DECISION SCIENCES INSTITUTE
An Automated Decision Support System for Sensor-Based Irrigation Scheduling

Sandip Patel, Ph.D.
Morgan State University
Email: sandip.patel@morgan.edu

ABSTRACT

A decision support system (DSS) that takes input from field sensors and decides when to turn irrigation on or off is a missing link in the layout for sensor-based irrigation scheduling. The commercially available DSS have several faults, including that they do not facilitate the complete automation. We developed a DSS that helps farmers with automated irrigation scheduling and remedies the identified faults. In this paper, we first present a layout for complete sensor-based automation. Then, we discuss the specifications of DSS that overcome the limitations of existing DSS. Finally, after presenting the implementation details, we suggest future research directions.

KEYWORDS: DSS, Automated Problem Solving, Sensor-Based System, Intelligent Decision Making, and Expert System

INTRODUCTION

A reason for low agricultural productivity (Naiqian et al, 2002) in many developing countries is poor water management (Minhas and Tyagi, 1998; Soto-García et al., 2013). Farmers in developing countries have limited resources and expertise to effectively and reliably determine the current conditions in the ground (soil salinity, moisture, and temperature), which leads to improper irrigation techniques (Karimov et al., 2010; Yadav, et al., 2009). These farmers are constantly faced with irrigation scheduling (Thompson et al., 2002) problem of when to irrigate and how long to irrigate and are left to guess (Ray et al., 2002; Laycock 2011). Manual data collection not only takes time, it also causes delays in interpreting the data properly to make accurate irrigation decisions (Prasad et al., 2006). Many farmers rely on interpreting inaccurate visual inspections of plants (Gontia 2010).

Sensors placed in soil can provide continuous, real-time, and remote monitoring (Lillesand and Kiefer, 2000; Reynolds et al., 2000) of field conditions (Garcia-Sanchez, et al., 2011). Using the data on salinity, moisture, and temperature collected using sensors, a simple decision support system (computer program) can accurately decide when and how much irrigation is needed (Duran-Ros, et al., 2008; Ray and Dadhwal 2000; Mishra et al., 2010). Remote monitoring as well as remote controlling could be achieved when sensors are integrated with wireless technology (Zade et al., 2005; Akyildiz et al., 2002). We have implemented soil moisture sensors to collect soil moisture and temperature data and automatize irrigation at Junagadh Agriculture University, India. This layout connects the automated drip irrigation system (Connolly and Little, 2010) with the decision support system (DSS).

Research shows that the farmers can save water and money by using integrated sensor-based irrigation methods (Sarma, 2002). If implemented, the proposed research can significantly
conserve resources such as water and electricity. This research was conducted under the Fulbright research program. In the following sections, we show the proposed a layout for completely automated field monitoring and controlling (Jongschaap, 2006) using sensors. We then show the results of the survey of the existing DSS software available in the market (commercial off-the-shelf: COTS) and identify their limitations. Then, we proposed design specifications of software to overcome these limitations. Then, we present the details of the DSS software that we have developed and implemented. Finally, we provide pointers to the future-research directions.

PROPOSED INTEGRATED-LAYOUT FOR WIRELESS MONITORING AND CONTROLLING

The problem with current automatized irrigation is lack of comprehensive, end-to-end integrated solution for automated irrigation. The current research enables automation of some parts of the data transfer from field sensors to the decision making process but the research leaves other parts manual. Figure 1 shows the proposed solution in which we provide the irrigation scheduling and monitoring via the Internet and cellular phone devices. The layout provides end-to-end solution from the field data to the Internet where the users have choice of manipulating the field data per their need without any additional cost. In the proposed system, the irrigation control is completely automated where the DSS makes the optimum irrigation scheduling.

Figure 1: Layout of the Automated System

Figure 1 illustrates that the field data from multiple soil moisture sensors are sent to a data logger which is also located on the field. Typically, the data logger sends these data to a personal computer using wireless technology such as GSM or GPRS (Aqeel-ur-Rehman, et al., 2011) although such transitions can be done using cables. Normally, the sensors come with application software that is installed on a computer to convert these data into readable format such as an Excel spreadsheet. In Figure 1, the arrow from the application software to the
spreadsheet shows this conversion. The program that we created provides a decision support system (marked a DSS in the figure) to turn the irrigation on or off. The DSS takes the spreadsheet as its input and outputs the decision. One of the major contributions of this research is the design and development of this DSS. Since the DSS automatizes the decision making, the system actually can be called an expert system. The decision taken can be transmitted to the Internet for remote monitoring (Lillesand and Kiefer, 2000). The decision can also be transmitted to field control system to start or stop drip irrigation system (Subbaiah, 2013; Patel, 1999). A mobile phone can also be used as a control and monitoring device for the irrigation system.

A PRELIMINARY SURVEY OF COTS FOR EXISTING SOFTWARE FOR IRRIGATION SCHEDULING

We surveyed existing software for irrigation scheduling. The following are some of the predominant scheduling software. For brevity, we are not including our analysis of these software here.


THE FAULTS AND LIMITATIONS OF THE COTS

In order to come up with a research design and objective, we thoroughly surveyed the irrigation scheduling software that is available in market today. Farmers, the end-users of the software product expect one software to fulfil their need of irrigation scheduling rather than buying and integrating multiple software packages. The user requirements can be conceptualized into the following categories:

- **Complete automation:** The user can control and monitor the irrigation scheduling from the front-end, which is a user interface, all the way to the back-end, which is the irrigation system installed in the field. Secondly, the software automatizes almost all irrigation-scheduling tasks so that the farmer has to perform minimal work to start and stop the irrigation. Thirdly, the monitoring should be on a continuous basis.
- **Code-and-data ownership:** The users have permission to edit the code so that the upgraded version incorporates any desired enhancement or meets their future needs. Secondly, the users should have the control and ownership of all the data they collect.
- **Ease-of-use:** The software is simple to use. For example, the user is not asked to input too much information before the user can start the irrigation scheduling.
- **Affordability:** The software is affordable.

Our survey of existing software concluded that this software did not meet the above requirements. In disparity with the above requirements, we found that the existing software had the following specific limitations:
The software does not provide the end-to-end solution. That is, they do not provide support from data from field to the decision making to taking appropriate irrigation action. For example, some software just provide decision making help but are not integrated with turning the irrigation system on and off.

Many techniques require that the farmers need to go to the field to collect some data.

Most analysis techniques provide only interment soil or crop information and fail to provide such information on a continuous basis.

The COTS help with only one type of field condition, such as mulching.

In COTS, the users do not have permission to modify/upgrade the code.

The COTS usually do not meet with 100% of user specifications.

The COTS we examined were highly complicated and required too much of use input.

Many software packages such as CROPWAT require its users to measure the amount of water supplied for irrigation, which adds to the burden for farmers.

Most companies do not give permission to send the collected data at your desired computer. The complete solution, such as the ones provided by Netafim, could be quite expensive.

In contrast with the above limitations, we decided to develop a DSS that:

- Fills a missing link between sensor output and automatic decision to turn irrigation on or off.
- Can be easily tailored for software for future needs.
- Can be tailored for specific needs.
- Provides continuous status of watering needs.
- Enables remote monitoring as well as controlling.
- Measures soil moisture at desired root-levels.
- Is simple to use and does not require too many input parameters.
- Does not mandate to measure water quantity.
- Works with different field conditions.
- Gives choice to send data at any non-proprietary server. That is, the users have a choice to send the collected data to their own computer.

**DSS DEVELOPMENT**

The process of software or system development follows well-defined phases of software development life cycle (SDLC) (Patel et al., 2014). These phases comprises of derivation of specifications and requirements, designing of the simulator software, coding, executing the software and collecting results, and finally maintenance. In the future research, we will provide more hardware details for the sensors and described how the programs were developed using SDLC. For brevity, we are eliminating these details here. However, we include the list of specifications that we developed for DSS as the following. These specifications were incorporated in our software in addition to the requirements listed in the above section:

- **Visual component:** We wanted to have the soil-tension data presented on the screen so that these data can be used for further analysis if needed.
- **Multiple mulching:** We wanted the software to analyze the sensor data for different types of mulching.
- **Multiple crops:** It was necessary that the software takes data from sensors installed for different types of plants.
- **Multiple root-depth:** We wanted the software to be able to incorporate data from sensors installed at different soil depth and aggregate them.
- **Types of models:** We implemented only one model where a hard-limit was assigned to turn on and off the irrigation system based on a certain value of soil moisture. However, wanted the flexibility to change the model to more complex model.
- **Simplicity:** Since the code needed to be simple, we chose the deterministic model because the
- **Flexibility:** We modularized the software so that it can model different scenarios. We anticipated that we would add and enhance the model at later time. A number of routines and procedures were designed maximizing the program modularity. We chose Visual Basic.NET since its object-oriented nature would enable us to write programs using reusable, modular objects.
- **Time Slices:** We designed to calculate average temperature per day. The data was collected at every three hours. Thus, total of eight data points were accumulated.
- **Aggregated data:** Although we used sensors at two root-levels, the programs are flexible in keeping them separate or aggregated.
- **Other parameters:** Keep the design flexible so that other features and parameters such as temperature could be incorporated in soil-irrigation decision making.
- **Portability:** For portability, we chose VB.NET which supports creating an executable file which that run on most computers.
- **Scalability:** In our simulator, we included a function that gives the user flexibility to choose a set of rows from the input Excel file. This function would later be used for enhancing the simulator for another case.
- **User-friendliness and convenience of use:** We provide the default input file-name and location. Users do not have to provide this information but have an option to overwrite the default file name and location.

**TESTING AND FIELD IMPLEMENTATION**

The Junagadh Agriculture University (JAU), India acquired smart sensors from i-Linc Technologies (http://www.ilinctech.com) and had implemented at a site. In collaboration with JAU, we examined the Digital Soil Moisture Recorders from Virtual Electronics Company (VCE: http://http://virtualweb.co.in/) and Irrrometer Watermark (http://www.irrometer.com/basics.html) sensors and finally selected VCE sensors. The sensor-based methods are also called graphical or bottom-line scheduling methods. These methods typically start with setting time-intervals of the soil-tension data, which can be reset later if necessary. Then, the data is collected. Finally, the time-intervals are increased or decreased depending on the steepness the intervals can be changed.

We used Visual Basic.NET (VB) for its simplicity and visual interface. The development included configuring VB for Web development environment and configuring the VB reference libraries to run the program. This program required that Microsoft Excel Object Library was added to read the data input file which was in Microsoft Excel format.

The actual field data was collected using the sensors from Virtual Electronics by the Department of Renewable Energy and Rural Engineering located at JAU. This data were passed to the DSS. The results of irrigation scheduling were also done manually to tally the DSS results.
Sensor data for various type of mulching were collected including silver & black plastic mulch, wheat straw, and crop without any mulching.

Unit testing and integration testing of DSS was completed successfully. That is, when a sample input file was supplied, the DSS calculated the irrigation scheduling correctly, which verified the integration testing. We also analyzed each module of DSS for unit testing. Input and output to each module was tested. Each data point was passed to the code modules and tallied with the results calculated manually. In addition, debugging code was inserted throughout the code which verified that each value at modular, routine, and functional levels was calculated correctly.

**FUTURE RESEARCH DIRECTIONS**

We suggest six specific future-research directions as following.

1. Currently, the decision criterion is to irrigate when soil moisture tension (Alves and Pereira, 2000) falls below 30. We plan to include a detailed model (Bastiaanssen and Ali, 2003). Plant available water (PAW) is a measure of amount of water readily available to plan. Soil moisture tension (Gontia and Tiwari, 2008), is converted to PAW using generalized graphs (see Figure 2). Using such detailed model, we plan to enhance DSS for its greater use.

![Figure 2: Typical soil moisture release characteristic curve (from Irrigation Guide, National Engineering Handbook)](image)

2. The last phase of software development life cycle includes maintenance in which, new requirements are produced and specifications are developed again. Through collaboration with JAU, we plan to use this software for two weeks in the field and examine the results. After using it for this period, we plan to add more specifications for the system and then update the software. We made these changes and further customized the software.
3. We plan to include the temperature into the model. Currently, the temperature data is collected by the sensors but we are not including as one of the factors to calculate the water need.

4. We plan to collect the data on how much water was saved after the developed DSS was installed. Currently, we do not have exact data on the amount of water and energy saved. Either field implementation will be installed by collaborating efforts with JAU, or simulated data from sample output files would be compared.

5. Further survey the systems. Survey/examine more DSS COTS. Based on the results, we will enhance the current version of DSS.

6. Add other features in COTS such as graphs, GUI, monthly/weekly reports, and so forth.

CONCLUSION

We have proposed an integrated-layout for wireless monitoring and controlling of fields for their irrigation needs. We identified the problems with existing irrigation-scheduling software, came up with specifications for new software, and then designed, developed, tested, and successfully implemented the DSS software with the real filed-data. The future research direction includes upgrading this DSS program with a more-intricate model and enhancing the software for its increased use by farmers around the world.

ACKNOWLEDGEMENT

This research was conducted under Fulbright core Academic and Professional Excellence Award for US Scholars for the year 2014-15.

REFERENCES


