International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:10, 2015

The Challenges and Solutions for Developing Mobile Apps in a Small University

Greg Turner, Bin Lu, Cheer-Sun Yang

***Abstract*—**As computing technology advances, smartphone applications can assist student learning in a pervasive way. For example, the idea of using mobile apps for the PA Common Trees, Pests, Pathogens, in the field as a reference tool allows middle school students to learn about trees and associated pests/pathogens without bringing a textbook.

While working on the development of three heterogeneous mobile apps, we ran into numerous challenges. Both the traditional waterfall model and the more modern agile methodologies failed in practice. The waterfall model emphasizes the planning of the duration for each phase. When the duration of each phase is not consistent with the availability of developers, the waterfall model cannot be employed. When applying Agile Methodologies, we cannot maintain the high frequency of the iterative development review process, known as 'sprints'. In this paper, we discuss the challenges and solutions. We propose a hybrid model known as the Relay Race Methodology to reflect the concept of racing and relaying during the process of software development in practice. Based on the development project, we observe that the modeling of the relay race transition between any two phases is manifested naturally. Thus, we claim that the RRM model can provide a *de fecto* rather than a *de jure* basis for the core concept in the software development model.

In this paper, the background of the project is introduced first. Then, the challenges are pointed out followed by our solutions. Finally, the experiences learned and the future works are presented.

***Keywords*—**Agile methods, mobile apps, software process model, waterfall model.

I. Introduction

AS computing technology advances, smartphone apps can assist student learning in a pervasive way. For example, with the availability of the PA Common Trees, Pests, and Pathogens System, middle school students learning about trees and associated pests/pathogens can take their smartphones (iPhones, Android phones, or Windows phones) to the field as a reference tool without bringing a textbook.

During the process of developing the three heterogeneous mobile apps, the project ran into numerous challenges. This paper describes the challenges and the solutions in the collaborative project between the Biology Department and the Computer Science Department for developing three smartphones apps including iPhones, Android phones, and Windows phones. In the remaining of this paper, the background of the collaboration is described first. Then the challenges are discussed. The solutions are presented.

Greg Turner is with the Biology Department, West Chester University, West Chester, PA 19383 USA (e-mail: gturner@wcupa.edu).

Bin Lu and Cheer-Sun Yang are with the Computer Science Department, West Chester, PA 19383 USA (e-mail: blu@wcupa.edu, cyang@wcupa.edu).

II. The Background

The idea of using an app on a smartphone as a learning tool was proposed to and accepted by the Pennsylvania Department of Conservation and Natural Resource (DCNR). This project proposed to develop some supplemental learning materials including three smartphone-accessible apps to compliment the PLT book "Common Trees of Pennsylvania" [5], [6]. Once available to PLT, these materials will help teachers better engage their students when learning about tree diseases and pest.

As a collaborative effort, the two departments, i.e., the Biology Department and the Computer Science Department began the software development process in a small university. This is a small teaching university with around 16,000 students and 800 faculty members without conferring a Ph.D. degree in the Computer Science Department. The developers were chosen from among the undergraduate students initially.

III. The Challenges

In the past, many researches study the challenges related to mobile software engineering. For example, some describe that the challenges may be caused by the inherent nature of developing mobile phone apps [10]. Others study the software engineering issues caused by the human aspects [3]. Very little research is focused on developing three heterogeneous mobile apps simultaneously in a software development shop such as a small university where the availability of developers is an issue. In this paper, our research focuses on the study of software engineering issues for the development of mobile apps especially in an environment where human aspects are involved. These challenges include: the human factors, the Mobile Application Software Development Life Cycle (MADLC) modeling as well as the detailed issues in the design, development, and testing phases. They are discussed below. The challenges are listed in Table I.

A. Human Factors

The first decision related to mobile applications is the type and platform for a mobile strategy. A mobile app can exist as a native application or a web application. A native app can operate at a location where the wireless Internet service or Wi-Fi, is not available. But the development of a native app must take into account the platform and the availability of developers. Once the decision to go for a native app instead of a web application; the issue of human factors [8] can impact the project planning drastically in a small university. Ideally, to develop three apps for three specific mobile phones respectively requires three separate teams. It is not feasible at

a development shop where the availability of developers is limited. For example, when some developers may graduate in one semester or two, the separation of software development life cycle becomes difficult. The situation can also occur in a development shop where the turnover rate is high.

TABLE I
THE CHALLENGES FOR DEVELOPING MOBILE APPS

No	Stage	Issues
1	Model	Which MADLC to use: Waterfall or Agile?
2	Schedule	Availability of Developer: How can the transition from one stage to another be ensured when the availability of developer may be at stake.
3	Schedule	How can scheduling be handled when three develop teams are not in synch?
4	Model	What is the model for developing three heterogeneous apps when the design can be shared?
5	Design	Which programming paradigm(s) should be used: structured programming or object-oriented, or others?
6	Coding	How can the prototype be used seamlessly for the development of the final code?
7	Data	What are the internal data structures?

B. Software Process Model Issues

A Mobile App Development Life Cycle (MADLC) was proposed in [9]. According to this paper, the MADLC includes the following stages: Identification, Design, Development, Prototyping, Testing, and Maintenance. The most noticeable point is that the prototyping appears after the development of modules. In our project, the life cycle is divided into Conceptualization, Design, Prototyping, Development, and Deployment phases. At each stage, software engineering principles are evaluated using the traditional waterfall model [7] and the more recent Agile Methods as published in the Manifesto for Agile Software Development [1]. The prototyping is used after the design stage for the purpose of the realization of the design. The challenge is to decide which model to use for mobile app development.

C. The Design Process and the Rapid Prototyping

The mobile app design and prototyping raises an issue when the duration exceeds four months, i.e., the duration of a typical semester. If so, the developers may graduate or become unreachable. Moreover, all three smartphones have inherent differences in terms of the user features such as buttons or screens. This is also the scenario in the industry when developers may become unavailable. What is the solution to the project management issue?

After the design of the user interface is determined, the individual developer begins to work independently on prototyping. The control of the project schedule is more complicated than that for a single homogeneous system. When three developers are working in different pace, the project scheduling becomes subtle.

When one developer is still working on the design of one app, another may begin the process of prototyping of another app. Neither the waterfall model nor the Agile Methods can be used to predict or model these behaviors since these two models are simply single-threaded for the development of various components in a homogeneous system.

D. The Development Process

For the development process, the traditional development model known as the Structured Programming Paradigm and the more recent Object-Oriented Programming Paradigm are two possible alternatives. Our ultimate goal is to maintain the same programming style for the consideration of readability and the maintainability. However, the three apps use different programming languages. For iPhone programming at the time of the development took place, the programming language is Objective-C. Another programming language, i.e. Swift, is announced in October of 2014. Android Phone App uses Java for the development. Windows App uses C# for the development. All three application development environments are unique. The development processes for all three types of smartphones are structurally different. The development process must take place independently. Maintaining the same style is challenging and probably not realistic for the development of three heterogeneous mobile apps.

The development stage is supposed to be simplified by the existing prototyping systems. In fact, the prototyping can be used as an aid for the user-interface design since it follows the principle of Rapid Prototyping. But the conversion of the internal data structure potentially may turn a dream into a fantasy. As indicated in Fig. 1, the internal data structure used for representing trees, pests, pathogens, and the associations is a number of arrays. To convert the representations to a database is a potential challenge. Without a proper strategy, the cost of conversion may be greater than a total rewrite.

```
//initialize plant table data

pest *pest1 = [pest new];
pest1.name = @"Hemlock Wooly Adelgi";
pest1.sciName = @"Adelges tsugae";
pest1.imageFile = @"pest_1.png";
pest1.details = [NSArray arrayWithObjects: @"Natural History: This insect was
    introduced to the U.S. from Asia and has severely impacted our â€œstate treeâ€.
    Signs of infestation include â€œwoolyâ€ egg sacs clustered near needles (hwa1)
    and pale green foliage.\n", @"Damage & Mortality: The small aphid-like insect
    (hwa2) is a sapsucker, feeding on phloem in young shoots, which weakens trees,
    making them susceptible to pathogens and other pests. Thousands have died, but
    some may be resistant, offering some hope that this tree may not disappear from
    all our forests. \n", @"Management:  Insecticides are injected into stems and
    travel to foliage where adelgids ingest it and die, but this is only practical
    in small areas. Biocontrol using insect predators has also been tested with
    some success, and may allow for widespread adelgid management in the future.\n"
    , nil];
```

Fig. 1 A Problem in The iPhone Prototype: Hard-Coded Data

IV. THE SOLUTIONS

The solutions are listed in Table II. In this project, the concept of Relay Race Methodology (RRM) is used for the general modeling method that is actually a hybrid model of Waterfall and Agile. The internal data structure is changed from the hard-coded style to a general relational database using the SQLite database manager. The availability of students are anticipated and arranged just as in the handoff of a relay race in a track-and-field game. They are discussed in detailed.

A. Human Factors

In our department, students can take additional options such as *Internship* or *Independent Study* as alternatives to taking regular courses. When developers are needed and funding is limited, these are good alternatives for identifying more

student developers. The scenario may be similar to the situation in the industry when turnover is anticipated. This practice leads to the solution to the mobile app software lifecycle described next.

TABLE II
THE SOLUTIONS FOR DEVELOPING MOBILE APPS

No	Stage	Solutions
1	Model	A hybrid MADLC is used to combine the macro view of Waterfall and the micro view of Agile Methods.
2	Schedule	Availability of Developer: Internship or Independent Study courses are used similarly to hiring contractors.
3	Schedule	Scheduling reflects a relay race method.
4	Model	A relay race model is proposed with the possibility of cooperation and the possibility of racing.
5	Design	Model-View-Controller paradigm is used as the structure of the various modules combining the concepts of structured programming, object-oriented programming, and event-driven paradigm.
6	Coding	The internal data structure is changed.
7	Data	An SQLite database with three tables are used.

B. Software Process Model Issues

The concept of Relay Race as in a track-and-field game is employed to prevent the interrupt of development. However, the availability of developers in summer is still challenging. This situation is manifested when the first runner in a relay race cannot hand off the baton effectively to the second runner. The model of Relay Race Methodology (RRM) requires principles and guidelines for achieving effective and seamless handoffs. The concept of the Relay Race Methodology [2] has been proposed in the past for the development of simulation and modeling. In our project, the RRM is employed for dealing with the human factors as well as for the study of effective handoff strategies and principles.

In the industry, the alternative is to hire some short-term contractors with the needed skills when the turnover rate is high. The metaphor of handoffs can be applied to the modeling of the real world situations.

In our project, the life cycle is divided into Design, Prototyping, Development, and Deployment phases. In each phase, software engineering principles are evaluated using the traditional waterfall model and the more recent Agile Methods as published in the Agile Manifesto. When the software development is based largely on the waterfall model, the transition from one developer to another is considered to conform to the Agile Model. The scenario of a relay race is in fact a hybrid model of the waterfall model and the agile model.

During our design of the user interface, the joint application development (JAD) [4] methodology is employed. All developers and the advisors work together with the "user", i.e., the Biology Department on the design of the user interface. The practice also complies with Agile Methodology as specified in the Manifesto for Agile Software Development [1]. Monthly meetings are held to review the design for the purpose of "customer collaboration" and "responding to change".

C. The Design Process and the Rapid Prototyping

After the design phase is completed, a Rapid Prototyping method is used as an aid to the conceptualization of the design. The database design totally transforms the structure of the project files. The database accessing is stored in the Model folder of the project, the code connecting the View and the Model is stored in the Controller folder of the project. The database is created by using the Firefox SQLite Database Manager, and later uploaded into the project. In Fig. 2, the tree table in the SQLite database model is illustrated. Figs. 3 and 4 illustrate the pest table and the profile table, respectively.

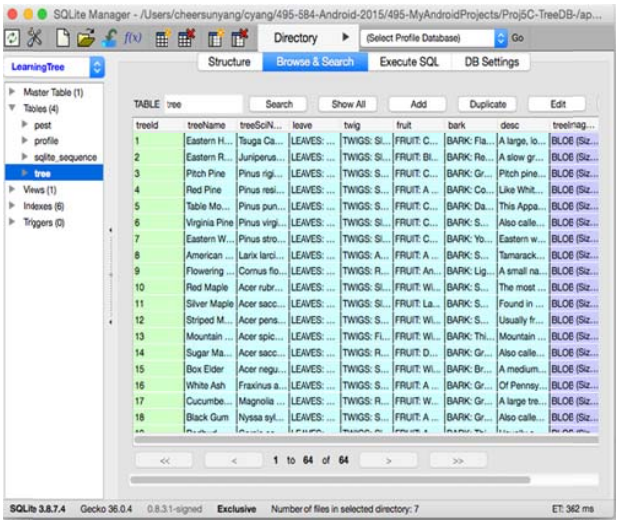

Fig. 2 Firefox SQLite Database: The Tree Table

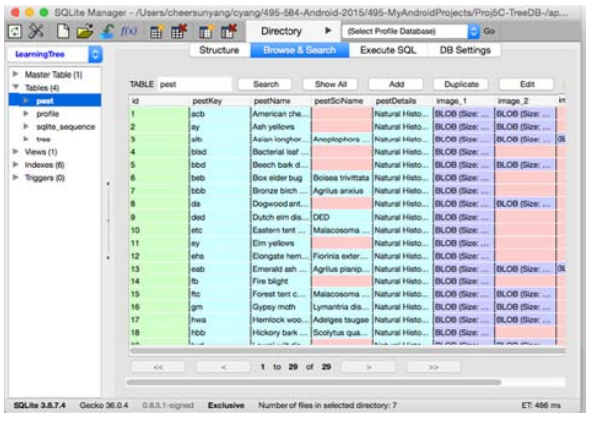

Fig. 3 The Pest Table

D. The Development Process - MVC

When structured programming and object-oriented programming are still general principles for all three apps, the identical style is not realistic. Our goal then is to maintain the same coding paradigm for readability. For the development phase, the *Model-View-Controller* paradigm is employed. The *Model* component of an app describes the structure of the database tables; the *View* component models the user interface layouts. The *Controller* component manipulates the adapting

between the model and the view components.

The MVC model is used in the final product as illustrated in Fig. 5 comparing with that in Fig. 1, the MVC model increases the readability and the maintainability for the project.

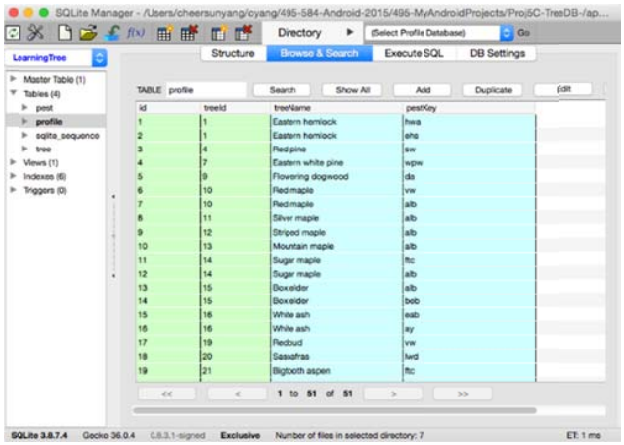

Fig. 4 The Tree-Pest Association: The Profile Table

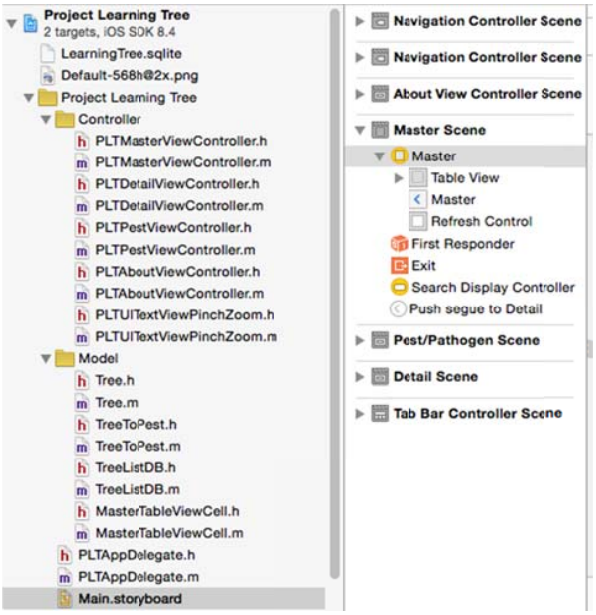

Fig. 5 The iPhone App Project Structure: MVC

V. The Final Product

A. The iPhone App

When the iPhone App is started, the screen displays a list of tree type and the search option as illustrated in Fig. 6.

If a user clicks on a tree type in Fig. 1, the detailed information about that tree species will be displayed as in Fig. 7. In addition, at the top right corner of the tree info screen, a "Pest" button is provided if there are pests/pathogens that attack this tree. Once clicked, a pest/pathogen screen will show the related pests or pathogens as in Fig. 8.

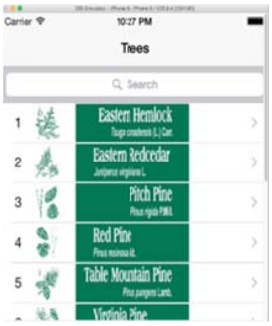

Fig. 6 iPhone App: The tree list

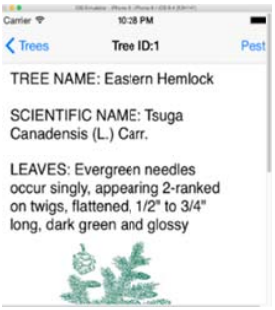

Fig. 7 iPhone App: Tree Details

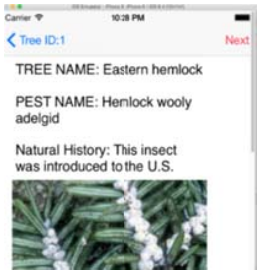

Fig. 8 iPhone App: Detailed Tree Information

Fig. 9 Android App: The Tree List

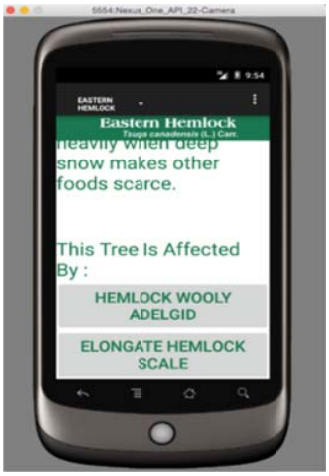

Fig. 10 Android Phone: The Pest/Pathogen Buttons

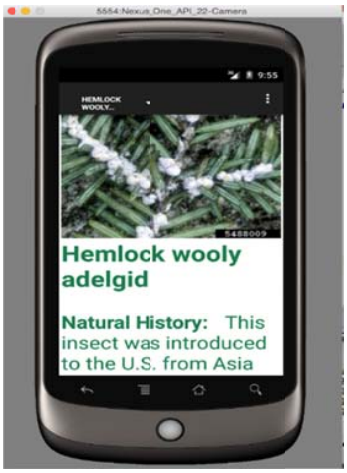

Fig. 11 Android App: Pest/Pathogen Information

B. The Android Phone App

The Android App consists of 12 Java modules, 12 layout XML documents, 6 activities, and 5 fragments. Each activity is associated with a User-Interface screen. When the App is started, the screen is displayed as in Fig. 9. When the tree info screen in scrolled to the bottom, more information will be displayed as in Fig. 5 including the scientific name, the trees, the fruits, the bark, and the pests/pathogens that attack the tree spices. A user can choose the pest/pathogen button to find more information as illustrated in Fig. 10. The pest/pathogen information will be displayed as in Fig. 11.

When the Tree Species option is clicked, a tree list menu and the detailed information about a tree type are displayed together. This capability demonstrates the merit of a big screen. When the tree list is scrollable, a user can retrieve the details about a specific tree type including the scientific name, its image, and the descriptions about the leaves, twigs, fruit, and the bark. Also, the list of pests/pathogens that attack the particular tree is displayed in yellow. A user can find the detailed information by choosing the button as in Figs. 12–14.

Fig. 12 Windows App: Menu

VI. The Experiences Learned

While managing the complex project such as this in well over one year and a half, we learn so many experiences. Some important questions are raised in practice.

A consistent view about software engineering principles is essential such as data modeling, database design, and the design principles for mobile devices with limited memory capacity.

The RRM model can only become an asset when each 'teammate' takes the responsibility for the assigned task within the specific period.

When the turnover rate of developers is expected to be high, the RRM can be more effective if the succeeding teammates can be identified as early as possible. However, this may not be possible especially in a small university environment when funding is limited. Poor transition can occur similarly to a track-and-field relay race. The 'baton' may be dropped and the effectiveness will be reduced. Hence, a consistent coding convention can help to reduce the transition cost. Good documentation can ease the burden of transition.

VII. Summary

In summary, the development of mobile apps demonstrates unique challenges to the practitioners of software engineering principles. The development of heterogeneous mobile apps in parallel also escalates the degree of complexity. This paper presents the challenges and our solutions to the software engineering principles. The project can be extended in the future to include some computer vision algorithms for the identification of tree types or pest/pathogen types. The framework and the experiences can also be applied to the development of other similar projects.

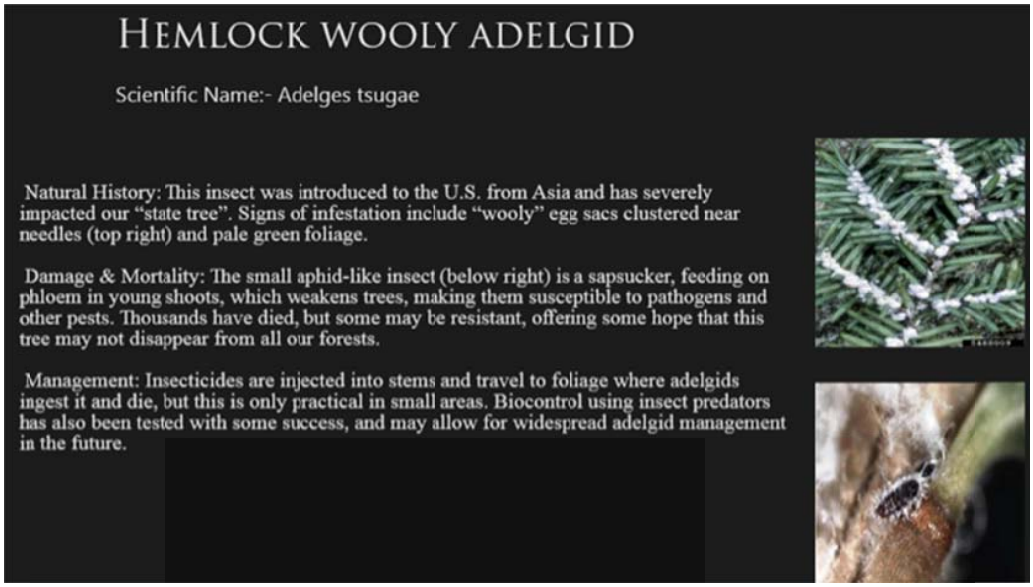

Fig. 13 Window App: Tree Info

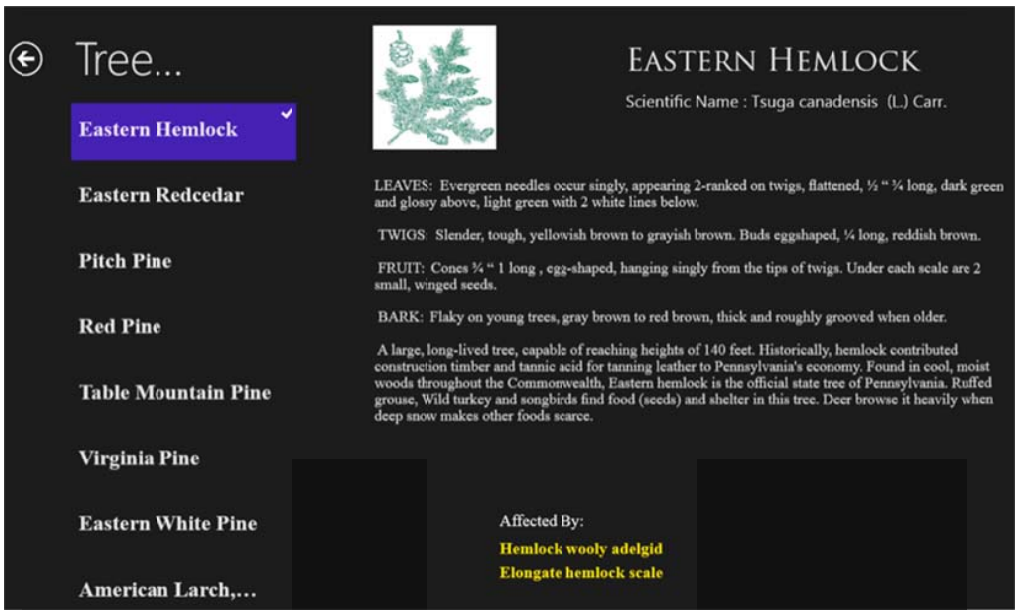

Fig. 14 Windows App: Pest/Pathogen Information

ACKNOWLEDGMENT

This project is partially funded by the Pennsylvania DCNR's Wild Resource Conservation Program. For their support, the authors are grateful, in particular, to Dr. Donald Eggen and Jean Devlin. Another PI in this project is Dr. Gerry Hertel (Biology). The authors would like to extend their thanks to the students involved in the project: Trevor Engle (Android App Development), David Grant (iPhone prototyping), Sai Krishna (Android App Database and Accessing), and Manoj Ram Tammina (Windows app and database development).

REFERENCES

[1] Beck, Kent et al., 2010. Manifesto for Agile Software Development. Agile Alliance, 2010.
[2] Evon M. O. Abu-Taeh, Asm Abdel Rahman El Shekh, Jehan Abu Tayeh, 2007. Simulation and Modeling: Current Technologies and Applications. IGI Global, Jordan, 2007, pp. 156-174.
[3] Orit Hazzan, E. J. Tomayko, 2004. Human aspects of Software Engineering. Charles River Media, 2004.
[4] Roman Soltys and Anthony Crawford, 1999. JAD for business plans and designs. DOI=http://www.thefacilitator.com/htdocs/article11.html. Last update time unknown. Accessed September 18, 2015.
[5] Pennsylvania Department of Conservation and Natural Resources (Pennsylvania DCNR), 2011. Common Trees of Pennsylvania. DOI= http://www.dcnr.state.pa.us/cs/groups/public/documents/document/dcnr_003489.pdf, 2011.

[6] Project Learning Tree in Pennsylvania, 2015. Pennsylvania Department of Conservation and Natural Resources (Pennsylvania DCNR). DOI= http://www.dcnr.state.pa.us/forestry/education/projectlearningtree/index.htm.
[7] Royce, Winston, 1970. Managing the Development of Large Software Systems. Proceedings of IEEE WESCON 26, 1970, pp. 1-9.
[8] Saqib Saeed, (ed.). Human Factors in Software Development and Design. IGI Global, 2014.
[9] Tejas Vithani, Anand Kumar. 2014. Modeling the Mobile Application Development Lifecycle. Proceedings of the International MultiConference of Engineers and Computer Scientists, 2014.
[10] Wasserman, A. I., 2010. Software Engineering Issues for Mobile Application Development. FoSER 2010, Santa Fe, New Mexico, USA.