Agent Building and Learning Environment
What are intelligent agents?
Agents model
Architecture of an intelligent agent system

ABLE - Overview
What are Javabeans?
The AbleBean the foundation for the framework

A simple Artificial Neural Network with ABLE
1. Able Agent Editor
2. Use of the AbleBean as an ordinary Java object
   3.1 Data files to train the neural network
   3.2 Use of the AbleImport data bean to get data to the network
4.1 Use of a serialized backpropagation bean
4.2 Advantages with use of serialized beans
What are intelligent agents?

- Agents are active, persistent software components that perceive, reason, act, and communicate.
- Software that **assists** people and acts on their behalf.
- Agents can help **people** and **processes**.
- Agents are used for automation and control.

+ Finding and filtering information.
+ Personalizing your environment.
+ Negotiating for services.
+ Automation of tedious tasks.
+ Taking actions you delegate.
+ Learning about you over time.
+ Collaborating with other agents.
+ Capturing individual and organizational knowledge.
+ Sharing knowledge.

+ Finding and fixing problems.
+ Automation of complex procedures.
+ Finding “best fit” procedures.
+ Pattern recognition and classification.
+ Predictions and recommendations.
+ Negotiate and cooperate with other organizations agents.
Agents model

Task level skills
- Task
  - Information retrieval
  - Information filtering

Knowledge
- A priori knowledge
  - Developer specified
  - User specified
  - System specified
- Machine learning
  - Neural networks
  - Genetic algorithms
  - etc.

Communication skills
- With user
  - Interface
  - Speech
  - Social
- With other agents
  - Language
Architecture of an intelligent agent system

Software system:
- Attention
- Perception
- Domain knowledge
- Learning
- Memory
- Reasoning
- Communication

Environment:
- Sensors
- Actuators
- System
ABLE – Overview

**Data mining**
- Classification
- Clustering
- Prediction
- Optimization

**Machine learning**
- Neural Networks
- Decision Tree
- Naïve Bayes
- k-NN
- Genetic algorithms

**Machine reasoning (rule based)**
- Scripting
- Simple if-then rules
- Rete’ net
- Prolog
- Fuzzy systems
- Planning
- Event correlation
- Templates
- Author/Test/Debug tools

**Agents**
- Autonomous
- Asynchronous event processing
- Timer-based processing
- Distributed multi-agent platform
- Conversational support
- Author/Test/Debug tools
What are JavaBeans?

- From Sun’s JavaBeans homepage:
  
  “JavaBeans is the platform-neutral, component architecture for Java.”

- JavaBeans are components, that when assembled make up an application.  
  A JavaBean is a Java class that defines properties and that communicates with other Beans via events.

- No knowledge of programming languages is required to create JavaBeans applications.  
  The pre-existing components are “wired” together, somewhat like a circuit diagram.

- Creating programs from JavaBeans takes minutes instead of hours or days.

- The Beans themselves must be written by programmers.  
  Each bean need to be written only once and then shared with others.  
  There are currently hundreds of beans available on the Internet.

- JavaBeans are active during application development.  
  You can see the components of your program working before you complete your application.

- Because JavaBeans are written in Java, applications built with JavaBeans can run on any computer.  
  Windows, Unix, Macintosh, or Java Console.

- JavaBeans are now a fundamental part of the Java core libraries.  
  Many programs are currently being written to create and manipulate JavaBeans.
The AbleBean, the foundation for the framework

- **AbleBean, AbleRemoteBean**
  A Java interface (local and remote).
  The ABLE framework and component library is built on the JavaBean component model.

- **AbleObject**
  AbleBean instantiation with autonomous thread
  Bean interactions through direct method calls and event passing.
  AbleBeans can be used as standard Java objects with method calls, or they can be used as autonomous agents, running on their own thread.

- **AbleEvents**
  Notification and Action events with synchronous and asynchronous event handling.

- **AbleBeanInfo and Customizer** required for use in Agent Editor (GUIs)
A simple Artificial Neural Network with ABLE

• Example neural network from Mitchells book page 107

1. Simple way – work with the **Agent Editor**
2. Use the the AbleBean (AbleBackPropagation) as an **ordinary Java object / JavaBean**.
3. Data to train the AbleBackPropagation bean from a **data file import bean** (AbleImport)
4. Use of a trained and **serialized** AbleBackPropagation bean in Java program.
5. Use of AbleEvents
1. ABLE Agent Editor (JavaBean editor)

- The ABLE Editor is a full featured IDE (written in Java) that can be used to design, test, and debug intelligent agents.

- Every component (bean) has an Inspector window

- The property window for the BackPropagation bean.
2. Use of the AbleBean as an ordinary Java object

```java
public static void bp1()
{
    AbleBackPropagation bp = null;
    double[] in1 = {1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0};

    double[] indata;
    double[] outdata;

    try
    {
        bp = new AbleBackPropagation("bp", "8 3 0 0 8 0");
        bp.init();
        bp.setNetMode(AbleBackPropagation.NNTRAIN);
        indata = (double[]) bp.getInputBuffer();

        for( ... )
        {
            indata = ...;
            bp.process();
            outdata = (double[]) bp.getOutputBuffer();
        }

        bp.setNetMode(AbleBackPropagation.NNRUN);

        ... 
    }
    catch ( AbleException ae) { System.out.println(ae); }
}
```

**NN-architecture**
- 8 nodes in input layer
- 3 nodes in hidden layer
- 8 nodes in output layer

Train the network
- Generate data to the input vector
- Process the backpropagation algorithm

Use the network
3.1 Data files to train the neural network

- Definition file digital.dfn
  
  input1 continuous input
  input2 continuous input
  input3 continuous input
  input4 continuous input
  input5 continuous input
  input6 continuous input
  input7 continuous input
  input8 continuous input
  output1 continuous output
  output2 continuous output
  output3 continuous output
  output4 continuous output
  output5 continuous output
  output6 continuous output
  output7 continuous output
  output8 continuous output

- Data file digital.dat
  
  1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
  0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0
  0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0
  0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0
  0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0
  0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0
  0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0
  0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1
public static void bp2()
{
    AbleBackPropagation bp = null;
    AbleImport im = null;
    AbleBufferConnection bc = null;

    try{
        im = new AbleImport();
        im.setDataFileName("kmo/able/ann/digital");
        im.init();
        im.open();

        bp = new AbleBackPropagation("Backpropagation 1", "8 3 0 0 8 0");
        bc=new AbleBufferConnection(im,bp);
        bp.setDataFlowEnabled(true);
        bp.init();

        for (...){
            im.process(); //Read data.
            bc.process(); //Move data.
            bp.process(); //Process data.
        }

        bp.setNetMode(AbleBackPropagation.NNRUN);

        ...
    }
    catch (AbleException ae) { System.out.println(ae); }
}
4.1 Use of a serialized backpropagation bean

```java
public static void bp3()
{
    AbleBackPropagation bp = null;
    double[] indata;
    double[] outdata;

    try
    {
        System.out.println("=== Method bp3 ===");

        bp = (AbleBackPropagation)
            AbleObject.restoreFromSerializedFile("kmo/able/ann/bp.ser");

        indata = (double []) bp.getInputBuffer();

        for (int i=1; i<=10; i++)
        {
            indata = ...;

            bp.process();
        }
    }
    catch ( Exception e ) { System.out.println(e); } }
```
4.2 Advantages with use of serialized beans

• Serialization of the JavaBean is the base for the mobile agent implementation. An agent can be stopped and saved with its internal state and send to another computer.

• Example: Soft sensor based on neural networks

  Training of the sensor network can be made off line and the serialized bean can then be distributed to different users.

• Able

  Once you have a trained neural network, you can save the neural network state in serialized form.

  This is true for all ABLE agents/beans, not just for neural networks.

  ABLE allows you to simply store and reuse in your application whatever has been learned.

  Since ABLE allows you to store a trained neural network in Java serialized form, these objects can be re-created anywhere that a JVM (Java Virtual machine) will run.