZEV = 0
UNDOPreCOndition: CompressabilityMode = compressedZIV
#
# Postconditions after the decompressor run
#
UNDOPostCondition: CompressabilityZEV = 1
UNDOPostCondition: FormatCompresabilityMode = ""
#
end of adapter
```

Appendix D

Example of a Partial Plan

This is the example of the partial plan that is presented on Figure 5.5. The names of tags are the methods, which are used by adapters to solve the problems of connections. The partial plan is used for the ordering of the actual adapters that use the same methods. The calculation of the partial plan also requires the lists of pre- and post-conditions for the adapters that would use these methods.

```
Format_Conversion: 1-10
Distilling: 1-10
Color_Drop: 1-10
Frame_Drop: 1-10
Quality_Drop: 1-10
LZ Compression: 10-30
Encryption: 30-40
Storage: 0-40
Buffering: 0-40
FEC: 50-60
```

Appendix E

Example of Planning Data

This is the example of planning data collected by the planner on a connection that consists of five nodes. Addresses of nodes 1000 to 1004 are internal ANTS node addresses that are associated with real Internet IP. Address -1 means that it is a link between two connection nodes. In this implementation, the tag "CPU" means the number of adapters that can be executed on the node. Links #3 and #4 have security less than 1.0. If a user application requires complete security equal to 1.0, the planner will use an encryption adapter. Link #3 has available bandwidth of 200 Kbps. If a user application requires more that 200 Kbps, data compression should be applied to this link.

```
#
# Planning data for 5 node connection
# Node 1000
Entity: 1000
Resource: CPU = 1
Resource: HD = 100
Resource: Memory = 100
#Resource: Monetary Cost = 100
#
# Link # 1
-1
Reliability = 1
Security = 1.0
Authentication = 1
Throughput = 7000
Buffering = 1
#
# Node 1001
Entity: 1001
Resource: CPU = 1
Resource: HD = 100
Resource: Memory = 100
#Resource: Monetary Cost = 100
#
```

```
# Link # 2
-1
Throughput = 1000
Security = 1.0
Authentication = 1
Reliability = 1
Buffering = 1
#
# Node 1002
Entity: 1002
Resource: CPU = 6
Resource: HD = 100
Resource: Memory = 100
#Resource: Monetary Cost = 100
#
# Link # 3
-1
Throughput = 200
Security = 0.8
Authentication = 1
Reliability = 1
Buffering = 1
# Node 1003
Entity: 1003
Resource: CPU = 10
Resource: HD = 100
Resource: Memory = 100
#Resource: Monetary Cost = 100
#
# Link # 4
-1
Throughput = 10000
Security = 0.5
Authentication = 1
Reliability = 1
Buffering = 1
#
# Node 1004
Entity: 1004
Resource: CPU = 1
Resource: HD = 100
Resource: Memory = 100
#Resource: Monetary Cost = 100
end of data
```

Appendix F

Example of User Preferences and Stream Characteristics

This is the example of stream characteristics and user preferences. In this example, the data stream requires 2000 Kbps, and if the bandwidth is not sufficient, the color of the stream can be dropped. The user application here requires full security, but the stream should not be encrypted according to one of the postconditions of the user application.

```
#
# Stream Resource Requirements / User preferences on the methods of
# adaptations
#
Throughput = 2000/Color_Drop
#
Security = 1/Encryption
#
Reliability = 1/FEC
Buffering = 1
#
#Monetary Cost = 1
# Postconditions of a user application
#
Conditions: CompressabilityZEV = 1
Conditions: CompressabilityColor_Drop = 1
Conditions: Security = 0
# end
end of data
```

References

[Balakrishnan95] H. Balakrishnan, S. Seshan, E. Amir, and R. Katz, "Improving TCP/IP Performance Over Wireless Networks," *Proceedings of the 1st ACM International Conference on Mobile Computing and Networking (MobiCom'95)*, November 1995.

[Bhattacharjee97] Samrat Bhattacharjee, Kenneth L. Calvert, Ellen W. Zegura, "Active Networking and End-to-End Argument," *Proceedings of 1997 International Conference on Network Protocols*, Los Alamitos, CA, 1997, pp. 220-228.

[Braden01] Bob Braden et al., "Active Reservation Protocol (ARP) - Lessons Learned," Presentation to DARPA Active Networks PI meeting in Orlando, Florida, Dec 3-5, 2001.

[Bretthauer95] Kurt M. Bretthauser, Murray J. Côté, "Nonlinear Programming for Multiperiod Capacity Planning in a Manufacturing System," *European Journal of Operational Research*, vol. 96, 1996, pp. 167-179.

[Bush99] S. F. Bush, "Active Virtual Network Management Protocol," *Proceedings Thirteenth Workshop on Parallel and Distributed Simulation, PADS 99*, Los Alamitos, CA, 1999. pp. 182-192.

[Calvert98] Ken Calvert, "Architectural Framework for Active Network," Active Networks Architecture Documents, http://www.darpa.mil/ito/research/anets/Arcdocs.html

[Choi00] Sumi Choi, J. Turner, and T. Wolf, "Configuring sessions in programmable networks," *Proceedings IEEE INFOCOM 2001*, Anchorage, AK, USA, 22-26 April 2001, vol. 1, pp. 60-67.

[Clark92] David D. Clark, Scott Shenker, and Lixia Zhang, "Supporting Real-Time Application in an Integrated Services Packet Network: Architecture and Mechanism," *SIGCOMM*, pp. 1-13, 1992.

[Cohen97] R. Cohen and S. Ramanathan, "Using Proxies to Enhance TCP Performance over Hybrid Fiber Coaxial Networks," Hewlett-Packard Laboratories Tech Report #HPL-97-81, 1997.

[Cormen86] Thomas H. Comen, Charles E. Leiserson, and Ronald L. Rivest, *Introduction to Algorithms*, The MIT Press, Massachusetts, 1986.

[Dean94] Thomas Dean, *Artificial Intelligence. Theory and Practice*, The Benjamin/ Cummings Publishing Inc., 1994

[East99] E. W. East, “Infrastructure work order planning using genetic algorithms," *Proceedings of the Genetic and Evolutionary Computation Conference*, San Francisco, CA, 1999, pp. 1510-1516.

[Fankhauser99] G. Fankhauser, M. Dasen, N. Weiler, B. Plattner, B. Stiller. “WaveVideo — An Integrated Approach to Adaptive Wireless Video,” *Mobile Networks And Applications (Special Issue on Adaptive Mobile Networking and Computing)*, 4(4): 255-271, December 1999.

[Feldmeier98] D. C. Feldmeier, A. J. McAuley, J. M. Smith, D. S. Bakin, W. S. Marcus, T. M. Raleigh, “Protocol boosters," *IEEE Journal on Selected Areas in Communications*, vol.16, (no.3), April 1998.

[Ferreria02] Vincent Ferreria, Alexey Rudenko, Kevin Eustice, Richard Guy, V. Ramakrishna, and Peter Reiher, "Panda: Middleware to Provide the Benefits of Active Networks to Legacy Applications," DANCE 02 Active Network Conference, June 2002.

[Fox97] A. Fox, S. Gribble, Y. Chawathe, E. Brewer, P. Gauthier. “Extensible Cluster-Based Scaleable Network Services.” *Proceedings of the 16th ACM Symposium on Operating System Principles (SOSP‘97)*, Saint-Malo, France, October 1997.

[Fox98] Armando Fox, Steven D. Griddle, Eric A. Brewer, and Alan Amir, “Adapting to Network and Client Variations Using Infrastructural Proxies: Lessons and Perspectives," *IEEE Personal Communications*, September 1998, 5(4), pp.10-19.

[Fu01] Xiaodong Fu, Weisong Shi, and Vidjay Karancheti, "Automatic Deployment of Transcoding Components for Ubiquitous, Network-Aware Access to Internet Services," *NYU Computer Science Technical Report CS-TR-2001-814*, March 2001.

[Garey79] Michael Garey and David S. Johnson, *Computers and Intractability. A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, New York, 1979.

[Gero98] John S. Gero and Vladimir A. Kazakov, “Evolving Design Genes in Space Layout Planning Problems," *Artificial Intelligence in Engineering*, vol. 12, 1998, pp. 163-176.

[Gribble99] S. D. Gribble, M. Welsh, E. A. Brewer, and D. Culler, “The MultiSpace: an Evolutionary Platform for Infrastructural Services,” *Proceedings of the 1999 USENIX Annual Technical Conference*, Monterey, California, June 1999.

[Gugmundsson] Olafur Gudmundsson, Brian Wellington, David Reeder, Modlyn Badger, and G. Russ Mundy, "Internet Key Management and Distribution: Architecture and Toolkit Report," *Advanced Security Research Journal*, NAI Labs, vol. 1, no. 1, Fall 1998, pp. 5-27.

[Hauskrecht00] Milos Hauskrecht and Hamish Fraser, “Planning Treatment of Ischemic Heart Disease with partially observable Markov Decision Processes," *Artificial Intelligence in Medicine*, vol. 18, Issue 3, March 2000, pp. 221-244.

[Hicks99] M. Hicks; A. D. Keromytis, “A Secure Plan. Active Networks," *First International Working Conference, IWAN'99 Proceedings*, Berlin, Germany: Springer-Verlag, 1999, pp.307-314.

[Hutchinson91] N. Hutchinson and L. Peterson, “The x-kernel: An Architecture for Implementing Network Protocols,” *IEEE Transactions on Software Engineering*, vol. 17, no. 1, January 1991.

[Ihrig96] Laurie H. Ihrig and Subbarao Kamhaampati, “Design and Implementation of a Replay Framework based on the Partial Order Planner," *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, Cambridge, MA, 1996, pp.849-854.

[Jo98] Jun H. Jo and John S. Gero, “Space Layout Planning Using an Evolutionary Approach," *Artificial Intelligence in Engineering*, vol. 12, 1998, pp. 149-162.

[Joseph96] Anthony D. Joseph, Alan F. deLespinasse, Joshua A. Tauber, David K. Giffort, and M. Frans Kaashoek, “Rover: A Toolkit for Mobile Information Access," *in Proceedings of the Second ACM International Conference on Mobile Computing and Networking (Mobicom ’96)*, November 1996.

[Kaebling98] Leslie Pack Kaebling, Michael L. Littman, and Anthony R. Cassandra, “Planning and Acting in Partially Observable Stochastic Domains," *Artificial Intelligence*, vol. 101, 1998, pp. 99-134.

[Kambhampati94] S. Kambhampati and Craig A. Knoblock, “Planning as Refinement Search: A Unified Framework for Evaluating Design Tradeoffs in Partial-Order Planning," Elsevier Science, 1994.

[Knoblock91] Craig A. Knoblock, “Search Reduction in Hierarchical Problem Solving," *Proceedings of the Ninth National Conference on Artificial Intelligence*, AAAI Press, Menlo Park, CA, 1991.

[Kelly88] F. B. Kelly, “Routing in circuit-switched networks: optimization, shadow prices and decentralization ”*Advances in Applied Probability*, vol. 20, 1988, pp. 112-144.

[Konstantinou02] A. V. Konstantinou, Y. Yemini, and D. Florissi, "Towards Self-Managing Systems," DANCE 02 Active Network Conference, June 2002.

[Korf93] R. Korf, "Linear-Space Best-First Search," *Artificial Intelligence*, 62, 1993, pp. 41-78.

[Legezda98] Ulana Legezda, David Witherall, and John Guttag, "Improving the Performance of Distributed Applications Using Active Networks," *Proceedings of IEEE INFOCOM '98*, vol. 2, New York, NY, 1998, pp.590-599.

[Levy98] J.Y. Levy, L. Demailly, J.K. Ousterhout, B.B. Welch, "The Safe-Tcl security model," *Proceedings of the USENIX 1998 Annual Technical Conference*, Berkeley, CA, 1998, pp. 271-282

[Li02] J. Li, M. Yarvis, and P. Reiher, "Securing Distributed Adaptation," *Computer Networks*, Special Issue on Programmable Networks, vol. 38, no. 3, 2002.

[Liatsos97] Vassilis Liatsos, Barry Richards, "Least Commitment – An optimal planning strategy," IC-Parc, Imperial Colledge London, 1997.

[Liljeberg96] Mika Liljeberg, Heikki Helin, Markku Kojo, and Kimmo Raatikainen, "Enhanced Services for World-Wide Web in Mobile WAN Environment," University of Helsinki, CSD Report C-1996-28.

[Ling97] Ling Zong, D. Y. Y. Yun, "A planning-based graph matching algorithm for knowledge structure retrieval," *Advances in Concurrent Engineering*, Basel, Switzerland: Technomic Publishing, 1997, pp. 223-230.

[Mallet97] A. Mallet, J. Chung, and J. Smith, "Operating System Support for Protocol Boosters," *HIPPARCH Workshop*, June 1997.

[Marcus98] W. Marcus, T. McAuley, T. Raleigh, "Protocol boosters: a kernel-level implementation," *IEEE GLOBECOM 1998*, Piscataway, NJ, 1998, pp. 1619-1623.

[Merigu99] S. Merugu, S. Bhattacharjee, Y. Chae, M. Sanders, K. Calvert and E. Zegura, "Bowman and CANEs: Implementation of an Active Network," presented as an invited paper at the 37th Annual Allerton Conference on Communication, Control and Computing, Monticello, Illinois, September 1999.

[Murphy01] S. Murphy, E. Lewis, R. Puga, R. Watson, and R. Yee, "Strong Secrity for Active Networks," *IEEE Openarch 2001*.

[Nikolsson98] Nils J.Nilsson, *Artificial Intelligence: A new Synthesis*, Morgan Kaufmann Publishers, Inc., San Francisco, CA. 1998.

[Noble97] Brian D. Noble, Dushyanth Narayan, James Eric Tilton, Jason Flinn, and Kevin R. Walker, "Agile Application-Aware Adaptation for Mobility," *Proceedings of the 16th ACM Symposium on Operating Principles*, St. Malo, France, October 1997.

[Petersen01] L. Peterson et. al., "An OS interface for active routers," *IEEE Journal on Selected Areas in Communications*, vol. 19, (no. 3), March 2001.

[Reiher00] Peter Reiher, Richard Guy, Mark Yarvis, and Alexey Rudenko, "Automated Planning for Open Architectures," Short paper presented at Openarch 2000, March 2000.

[Roberts84] Fred S. Roberts, *Applied Combinatorics*, Prentice Hall, New Jersey, 1984.

[Rudenko01] Rudenko Alexey and Peter Reiher, "Experience with automated planning for Panda," Tech Report CSD-TR 010041, Computer Science Department, UCLA, 2001.

[Russel95] Stuart J. Russel and Peter Norvig, *Artificial Intelligence*, Prentice Hall, Englewood Cliffs, New Jersey, 1995.

[Schwartz99] B. Schwartz; A. W. Jackson; W. T. Strayer; Wenyi Zhou; R. D. Rockwell; C. Partridge, "Smart Packets for Active Networks," *1999 IEEE Second Conference on Open Architectures and Network Programming Proceedings, OPENARCH '99*, Piscataway, NJ, 1999, pp. 90-97.

[Silva98] S. da Silva, D. Florissi and Y. Yemini, "Composing Active Services in NetScript," position paper presented at DARPA Active Networks Workshop, Tucson, AZ, March 9-10, 1998.

[Spatschek98] O. Spatscheck; L.L. Peterson, "Defending against denial of service attacks in Scout," *Operating Systems Review*, ACM, Winter 1998, pp. 59-72.

[Sudame98] Pradeep Sudame and B. R. Badrinath, "Transformer Tunnels: A Framework for Providing Route-Specific Adaptations," *Proceedings of the 1998 USENIX Annual Technical Conference*, New Orleans, LA, June 1998.

[Tennenhouse97] David L. Tennenhouse, Jonathan M. Smith, W. David Sincoskie, David J. Wetherall, and Gary J. Minden, "A Survey of Active Network Research," *IEEE Communications Magazine*, January 1997, 35(1), pp. 80-86.

[Ventkatasubramian02] Nalini Ventkatasubramian, "Safe 'Composability' of Middleware Services", Communications of the ACM, vol. 45, (no. 6), June 2002, pp. 49-58.

[Weld94] Deniel S. Weld, “An Introduction to Least Commitment Planning," AI Magazine, vol. 15, (no. 4), Winter 1994, pp. 27-61.

[Whetherall98] D. J. Wetherall; J. V. Guttag; D. L. Tennenhouse, “ANTS: a toolkit for building and dynamically deploying network protocols,*" 1998 IEEE Open Architectures and Network Programming*, San Francisco, CA, 3-4 April 1998, pp.117-129.

[Wetherall99] D. Wetherall, “Active network vision and reality: lessons from a capsule-based system," *Operating Systems Review*, vol. 33, (no. 5), ACM, December 1999, pp. 64-79.

[Yarvis99A] Mark Yarvis, An-I A. Wang, Alexey Rudenko, Peter Reiher, and Gerald J. Popek., “Conductor: Distributed Adaptation for Complex Networks," UCLA Tech Report CSD-TR-990042, August 1999.

[Yarvis99B] Mark Yarvis, Peter Reiher, and Gerald J. Popek, “Conductor: A Framework for Distributed Adaptation," *Proc. Seventh Workshop on Hot Topics in Operating Systems (HotOS VII)*, Rio Rico, AZ, March 1999.

[Yarvis00] Mark Yarvis, Peter Reiher, and Gerald J. Popek, “A Reliability Model for Distributed Adaptation," *Openarch 2000*, March 2000.

[Zegura98] AN Composable Services Working Group, “Composable Services for Active Networks," Ellen Zegura, editor, http://www.ittc.ukans.edu/anserve/

[Zennel98] Bruce Zennel and Dan Duchamp, “A General Purpose Proxy Filtering Mechanism Applied to the Mobile Environment," *Proceedings of the Third Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom’ 97)*, October 1997, pp. 248-259.

[Zhou97] G. Zhou, M. Gen, "Evolutionary computation on the multicriteria production process planning problem," *Proceedings of 1997 IEEE International Conference on Evolutionary Computation* (ICEC '97) New York, NY, 1997, pp. 419-424.