ABSTRACT

A computer-based simulation has been a well-touted information systems for managing the risk of disease spread. This paper shows how to create/improve a computer simulation for better decision making through a project that integrates a multi-agent framework into the traditional herd-based epidemiological simulations. The simulation incorporates interferences of individual objects like transportation, human workers, and herds of animals. The embedded stochastic module estimates the epidemic spreading behavior, which is required for better and faster decision-making and responses in epidemiological emergencies. This paper contributes to students’ understanding of the whole process of technology innovation by demonstrating a practical project.

KEYWORDS: Simulation, Multi-agents, Decision-support, XML, Ladybug, MASON

INTRODUCTION

This paper aims at creating an example as a practical guidance to those students who want to utilize or program computer simulation in any applied setting like business, engineering, pharmacy, and other health industries. Students in programing should understand the whole process of simulation programing from identifying the need or opportunity for simulation through conceptual modeling to physical coding. This paper illustrates the whole process of developing a computer simulation so that students can use this as a handy reference.

COMPUTER SIMULATION IN EPIDEMIOLOGICAL ESTIMATION

The field of computer Information Systems has been creating a great synergy by incorporating the potent computing power into complicated details of studies across various fields like health information systems in the health industry. Likewise, a computer-based simulation has been well-touted not only for business and software development (Hua & Bapna, 2013; Ghosh, 2009), but also for management and assessment of the risk of disease outbreak and spread, given that field trials are too expensive or illegal (Lim & Kim, 2013). Disease outbreaks like Severe Acute Respiratory Syndrome (SARS), West Nile Virus, or Foot-and-Mouth Disease (FMD) may strike livestock rapidly and without warning. The recent outbreak of disease among livestock in the United Kingdom in 2007 and Japan and South Korea in 2010 clearly demonstrated the importance of effective early response. Apltae epizooticae, known as one of FMDs, is highly
contagious and affects all cloven-hoofed animals such as cattle, pigs, deer, goats, and buffalo. Since this disease spread rapidly after the outbreak, governments were forced to collect, kill, and bury millions of cattle, pigs, goats, and deer. For example, South Korean lost approximately 1.78 billion USD in 2010-2011 (Yonhap News, 2011). Therefore, it is important that states collaborate, communicate, and pre-plan to respond to these emergency outbreaks at the very beginning of the planning process. Disease control strategies should be established based on proper estimation of the grade and range of disease.

In traditional epidemiological estimation models, the disease manifestation and transmission are represented at the level of herds, not at the level of individual animals (Audigé et al., 2000; Team, 2008). In other words, in a herd-based simulation, the status of a herd is changed from one state to another state (i.e., from a susceptible state to an infected state). In addition, stochastic modeling is used to indicate the various random processes that are responsible for the spread of disease (Team, 2008). Therefore, a simulated outbreak is represented by the final result of a unique series of random events and processes. However, consideration of livestock transportation in the simulation will make the estimation more difficult. The pattern of the spread of disease could be complicated and very different from traditional estimation framework. In addition, the traditional, herd-based simulations use stochastic sampling methods, which have limited capabilities for representing the behavior of individual animals in the simulation (Asmussen, 1999). For example, FMD can be transmitted by close contact between animals, by the clothes and skin of animal handlers, by the fodder the animals eat, or by motor vehicles (Figure 1). In addition, most traditional stochastic simulations are designed to run on a single CPU environment. The limited computing power may cause uncontrolled latency problems when the given domain size is large. Given these complications, it is clear that a more sophisticated, practical, and inclusive model is needed.

These problems suggest that traditional methods will not be adequate for circumstances that require fast responses from large area simulations, which calls for developing more practical and sophisticated simulation model. Therefore, this study pays attention to the synergy between stochastic sampling methods and potentials of agent-based model. The proposed model accounts for many restrictions, randomness, unexpected interactions across a variety of agents, and autonomous activities within and across environmental variables.

Figure 1. Disease-spreading among herds in traditional herd-based models
WHY MULTI-AGENT SYSTEM?

Agent based modeling is a top-down, micro modeling approach. The whole system is represented by a large set of small components or agents, local behaviors of each component, and interactions between components (communication between the agents). This modeling approach focuses on discovering the global system of behaviors that emerge from the actions of agents and their interactions (Wooldridge, 2002). Further, it incorporates autonomous agents, and simulates their actions and interactions to help evaluate the whole phenomenon (Niazi & Hussain, 2011). Each individual agent is typically bound by certain restrictions like geological or behavioral boundaries. Therefore, an agent uses its peculiar heuristics or simple decision-making rules (Axtell, 2003). Further, these agents may experience learning, adaptation, and reproduction (Bonabeau, 2002). The complexity of today's workplace and the dramatic increase in the amount of available information broadens the role of intelligent agents, which results in alleviating human burdens while agents are working with or without human intervention. Agents may extract valuable information from large amounts of raw data, give users valuable advices for appropriate decision, and control other instruments without human intervention.

The study of multi-agent systems (MAS) focuses on systems where many intelligent agents interact with each other. For example, when a multi-agent system represents a traffic simulation, the whole system can represent a set of agents (i.e., various objects in the simulation such as cars, roads, and traffic lights) and the interactions between the agents (Khalesian & Delavar, 2008; Hackney & Axhausen, 2006; Wooldridge, 2002). Multi-agent systems are often used to simulate the interdependencies between social networks and travel behaviors. In such a simulation, the model simultaneously generates social network(s) and travel behavior(s) by defining social-networking visits as travel activities. Information about space and other agents flows only via the social network (Hackney & Axhausen, 2006). In addition, a multi-agent based simulation is also very flexible and easy to extend. It can be easily distributed over multiple computers, even over the Internet-based network or cloud computing, which enables any number of agents to join a simulation with fast response time.

Figure 2. Disease-spreading in the multi-agent framework
In our proposed system, an agent represents any object and its behavior. The object can be an animal, a group of animals (herd), a worker, or a transportation unit. The behavior of an agent can be decided by its role. For example, the main role of a herd agent is estimating the spread of the disease based on its stochastic model in case of infection. Consequently, the behavior of the herd will be changed based on the estimation. The goal of the proposed study is to use a multi-agent framework to design and implement a computer-based epidemiological simulation model that combines the traditional herd-based epidemiological methods with the role of transportation and the interferences of individual objects for herds. Since this proposed simulation model is capable of representing the behaviors of individual objects beyond keeping the benefits of stochastic herd-based simulation, it will provide more accurate and flexible simulation results that can be used to facilitate the early responses to emergency outbreaks. Figure 2 shows an example of multi-agent framework in our proposed system.

MODEL DEVELOPMENT

The whole system of agent-based modeling is represented by a large set of small agents, local behaviors of each agent, and interactions or communication between agents. Therefore in this research, we design and implement a multi-agent system by incorporating modern multi-agent framework into the traditional stochastic model. The implementation includes (i) the design and verification of a stochastic simulation module for herd-based disease-spread evaluation, (ii) the design and verification of multi-agent based simulation-modules to represent the interferences among herds, human workers, and the individual transported animals, and (iii) the development of graphical user interface modules to control the overall simulation process. The resulting simulator enables users to put any type of estimation models (stochastic or agent based) into simulation. Also, a very large size of simulation is even possible since our simulation system allows any number of agents to be deployed into simulation. The implemented features include:

- **Graphical user interface (GUI):** The simulator is designed to provide estimations of disease outbreaks to the users in graphical user interface format. The GUI module covers the file and log managements, the configuration of simulation in graphical forms, and the control of simulation.
- **Graphical status report:** The geographical map displays the location and epidemic status of the farms and herds during simulation. The detailed status window also shows the number of animals in each status (susceptible, exposed, infected, and recovered). The result of each simulation also is reported to the user with graphical charts.
- **Capability of simulating an individual object:** In addition to simulating disease transmissions in herds, the proposed simulator is able to estimate the role of livestock transportation, roles of workers at the farm, and the individual effect of the transported animals in disease spreading. The roles and weights of interaction among each individual objects are easily configured in a simulation through a scenario in XML (extended markup language) format. Thanks to these features, users can control their simulations to be more flexible and accurate than traditional herd-based simulators.
- **Compatibility:** The proposed simulator is written in Java, which is one of the most compatible computer languages. Thanks to its compatibility, the proposed simulator is adequate for any computer environment. For example, users can select any computer from a tablet to a super computer based on the desired simulation size.
- **Extendibility:** The multi-agent framework and the modules written in Java computer language provide extensibility and reliability in simulation. Users can determine not only the size of herds, workers and transportsations, but also their roles with a XML
configuration file. As a result, a user can execute a very large size of simulations without any modification, and the capacity of hardware that runs the simulation should be considered. Since the API (application programming interface) modules are written in Java that allows easy maintenance and modification, each module can be updated independently without interfering with other modules.

**METHODOLOGIES**

To accomplish the goal, we use stochastic modeling, multi-agent framework, graphic user interface modules, geographic map modules, JFreeChart for chart report, Extended Markup Language, and various Java tools.

**Stochastic Modeling**

A herd in our simulation is a group of multiple units that consist of multiple individuals. Epidemic estimation in the simulation is represented by a stochastic model. The stochastic estimation is implemented with Ladybug. *Ladybug* is a simulation and parameter estimation of Susceptible-Exposed-Infected-Recovered (SEIR) epidemics, which was originally written by Höhle and his colleagues (Höhle et al., 2005; Höhle & Jørgensen, 2002). Since its codes were initially written for a single herd model with command line interface (CLI), this CLI and lack of functions for communication with other modules were major problems for a large simulation. To overcome these issues, we modify Ladybug for our multi-agent framework. The modification and extension include analysis, optimization, and rewriting of essential codes in Ladybug. In addition, a number of new functions are added to our modules, which is mainly required for communication and multi-tasking during a simulation (Figure 3).

**Figure 3. Architecture of the simulator**

With the modified Ladybug module, we design and implement ‘Herd’ agents in Java, which simulate the operation of a stochastic module for herds in the multi-agent framework. The implementation requires Ladybug to be embedded into a Herd agent, and so we wrap Ladybug
into a Herd agent through the Ladybug-bridge module that we developed. As a result, the proposed simulator allows virtually unlimited number of Herd agents, which simulate the behavior of a stochastic model with communication capability. The herd agent in this simulation system can host any type epidemic models from the traditional stochastic models to the modern agent-based models, so that it helps the simulation achieve flexibility and scalability.

**Multi-Agent Framework**

We also implement a multi-agent system to host various types of agents and simulate their roles. The implementation covers: (1) the effect of disease transmission by farm workers between livestock herds and (2) the role of transported animals in disease transmission between herds. For this task, we use a Multi-agent tool kit and Message exchange package in Java along with MASON (Multi-agent Simulator of Networks). MASON is a fast discrete-event multi-agent simulation written by the researchers in George Mason University (Luke et al., 2005; Balan et al., 2003). It is designed to be a foundation for large custom-purpose Java simulations and to provide sufficient functionality for many lightweight simulating needs. The main roles of MASON in our project are (i) to create and deploy agents, (ii) to provide communication between multiple agents, and (iii) to control the simulation and collect status reports. As shown in Figure 3, each of the stochastic models is wrapped with the agent framework. The bridge module embedded in each of the stochastic module provides conversions and communication with the herd agent module. As a result, the wrapped module becomes an agent, which is specialized in stochastic estimations. With the message exchange feature, each agent is able to exchange status reports with others.

**Graphical User Interface Modules**

Graphical User Interface (GUI) is a program interface for users to provide conveniences based on the computer’s graphic capabilities. In this proposed simulation systems, GUI modules are written with Swing and AWT in Java (Geary, 1999; Elliott, 2002). The implemented GUI modules for users’ run of simulation include menus, file selection, and various panels and frames. Further, we implement a Map frame to display the geographic map for simulation, where users are offered the Agent Communication Display frame which demonstrates the interactions between agents, and the Status Report frames based on the JfreeChart package. The log display panel is also written in Java, and so users can track down the progress of simulation through this panel.

**Geographic Map**

Understanding of the environment is important for an epidemic intervention when there is a breakout. The geographic information of farms and herds help users to estimate the sequence of epidemic spreading, which will allow users to make a prompt decision. For this purpose, we design and implement a map frame, which provides the geographical information and the status of farms during simulation. Our map agent displays the geographic information and the status of agents (farm and herd) on a map based on World Wind version 1.5.1. We also design annotations and annotation layers to display the status of agents as well as warning messages during simulation.

World Wind is an open-source Java Package tool developed by NASA. It represents a virtual globe, which has become one of the most popular open-source graphical map package with large user communities (NASA, 2013; NASA, 2011). Unlike other commercial geography
browsers such as Google Earth (Google), World Wind focuses on science and education activities, and it has been built based on multiple views of the earth. Because of its design, World Wind is known to have better performance than the free version of Google Earth.

Chart Reports

*JFreeChart* is an open-source, free domain chart generator, which provides various types of charts in Java API (*JFreeChart*, 2013). In this project, we use *JFreeChart* version 1.0.17 to generate four types of charts as simulation results: (i) pie charts for farms and herds, which show the number of animals in each epidemic status based on the SEIR categorization, (ii) pie charts for units in a herd when the detail reports are requested, (iii) scattered XY charts to show the infection ratio in herds for each simulation step, and (iv) bar charts for farms to display the number of animals in each epidemic status at each simulation step. Implemented pie charts, XY-scattered charts and bar charts are listed in Figure 8, Figure 9, and Figure 11 respectively.

Extended Markup Language

Extended Markup Language (XML) is a technology for creating markup languages to describe data structures of virtually any type in a structured manner (Raisinghani, 2005; Deitel, 2000). Thanks to its capability to define a set of rules for encoding documents and describe data structures in a well-arranged format, XML has become popular for programmers to represent various types of structured information (Pons, 2004; Reyes et al., 2002). Since one of advantages of XML is to represent data without any additional information (Williams et al., 2000), we utilize XML for scenario configuration files. We use Java Architecture for XML binding (JAXB) to translate XML files to Java classes (Benz & Durant, 2009) so that a scenario in XML can be easily converted to a *Scenario* class, which will initiate a simulation. In implementation, *ScenarioFactory* class reads a scenario, and then produces an instance of *Scenario* class. This *Scenario* instance is transferred to *Deployment* module to initiate a simulation.

Figure 4. A Sample Scenario in XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xmlScenario xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns="http://www.example.com/scenario" xsi:schemaLocation="http://www.example.com/scenario scenario.xsd">
  <description>
    Simple Scenario in JAXB based reader format.
    This scenario shows epidemic spreading for two imaginary farms in South Carolina.
    This scenario includes two farms 10 in ten-miles distance, multiple herds, a
    breakout starts from a single infected animal in the first farm, and is
    created: 11-14-2013
    Last modified: 3-31-2014
  </description>
  <farmList>
    <farm>
      <description>Farm 1: A farm near Orangeburg South Carolina</description>
      <longitude>-80.88557</longitude>
      <latitude>33.484274</latitude>
      <herdList>
        <herd>
          <id>F11</id>
          <displacement>0.1</displacement>
          <diseased>0.1</diseased>
          <noInfected>0</noInfected>
          <dataFileName>data\dataFile1\data\dataFile1</dataFileName>
        </herd>
        <herd>
          <id>F12</id>
          <displacement>0.2</displacement>
          <diseased>0.2</diseased>
          <noInfected>0</noInfected>
          <dataFileName>data\dataFile2\data\dataFile2</dataFileName>
        </herd>
      </herdList>
    </farm>
  </farmList>
</xmlScenario>
```
shows an example of scenarios in XML. Users can determine the number of agents, their names, geographic locations, other configurations for a simulation through a XML file. Since XML is written in text, users can quickly modify simulation scenarios with a text editor.

**Programming Tools**

We also use Netbeans version 8, which is an official a Java IDE (Integrated Development Environment) for Java (Oracle). The Netbeans IDE is used to write and debug various Java modules for our proposed simulation systems. Those Java codes are executed by a Window 7 computer with 2 Xeon CPUs and 12 GB of memory.

**IMPLEMENTATION**

**Agents and Their Roles**

Various agents are implemented, and agents’ roles and their interactions are summarized in Figure 5 and Table 1. Herd Agents’ main roles are to run the embedded stochastic epidemic model in Ladybug, and to collect the status reports from the model. The collected herd reports are submitted to a Farm agent for compilation, which results in consequent reports to upper level agents. Herd agents also communicate with Worker agents to simulate the epidemic interactions between herds and farm workers. Herds and workers exchange their epidemic status so that they can update their status with a random-based infection function. We also implemented farm agents in multi-agent framework.

Since a farm includes a group of herds and farm-workers in our simulation model, our farm agents should be able to monitor the status of herd agents and worker agents in their respective farms. Therefore, we design the farm agent such that it creates herd agents and worker agents when a farm agent is initiated. Then, a farm agent monitors the status of its herd agents by collected reports from them. The collected reports will be stored in a history file with a time mark. It is also a part of farm agents’ mission to compile and summarize and submit the collected reports the map agent during simulation. Given that the proposed simulation represents and estimates the disease spreading between herds, a worker agent is made to
interact with all herds in a farm and to randomly (using a random epidemic spreading function along with the interaction frequencies) transfer infections between herds.

Transportation Agents are designed to simulate the epidemic spreading between farms. In our simulation model, transportation agents exchange animals between farms with random-based selection algorithm, which affects the epidemic status of corresponding farms in the next simulation step. Users can decide the frequency of exchange and the number of animals in each exchange with a simulation configuration. A map agent collects status reports from farms and herd agents and displays the reports on the map through the WorldWind map package. The displayed information includes: (1) the geographical information of farms and herds, (2) The infection level of farms and herds in color (green, yellow, and red), (3) The message exchanges between agents, and (4) Warning messages from breakout events. As a part of controller, a monitor agent controls simulation steps when the batch mode is selected. Its missions include the start/stop of a simulation, creation of a new simulation instance, and collection/management of the results. The stored records are utilized to generate a batch report.

Table 1. Role of Agents

<table>
<thead>
<tr>
<th>Agent Name</th>
<th>Roles</th>
<th>Communication with</th>
</tr>
</thead>
<tbody>
<tr>
<td>Herd Agent</td>
<td>• Child of Farm Agent &lt;br&gt; • Control embedded simulator in herd based &lt;br&gt; • Send status reports to Farm Agent</td>
<td>Farm Agent Worker Agent</td>
</tr>
<tr>
<td>Farm Agent</td>
<td>• Collect and compile status reports from herds &lt;br&gt; • Submit summary of reports to the map agent</td>
<td>Map Agent, Transportation Agent</td>
</tr>
<tr>
<td>Worker Agent</td>
<td>• Child of Farm Agent &lt;br&gt; • Represent the role of worker in a farm. A worker agent interacts with herd agents to simulate the spreading of a disease between herds &lt;br&gt; • The selection and infection are represented by with random based functions.</td>
<td>Herd Agent</td>
</tr>
<tr>
<td>Transportation Agent</td>
<td>• Represent the role of transportation between farms &lt;br&gt; • This agent interacts with farm agents to simulate the epidemic spreading of a disease between farms</td>
<td>Farm Agent</td>
</tr>
<tr>
<td>Map Agent</td>
<td>• Collect status reports from farm agents and displays reports on the Map frame graphically &lt;br&gt; • Collect Warning messages on the Map frame</td>
<td>Farm Agent</td>
</tr>
<tr>
<td>Monitor Agent</td>
<td>• Control and monitor the simulation steps in batch simulation mode. &lt;br&gt; • Record farm status reports for each batch step.</td>
<td>Simulation Console</td>
</tr>
</tbody>
</table>

Other Major Modules

To combine packages and tools mentioned in methods, many other Java modules are also implemented. The following list illustrates some of major implementations:

- Design and implementation of agents and their interface for MASON.
- The control module, which controls the simulation tasks of MASON via GUI.
- GUI modules for menus, panels and frames in Java.
• Modification and optimization of Ladybug stochastic model.
• Ladybug Java bridge for Herd Agent.
• Java Frame for World Wind package and Java codes for annotation and control for World Wind.
• Chart reports and Status panel developed in JFreeChart and Java Graphics.
• XML file reader/writer using Java JAXB.

APPLICATION OF THE SIMULATION TO FARMS AROUND A RURAL AREA

Multiple scenarios were created and simulated for this proposed simulation systems. However, for demonstration purposes, we selected a scenario with imaginary farms near the rural Orangeburg, South Carolina. The selected scenario consists of five farms. Two to four herds and multiple farm-workers were assigned to each farm, and multiple transportations connected the farms. Each herd consisted of 8 x 16 units with 2,584 animals. This surmounts to a total 25,088 animals. The entire description of this scenario is listed in Appendix. Figure 6 illustrates the framework for agents’ communication in the selected scenario.

Figure 6. Agent Communication Display Frame for the Selected Scenario

Figure 7. Map Frame for the Selected Scenario
Figure 7 shows a screenshot of the map frame of the five farms during the simulation phase. The farm status panel (right column of the frame) displays the current status of each farm in the simulation. The yellow color warns users that Farm-2 and Farm-4 have some infected animals in a herd. The ellipse for the Farm-5 (at the bottom left of the map) shows the status of four herds. One herd is in normal status, but the other three are in warning status (yellow and red).

When Farm-1’s status panel is clicked, a list of herd status panels will be displayed. Figure 8 shows a screenshot of the herd list panel. As mentioned in the previous section, each pie chart explains the epidemic status of a unit (Susceptible, Exposed, Infected, and Recovered) in a herd with colors of red, blue, green and yellow respectively. For example, the unit (4, 7) which is located at 4th column and 7th row of the grid shows a significant portion of animals in the exposed status.

Figure 8. Detail Status of Units in Herd-1 of Farm-1

Figure 9 shows a graph for the number of infected animals with time steps as the independent variable. This was generated by a single iteration simulation (interactive mode). The graph shows the breakout steps as well as the most active epidemic spread steps in each herd. Figure 10 shows the graph resulting from running batch mode. We iterated through the selected scenario 10 times, and the simulation generated the chart from the average value. Figure 11 and 12 show the same outputs through a different angle with bar charts.
Figure 9. XY Scattered Chart for Number of Infected Animals and Steps from Interactive Mode

Figure 10. XY Scattered Chart for Number of Infected Animals and Steps from Batch Mode
Figure 11. Bar Chart for Farms based on Status from Interactive Mode

![Bar Chart for Farms based on Status from Interactive Mode](image1)

Figure 12. Bar Chart for Farms based on Status from Batch Mode

![Bar Chart for Farms based on Status from Batch Mode](image2)
CONCLUSION AND FUTURE WORKS

The results from the farm simulation demonstrated and supported the proposed features and benefits of our simulator systems. The multi-agent framework was able to create and deploy any number of agents from the scenario rather than focus only on the herd base, and each agent operated correctly based on their roles. The stochastic module embedded in the herd agent also effectively estimated the epidemic spreading behavior for a herd. The interaction among agents was operating correctly to represent the epidemic spreading behaviors between herds and farms. The simulator also notified warnings and generated chart reports in a timely manner so that users can clearly and promptly observe and understand the overall epidemic spreading patterns. This study shows that the lack of consideration of autonomous or random interaction among various agents in traditional herd-based simulations can be well-addressed through integration of multi-agent framework into the herd-based stochastic simulation model. The framework demonstrated in our proposed simulation model would contribute to the field with capabilities for better and faster decision-making and responses in epidemiological emergencies.

While our epidemic simulator model is properly equipped with most of the fundamental functions and features to provide accurate predictions, there remain some opportunities for potential extensions. First, to build a hybrid simulator that combines traditional stochastic models with a multi-agent framework, our research focused on the design of agents and their agent-to-agent communication methodologies. We simplified the actual epidemic transition mechanism only for workers and transportation methods. However, the flexibility of the multi-agent framework allowed us to add any type of agents in future including air and water agents. The introduction of more agents of different types in nature, which are equipped with accurate interaction functions, will make this simulator more practical. Second, thanks to the extensibility of multi-agent systems, our model has promising capabilities of extension and scalability. Equipped with proper hardware, the epidemic simulator could cover bigger areas without any modifications, while covering multiple regions will require a tremendous amount of computing power that cannot be covered by a single computer system. A distributed system, which allows any number of networked computers to join the simulation, may be a solution. Therefore, building a distributed multi-agent based simulation will be a promising research field of the near future. Third, as mentioned in the methodology section, we used a random-based function to represent the epidemic transitions among agents. However, more sophisticated model based on the characteristics of the agents such as probabilistic models may improve the accuracy of the overall simulation. Finally, research on decision-making systems for time-sensitive matters will be another extension for this research. Combined with a decision-making support model, the simulator will be able to provide more realistic predictions, and suggest the best actions to be taken at a given time when a disease outbreaks.

Overall, the present study shows students a great example of developing and improving decision-making process through technology innovation. This paper contributes to students' understanding of the whole process of technology innovation by demonstrating a practical project from the beginning to the end of the project.

ACKNOWLEDGEMENTS

This work was supported by the USDA National Institute of Food and Agriculture, Evans-Allen Research Program.
REFERENCES


Raisinghani, M.S. (2005), An Interview with Mike Willis CPA and Founding Chairman of Extensible Business Reporting Language (XBRL) International Steering Committee and a Partner at Pricewaterhouse Coopers, Journal of Global Information Technology Management, 8 (1), 64-69.


