Haaga-Helia
University of Applied Sciences

Machine Learning and Dynamic User Interfaces in a Context Aware Environment

Nathaniel Ham

Bachelor's Thesis
2015

Haaga-Helia
University of Applied Sciences

Abstract

December 3, 2015

Author(s)

Nathaniel Ham

Degree program

Business Information Technology

Report/thesis title	**Number of pages and appendix pages**
Machine Learning and Dynamic User Interfaces in a Context Aware Environment	**23 + 2**

The increasing usage of smartphones in daily life has received considerable attention in academic and industry driven research to be utilized in the health sector. There has been development and dissemination of a variety of health-related smartphone applications. However, to date, there is few to none applications based on nurses' historical or behavioral preferences. Mobile application development for health care sector requires extensive attention to security, reliability, and accuracy. In nursing applications, the users are often required to navigate in hospital environments, select patients to support, read the patient history and set action points to assist the patient during their shift. Finally, they have to report their performance on patient related activities and other relevant information before they leave for the day. In a working day, a nurse often visits different locations such as the patient's room, different laboratories, and offices for filing reports.
There is still a limited capability to access context relevant information on a smartphone with minimal recourse such as Wi-Fi triangulation. The WI-FI triangulation signals fluctuate significantly for indoor location positioning. Therefore, providing relevant location based services to a mobile subscriber has become challenging.

This paper addresses this gap by applying machine learning and behavior analysis to anticipate the potential location of the nurse and provide the required services. The application concept was already presented in IMCOM 2015. This paper focuses on the process to ascertain a user's context, the process of analyzing and predicting user behavior, and finally, the process of displaying the information through a dynamically generated UI. The case study application is designed, developed, and assessed in Finland based on the mobile learning usability and user experience (mLUX) framework.
To make the machine learning and behavior analysis anticipate or confirm the nurse's location more efficiently, we have applied various technological solutions. In the first attempt, we utilized Knime server and in latest implementation we applied Python with Scikit-Learn to increase the speed and to use primarily open source solutions. This paper describes the development process, applied methodology, our technological experiment results and the user evaluation experience. This research result is applicable to software development methodology, practitioners, and academicians.

Keywords

Mobile Application; Machine Learning behavioral analysis)

Table of contents

List of Tables and Figures

Abbreviations

AJAX	Asynchronous JavaScript and XHTML
CANA	Context-Aware Nurse Assistant
CSV	Comma Separated Values
GPS	Global Positioning System
HMI	Human Machine Interaction
HTTP	Hypertext Transfer Protocol
MAC	Media Access Control
NFC	Near Field Communication
RFID	Radio Frequency Identification
SSID	Service Set Identifier
SVM	Support Vector Machine
UCD	User Centered Design
UI	User Interface
WCF	Windows Communication Foundation

1 Introduction

In today's hospitals, nursing staff are finding themselves increasingly stressed due to various reasons such as over-working and under-staffing (P. Van Bogaert, H. Meulemans, S. Clarke, K. Vermeyen, and P. Van De Heyning 2009). As work-related demands continue to increase, nurses must be outfitted with the most current technology in order to best complete their job quickly and efficiently. Research shows (M. Prgomet, A. Georgiou, and J. I. Westbrook 2009) that hospital staff would greatly benefit from a mobile application that not only aggregates a majority of the data and services required for daily tasks, but also provides context-aware capabilities based on the high mobility and dynamic nature of the job. Having automatic predictions and form generation would reduce the work load of filling reports and allow nurses to spend more time on patient care.

The Context-Aware Nurse Assistant (CANA) project aims to develop an application that is both mobile and context-aware to address the issues faced by the nursing staff. We have previously presented a conceptual design of the application (M. Dirin, A. Dirin, and T. H. Laine 2015). In this paper, we describe the design and implementation of a context-awareness module through which CANA can assist nurses by providing context-sensitive information in any work context.
In order to predict what information would be most useful to a nurse in any given context (location, time, current activity), a machine learning (E. Alpaydın 2014) algorithm will be employed. This algorithm will be trained by a nurse's past behavior, and based on the training data it will learn what resources are most useful to the nurse when a certain action is performed. This will simplify Human Machine Interactions (HMI) (J. Cannan and H. Hu 2011) and allow the nurse to focus primarily on the current patient.

After the machine learning algorithm makes its prediction, the information will be sent to another process which is in charge of serving up the predicted information dynamically in a way that best meets the nurse's usability needs and requirements.
In addition, in order to adapt to each user as a unique individual, there will be a second machine learning server that will create forms and display information dynamically. This will allow each user to have their own customized UI that conforms to their changing usability needs.

In this paper, there is a literature review section which overviews the pros and cons of geolocation methods for determining context, machine learning algorithms for making predictions, and algorithms for developing a dynamic user interface.

The application is being developed in cooperation with Haaga-Helia UAS located in Finland, Ajou University in South Korea, and HES-SO UAS in Switzerland. From the private sector, consulting is provided by UXIT Consulting.

2 Objectives and Scope

The objectives of this paper are:

1) Find a quick and cost effective approach to determining a user's context while indoors. In order to properly implement context-aware services, this system must have a reliable approach to determining the user's physical location through which other context data can be inferred. For outdoor locations, there already exists a wide variety of open source satellite-based technologies that can be leveraged. However, satellite-based technology can be unreliable indoors and alternate approaches must be adopted to ensure consistency.
2) Determine an effective machine learning algorithm for predicting user's behavior patterns based on the context provided from the objective above. The machine learning service must be capable of handling a large amount of user data. After analyzing the data, the service must then be able to make accurate and quick predictions on the user's behavior.
3) After having made an accurate prediction on anticipated user behavior, this project aims to dynamically generate a personalized UI with information that is immediately useful based on the user's current context.
4) After completing the above objectives, the goal is to implement these objectives into a singular, cohesive proof-of-concept application.

Throughout the entire process, it is a goal for the CANA project to adhere to User-Centered Design concepts in order to create an application that meets and adapts to a user's usability needs. To that effect, when at all possible, test users will be incorporated into the testing process to get immediate feedback on newly implemented features.

3 Background Study

3.1 Determining user context/Context aware technologies

One of the key components of the CANA project is the ability for the application to determine the user's physical location. Knowing the user's physical location is a large part of a user's current "context". In order to properly supply the machine learning algorithm with helpful data, great care must be taken when implementing the location determining service.

For outdoor geolocation, there are a wide range of solutions that provide a great deal of both speed and accuracy available under open source licensing. However, determining a user's location becomes more difficult when that location is inside a large building, such as a hospital. The metal and concrete in large structures interferes with the GPS signal and the service will be unreliable at best, and impossible at worst.
In order to work around this limitation, there are several alternatives that can be leveraged to still provide accurate, quick and reliable geolocation.

3.1.1 GPS Dead Reckoning

GPS dead reckoning is a modern incarnation of techniques used in navigation aboard naval vessels. The main concept relies on taking an initially known position, and making an educated guess on the user's current position based on initial speed and direction (L. Ojeda and J. Borenstein 2011). This method can be used to fill in gaps in traditional GPS services, which frequently occurs in large buildings. This method would conceivably work by taking a user's initial location, speed and bearing at predetermined locations and extrapolate the user's future location with a series of calculations and educated assumptions.

The main issue with GPS dead reckoning is its high level of uncertainty, as all of the results are at best, "and educated guess" (S. Beauregard and H. Haas 2006). While this does have the possibility of being correct and accurate, it also has the possibility of being very inaccurate. As hospitals can be quite busy, it would be troublesome to properly calculate the average length of time it would take to get from point A to point B. In addition, each individual possesses a different pace length and speed. Calibration would be required to properly determine the actual speed of the nurses while in transit between locations.

Since this method would extend the usage of existing GPS features on mobile devices, no extra sensors or devices would have to be installed. This makes the method reasonable from a cost perspective. However, developing algorithms to track and predict user movement would prove both costly and also time consuming for those responsible for the implementation.

3.1.2 Wi-Fi Positioning

Wi-Fi positioning (C. Yang and H. Shao 2015) systems are ideal for indoor environments where traditional GPS access is limited or not available. This method leverages the increase in availability of wireless access points in most public and commercial locations. A user's location is determined by the resulting strength of the wireless signal(s) at their current position. As with traditional triangulation techniques, the more reference points being used, the more accurate the location measurements will be (F. Evennou and F. Marx 2006). In order to properly employ a wireless access point, a virtual map of sorts will need to be constructed that has the location of each wireless access point, along with its SSID and MAC address. With a carefully laid out virtual structure, a user would then continue their activities as normal, while the mobile device would track its position in relation to the access points currently available.

One of the benefits of implementing this technology is that many, if not most, modern buildings already have wireless access points installed. Although additional access points would still have to be installed to ensure greater accuracy, this would reduce the initial cost and complexity of the installation. Additionally, virtually all modern mobile devices are standardly equipped with wireless access features. Again, this makes this option more attractive as new sensors or devices would not have to be purchased to run the app at a user level.

As accuracy determines heavily on the presence of stable access points, there are several factors that can reduce the accuracy of the triangulation calculations. The first factor that affects positional accuracy is signal strength/availability. In such a system, the presence of humans will greatly affect the signal propagation and therefore strength. On a slow day at a hospital, the available signals will have much more strength as there are fewer bodies to absorb signals. But at a time when there are many people present at the hospital, the signal will be absorbed and behave in a much different way. This would give a user false readings as the signal would not behave in a predictable or reliable manner. The cost for this technology depends greatly on the size of the facility in question and the existing wireless infrastructure. If a hospital already has an extensive wireless network, additional installation costs may be marginalized. However, regardless of building size,

the location of each wireless access point will still have to be determined. As most hospitals tend to be large in nature, the costs involved for installation could be considerable.
In order to turn signal strength and wireless access into a meaningful location, the application must run calculations and algorithms to produce the user's position (C. Yang and H. Shao 2015). These calculations contribute to the complexity of implementing this technology. All of the above mentioned factors of signal strength must be taken into consideration and accounted for. During the installation phase, precise calibration and testing would have to be done to determine the location of the wireless access points in relation to each other. This feature would make the technology less appealing to a client as implementation would require a larger cost and complicated algorithms.

Overall, the Wi-Fi positioning method should be considered primarily for hospitals with a large existing wireless network and where a high degree of accuracy is not required. Hospitals of any size could also consider it if they find that human traffic stays relatively consistent (although, again, high accuracy cannot be guaranteed)

3.1.3 Near Field Communication

Newer generations of smart phones and mobile devices have begun to include NFC (Near Field Communication) sensors to enhance networking and inter-device communications. Unlike its predecessor, RFID (Radio Frequency Identification), NFC tags support encryption and are thus much more secure (G. Madlmayr, J. Langer, C. Kantner, and J. Scharinger 2008). As hospitals deal primarily with private personal information, such a technology would lend itself very useful in protecting location and sensitive information. Another feature of NFC is its short range. While this may at first sound like a disadvantage, it again supports security based on confidence of location. For example, it is not possible to access a specific NFC tag without being in basic proximity to it (0.2 m) (V. Coskun, B. Ozdenizci, and K. Ok 2013). In application, this technology would work by placing NFC tags at physically meaningful locations, such as one per department, or where more accuracy is required, one per patient room or lab. When a nurse enters the desired area, she would need only to scan her device over the tag and the app would automatically update its location based on the tags predetermined location in relation to the hospital.

As with wireless capabilities, most modern mobile devices are equipped with NFC capabilities. This would mean no special installation would be required client-side. Since the technology is established, major mobile operating systems (Android, iOS, Windows Phone, etc…) provide open APIs to implement these features.

In keeping with the cost constraints, NFC tags have a relatively low cost (approx. 1 Euro per tag at the time of writing). Since there is no cost for client-side sensors or installation, a hospital could conceivably install the NFC tags for under 100 euros, depending on their size and intended level of accuracy.

A particular downside to utilizing the NFC tags is that the nurse would have to physically introduce the mobile device to the tag. While not a particularly complicated gesture, it does require the nurse to remember to swipe the device when location is changing. The complexity of installing the NFC tags amounts to registering the tag id number and assigning it to a relative location in the hospital. Installation of the tags physically is as simple as attaching a sticker to a surface.

3.2 Analyzing and Predicting User Activity with Machine Learning

3.2.1 What is machine learning?

Machine learning is a specific discipline under artificial intelligence that aims to implement algorithms that can programmatically learn from an input source and provide predictions as output. This will allow the algorithm to adapt to changing datasets automatically without input from the user(P. Domingos 2012).

3.2.2 Supervised vs Unsupervised Learning

In machine learning, there are two main categories, supervised and unsupervised learning. The first category, supervised learning, gives the algorithm a specific set of instructions along with a definition of what the predictions should look like (A. Ng 2012). For example, the algorithm can be shown a data set containing a time of day, weather and a sales amount. The algorithm then knows the format it can expect the data to be in, and what an example of a “good” prediction is. This method is useful in situations when the data to be analyzed is consistent. In unsupervised learning, none of the input data is labeled, and the algorithm must decide what the appropriate output should be based on data clustering from the example data (Z. Ghahramani 2004). This method is useful in scenarios when there would be a wide range of acceptable predictions. A commonly used example would be facial recognition. For the CANA application, supervised learning will be used to help the predictive service determine which outcomes we are looking for and provide it with examples of data.

3.2.3 Classification vs Regression

Another subset of machine learning is classification and regression (Y. Baştanlar and M. Özuysal 2014). A regression algorithm deals with continuous numeric data analysis, while classification handles nominal labels. Calculating sales figures based on the time of day and weather would be an example of regression, while predicting what movie a user should watch based on past favorite movies would be a classification problem (S. B. Kotsiantis 2007). In the case of the CANA project, the service will need to be able to predict a specific activity for the user. This would be an example of classification as nominal values are being used.

3.2.4 Common Classification Approaches

Naïve Bayes

A Naïve Bayes classifier calculates the probability that a specific outcome will belong to a certain class (H. Zhang 2004). In this context, naïve simply means that the algorithm assumes that all input variables are completely independent and do not influence each other in any way. While one of the more simple solutions, Naïve Bayes classifiers are highly scalable and are often one of the most cost effective solutions.

Decision Tree Learning

Decision Tree Learning has a fundamentally different approach than the Naïve Bayes. This approach seeks to divide all of the training data into sub-categories until the data either matches the prediction or does not match the prediction. Each “tree” is split according to possible decisions (J. Su and H. Zhang 2006). Each node represents a decision to be made, while each branch represents an option. Each end node will represent a classification that can be predicted.

SVM (Support Vector Machines)

A support vector machine is a non-probabilistic binary linear classifier (W. S. Noble 2003). An SVM splits the data along a hyperplane and effectively divides the data into two sets (binary classifier).

3.3 Generating Dynamic User Interfaces

Once the user’s context has been successfully determined and the next activity predicted, the application needs to display the information in a dynamic and meaningful way . The generation of content will be an extension of the work done by Michael Dayern (2015).

4 Methodology

4.1 User-Centred Design

The term User-Centered Design (UCD) originated by Don Norman during 1980s after publication of book named: User-Centred System Design, New Perspective on Human-Computer Interaction (Norman and Proper, 1986). UCD together with the development of interactive systems and devices have an increasing importance in product development organizations (Nieminen, 2004). In addition, UCD is a common method for developing smart products. It has been argued that in a usable system, we need to involve users' continually, and based on user's feedback, modify the design (Gould and Lewis, 1985; Gouldand Boies, 1997). UCD cuts both costs (Bosert, 1991) and (Gulliksen et.al, 2003) and improves usability, since it continually focuses on the essential needs of the customer as early as possible. The user's requirements are the focus in all stages of the product development cycle. Human-centered design (ISO 9241-210, 2010) processes for interactive system define three different design solutions for UCD as: (i) Cooperative design where designers and users involved in all stages; (ii) Participatory design where users' occasionally participate in the design process; and (iii) Contextual design that is based on the actual context. The UCD phases are as follows: (i) Get to know the users; (ii) Analyze user tasks and goals; (iii) Establish usability requirements; (iv) Prototype and then design a concept; (v) Test usability of the concept; and (vi) Repeat the stages as there are more features/services.

4.2 Concept Development Methodology the mLUX Framework.

For the CANA application concept development, we applied the mobile application development framework (mLUX) (A. Dirin and M. Nieminen 2015). The mLUX framework consists of three layers: 1. The role-players, 2. The process, and 3. The context of use. Figure 1 presents the overview of the adaptive learning application development framework.

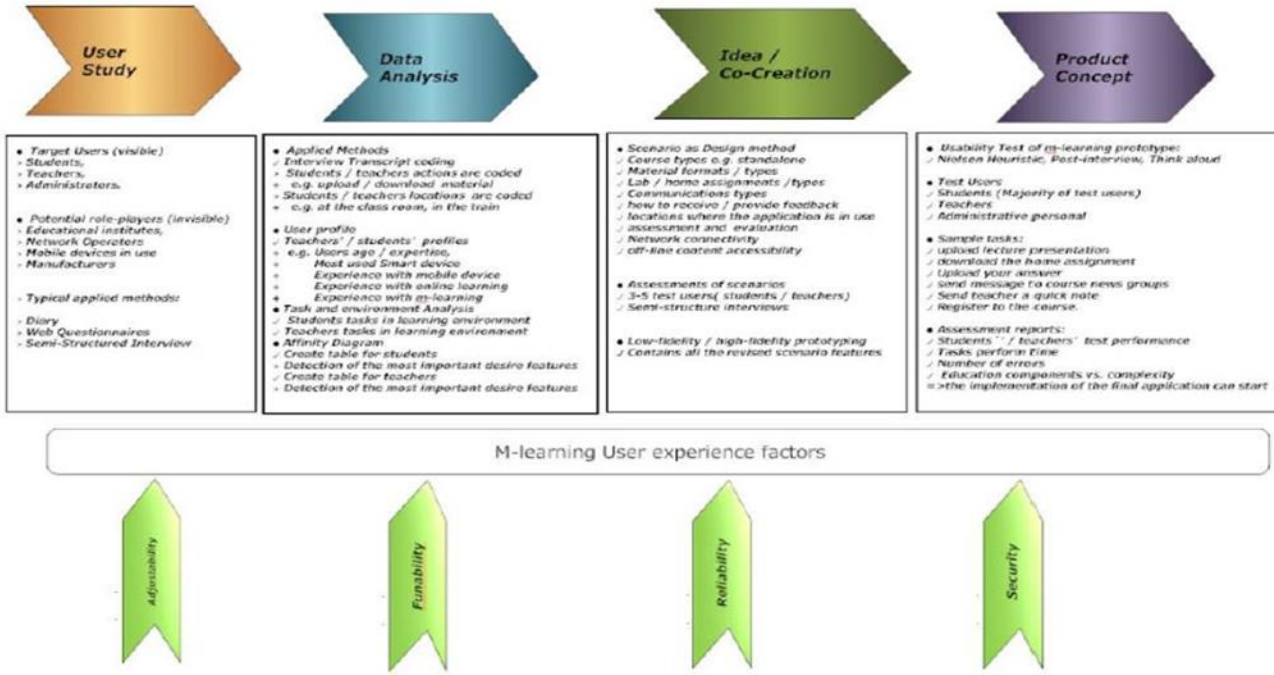

Figure 1. MLUX development framework

4.2.1 User Study phase (Understanding the user)

At this phase the stakeholders of the CANA application are identified and their real needs are analyzed by applying various user study methods such as questionnaires and interviews. The user study methods help the designers to learn about the users and their environments in which they often interact. Standard UCD methodology recommends 3-10 target users' involvement at various user study stages (Usability Research Group, 2002).

4.2.2 Data Analysis phase (Identifying the interactions)

At this phase the gathered data are analyzed by applying various data analysis methods such as transcript coding, user task and environment analysis and affinity diagram to classify and categorize the users' requirements.

4.2.3 Idea Creation phase (Producing design Solution)

At this phase we need to confirm the categorized and prioritized requirements with the potential test users. This confirmation ensures that designers understood the users' needs properly. Additionally, this is yet another opportunity for target users to impact the proposed requirements. We present the categorized requirements as scenarios. Scenarios

are proved to be an appropriate approach that we promote in the mobile learning application concept design method, as the scenario speaks user's language and avoids technical terms and complexities.

4.2.4 Product Concept (Evaluating the designs)

In this phase, the application concept is modified based on the gathered data in the previous phase. The application concept is now ready to be implemented as a non-functional prototype. The prototype consists of the potential user interface components and the navigations of various screens. In the last stage of users' involvement in the design process, we conduct usability evaluation test for the proposed application prototype. The target application design refinement is based on the usability test results.

The UCD for mobile learning application is an iterative design method. In this method, the concept development mandates the users' involvement at all phases of the mobile learning application development, which minimizes the application failure and maximizes the penetration of the application among the target users.

4.3 Methodology Implementation

The CANA project was started to specifically address the needs of nurses working in hospitals. The goal was to have university students work both on the development process while also conducting research. One of the major focuses of the CANA project was to develop a solution that conforms to UCD principles. Because of this, there was close cooperation with healthcare professionals in both design and user testing phases.

During the requirements phase of the development lifecycle, the scope and objectives were set by the product owner and only revised minimally based on feedback from user testing.

In the autumn of 2014, an initial proof-of-concept was created for the user interface and was tested in a hospital. At the time of this writing, the user interface is available in both Finnish and English.

5 Implementation

5.1 Architecture

The following sections detail the architectural aspects of the CANA application.

5.1.1 Hardware Requirements

At this stage of development, the mobile client is designed to run on Android devices. CANA's prediction server relies on a Linux server environment, but has the capability to port to a Windows server environment.

5.1.2 Software Requirements

The mobile client is optimized for Android devices. The predictive services and HTTP server run on Python 3.5 in 64-bit environments.

5.1.3 Database Model

During the prototype phase of the project, the machine learning service will rely on a simple CSV file to store and retrieve user behavior. The CSV file is hosted on the application server for easy access by the prediction service.

5.1.4 Physical Architecture Model

The client side features of the CANA application run on a mobile device running an Android OS. The device's NFC reader is also facilitated anytime the device is introduced to an NFC tag. This introduction to an NFC tag deploys a call to the server side features.

5.1.5 Application Architecture Model

The NFC feature of the CANA application runs on the mobile client as a background process. This process will be triggered anytime the mobile device is in proximity of an NFC tag. The NFC tags will be programmed with the hospital's room number. When the process is triggered, it sends an http request to the machine learning service residing on the server containing the user id, time of day, and current location of the user.

The prediction and analysis of user behavior is located on the server and will handle client requests with standard http requests. The process will require the client to provide the current user id, the time of day, and the current location of the user. This data is then put into nominal form for the algorithm to properly make the prediction. Next, the application will load previous stored data into memory and fit the data into a decision tree. At this point,

the client's data is then processed according to the previous results and the most likely result is sent to the client. Finally, the process updates the database with the most current user action.

When the client receives the result of the prediction, it will match the result provided from the server with the corresponding form. The UI is then appropriately updated with the information that the user is looking for. As the user continues to use the form, the database is updated dynamically via AJAX methods.

A WCF client is planned to handle all server requests. This will run automatically in the background and will be responsible for fetching predictions and informing the user of UI updates. The WCF will make the query to the machine learning service in order to provide a secure connection. This will ensure that only users with proper access rights can access the sensitive data. Figure 2 outlines the structure of the architecture.

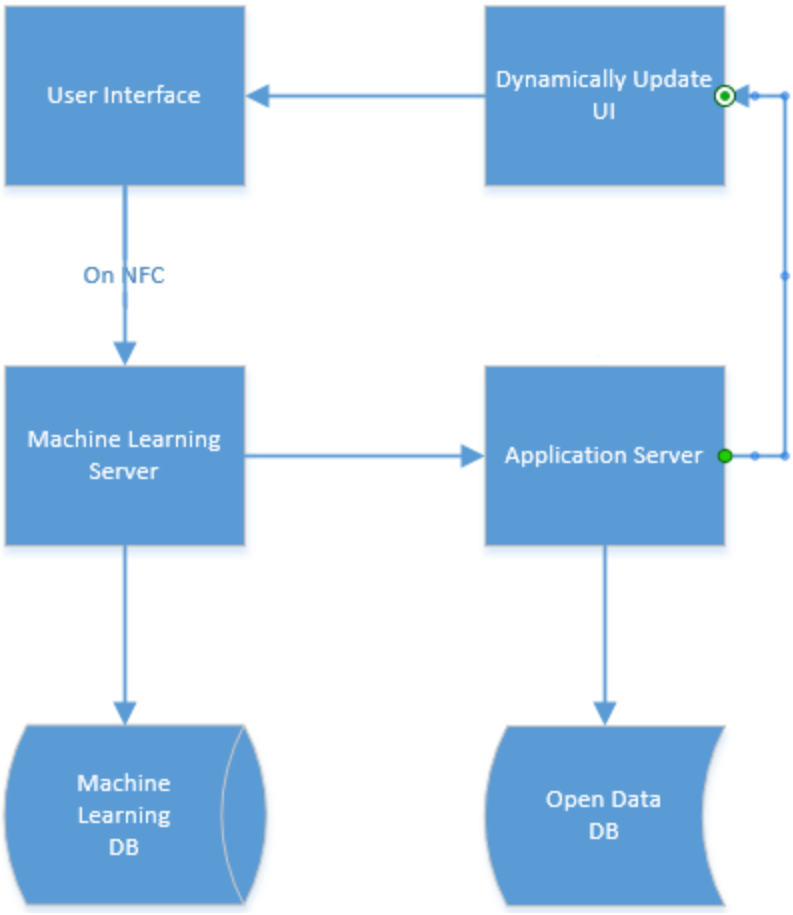

Figure 2. Overview of Client and Server

5.2 Technology

The following tools and technologies were implemented in throughout the development process of the CANA project.

5.2.1 User Positioning

In order to effectively and reliably determine the user's position indoors, the following approaches were used.

Hardware

NFC Tags

In the first two implementations, NFC tags were used to determine a user's location and context. When a user would scan their mobile device to the NFC tag, the CANA project would send the NFC tag's id (which corresponds with a physical location in the database), the current time, and the user id.

Android Phone

At this phase in the development cycle, the CANA project is being developed solely for Android devices. The device being used for testing is a Samsung Galaxy SIII and runs Android OS version [version number].

Software

PhoneGap/Android SDK

The open source distribution of Cordova, PhoneGap, was used in conjunction with the Android SDK to deploy solutions to an Android device for testing. Using PhoneGap allowed for the use of NFC libraries written in JavaScript. By using the JavaScript libraries, more focus could be put on testing and development, instead of spending time writing native Java code. PhoneGap also provides CLI tools for deployment which were used to package solutions and transferring them to a mobile device. In addition, Phonegap offers platform dependent support and would allow for the same project to be deployed for iOS, Windows Phone, Android or Tizen.

NFC Library

Chariot Solutions (ChariotSolutions 2015) provides an open source library compatible with PhoneGap that gives access to several key NFC reading features. In the case of the CANA project, the library was used to access the phone's NFC reader. It was on top of this library that the CANA project built its NFC capabilities.

5.2.2 Predicting User Behavior

In order to predict the user's behavior, the following technologies have been used.

Software

Amazon Web Services (AWS)
Amazon Web Services provides remote computing services, allowing for an implementation of cloud services. AWS was chosen as our cloud server because an academic version was available. In addition, the server is flexible and allows for different versions of Python to be installed and configured. Since the machine learning was written in Python version 3.4 (64-bit), it was imperative to have a server environment with the same version. Finally, AWS also offers SSH connections which allowed a degree of customization for web services and HTTP hosting.

Scikit-Learn
Scikit-learn is a library built in Python with a wide range of tools focused on machine learning. Scikit-learn was chosen because it offers solutions for algorithms in Naïve Bayes Classifiers, decision trees and support vector machines (SVMs). Using Scikit-learn to implement the algorithms allowed more focus to be placed on the proper application of the algorithm. In addition, Scikit-learn provides graphical features for generating images to represent prediction results and show a graphical implementation of the decision tree.

Numpy
Numpy is an extension of the Python programming language and allows for n-dimensional arrays which are useful for handling large sets of data. In the CANA project, Numpy is used to efficiently create arrays and store the data in. This provides an efficient manner for storing the data currently being used in memory once it has been loaded from the data source.

Flask
In order to handle HTTP requests and responses from both the server and the client, Flask is used on the server side to simplify this process. Flask simplifies the URL routing and will handle user authentication during the prototype phase of the CANA project. Flask is intended only for development stage and a more robust HTTP Python server will be used during actual deployment.

Anaconda

Anaconda is a free distribution of Python which bundles a wide variety of popular Python packages for ease of install and configuration (Anaconda 2015). This included, Scikit-Learn, Numpy, and Flask.

5.2.3 Displaying Dynamic Form Information

Hardware

Android Phone

As with the NFC portion of the CANA project, an Android phone is used to dynamically display the information predicted by the server.

Software

The software portion of generating the user interface will be also handled with Python on the server side. On the client side, Phonegap will again be used to handle requests and responses from the server and updating of the user interface.

5.3 Implementation Process

At the time of this writing, there have been three implementation phases.

5.3.1 First Approach

The first iteration was focused on generating meaningful test data that could be used to test the validity of the machine learning algorithms. It was important to generate data that reflected the actual daily activities at a hospital. This would ensure that the data are not too random for the algorithm to provide meaningful predictions. The data was generated to simulate a nurse's ordinary shift. For the purposes of the testing, a shift is defined as an eight hour period that can take place during one of three times: 7am – 3pm, 3pm – 11pm, and 11pm – 7am. During this stage of testing, in order to limit scope, the nurse will be responsible for five rooms and perform routine duties throughout the scheduled shift. Generating data in this way will give the machine learning algorithm a chance to develop a basis for normal behavior patterns. For testing purposes, the data is stored in standard CSV files which are then read and written to by the application. Figure 3 shows a sample of the data collected in the file. In the final production environment, the CSV format will be replaced with an SQL server/database environment.

	A	B	C
1	RoomNumber	TimeOfDay	Activity
2	1	290	1
3	1	393	2
4	2	396	1
5	2	399	2
6	3	402	1
7	2	405	2
8	4	408	1
9	4	411	2
10	5	414	1
11	5	417	2
12	1	420	5
13	2	430	5
14	3	440	5
15	4	450	5
16	5	460	5
17	100	480	2
18	100	480	2
19	100	495	2

Figure 3. Sample of User Behavior Data

5.3.2 Second Approach

In the second iteration, each separate portion of the CANA application was developed on its own environment. At an early stage, it was important to establish that each individual technology was feasible and capable to implement within scope.

User Positioning

The user positioning portion of the application was built specifically for Android mobile platforms. In order to focus on rapid implementation, PhoneGap was used to deploy the code to an Android environment. Since PhoneGap was used, the development for the NFC reader was done using JavaScript.

For testing purposes, NFC tags were programmed with id values to represent room numbers. This was to demonstrate that a mobile device could effectively and quickly scan a tag and display its information.

Predicting User Behavior

User behavior was predicted using Python's Scikit-learn library. Scikit-learn provides an interface to implement a variety of common machine learning algorithms. A decision tree algorithm was chosen to handle the interpretation of the user behavior and providing an

accurate prediction. The machine learning server receives an http request from the client and proceeds to make a prediction based on the request data.

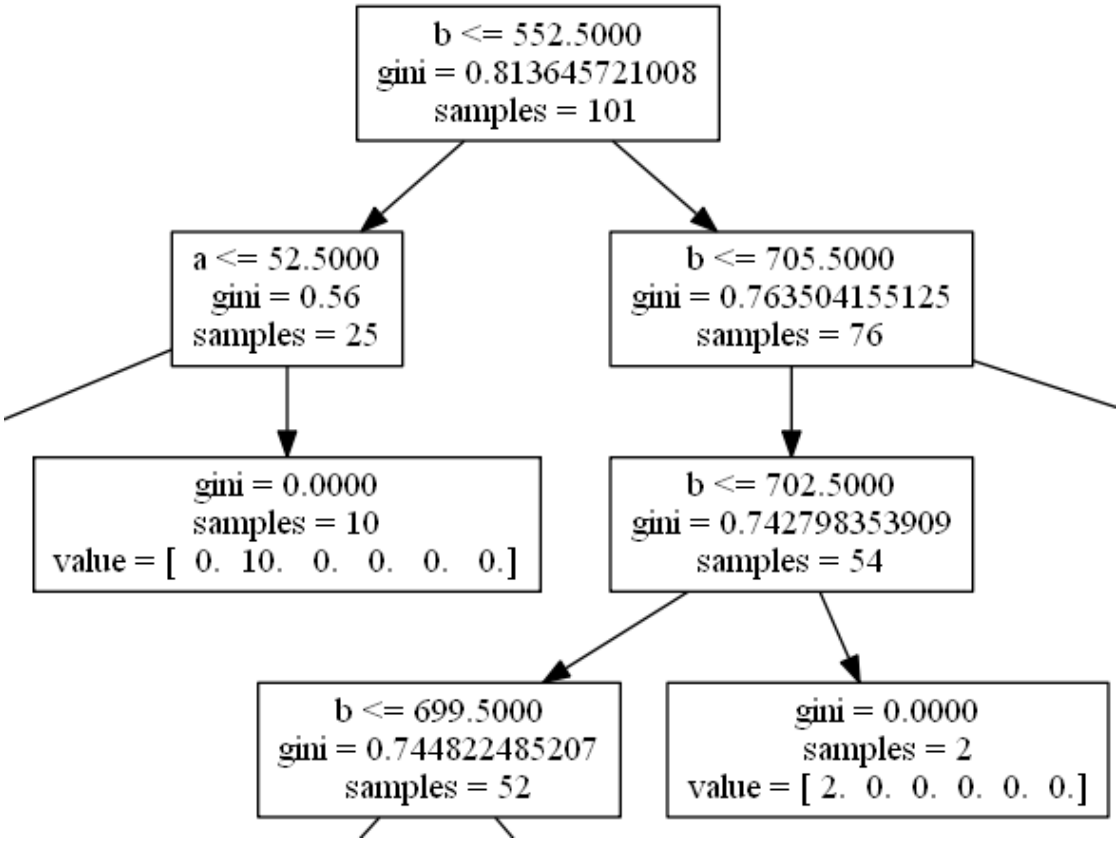

Figure 4. Sample of Decision Tree Data

The data is split automatically according to key features. Each end node on the tree represents a possible outcome. At this stage of development, there are six possible predicted outcomes. The end nodes represent a clustering of data points that match the criteria of a specific outcome. This can be seen in the "value" array in Figure 4. The position of the integer determines which prediction is represented, while the value of the integer shows how many samples are present.

Generating Dynamic User Interface

Once the user's behavior has been predicted, the CANA application will return the predicted action back to the client UI. This is done via a simple HTTP response. The response is handled by the client, which will then display the corresponding form. By updating the UI's current form automatically, this allows the user to focus on the task at hand, rather than on manipulating the UI itself.

5.3.3 Third Approach

In the third and final iteration, the focus was on synthesizing each of the individual aspects of the CANA application into one working unit. With all of the elements being deployed, proper testing could be conducted. This allowed a more general, top-level, view to be taken of the CANA application. Finding a suitable server environment to deploy on was crucial at this stage.

As the different portions of the CANA application were being brought together, it was important to properly design the architecture. Determining user positioning was labeled as a client side feature, while predicting user behavior and displaying dynamic form information were labeled as server side features.

5.4 Test Design

In this phase, the testing focused on how the application handled as a complete environment. In consideration to scope and budget constraints, the testing was conducted in a closed environment to demonstrate a proof-of-concept. Establishing the testing environment involved placing NFC tags around a room. Each tag is clearly labeled as if it were a hospital room number. For the purposes of this test, they were labeled numbers 101 – 105 consecutively. This allows for the simulation of a hospital environment for the test users. The test users are given a list of tasks to perform in several of the rooms.

Users are asked to perform a list of various tasks within the simulated testing environment using a provided mobile device. The users' performance is measured based on their ability to complete their tasks and the time taken to complete the task. At the conclusion of the tasks, the users are asked about their overall impression of the application flow and the accuracy of the predictions.

The tasks are as follows:

1) Go to room number 101 and scan the tag
2) Check/Update Patient Measurements
3) Go to room number 103 and scan the tag
4) Check/Update Patient prescriptions

Follow-up questions:

1) Was the application flow intuitive and easy to use?

2) How accurate were the predictions?

3) Were the correct forms displayed in a meaningful way?

6 Assessment and Analysis

Testing was conducted with test users in a closed environment to assess the viability of the proposed methods. The results aim to demonstrate the accuracy of the predictive capabilities as well as test the viability of the entire CANA prototype. The test was conducted with six users and their results were stored in Microsoft Excel format. Figure 5 shows the layout of the testing responses.

	User 1	User 2	User 3
Successfully Launch App	Yes	Yes	Yes
View All Patients	Yes	Yes	Yes
Scan Tag for room 101	Yes	Yes	Yes
Check/Update Patient Measurements	Yes	Yes	Yes
Was Prediction Accurate?	Yes	No	Yes

Figure 5. Sample of User Testing Results

As the tests were conducted in a closed environment, the accuracy levels will not completely reflect results taken in a working environment. Although the data was generated to reflect an average work shift, there will be deviation.

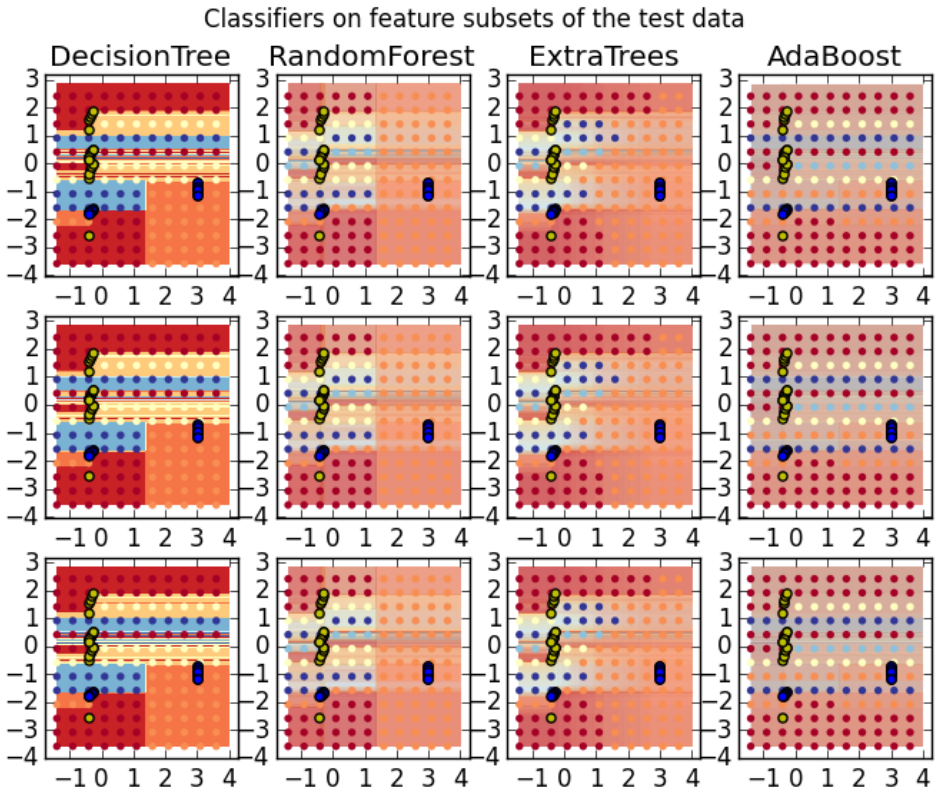

Figure 6. Distribution of Prediction Accuracy

For the users that were tested, the machine learning service provided 83% accuracy with an average run time of 31 milliseconds. The distribution of predictions is graphically represented in Figure 6. The accuracy is based on the end user's experience with the prediction service. Further testing will allow the prediction service to get better, more accurate results as it learns as the data grows.

Considering the level of accuracy and speed of the prediction, the testing phase is being viewed as successful. However, during future implementations, work will be done to further optimize the machine learning algorithm to better fit the data sets. One factor that limits the accuracy of the prediction is the scope of the test data itself. In order to have the most accurate predictions, it is necessary to have a broad range of user behavior stored. Once a normal pattern has been established, the prediction service will have a better model to make predictions.

7 Conclusion

The goal of the CANA project is to deliver a reliable context aware, predictive solution to be deployed in a hospital environment. The current proof of concept prototype demonstrates the CANA application's ability to accomplish the goals established in the Scope. This prototype is capable to provide contextual cues to a predictive service which will then supply the user interface with the appropriate form.

For the initial stages of the application, Knime services served as a starting point, but licensing fees would have been outside of the project's budget. Therefore, by developing the machine learning aspects of the application in Python, we were able to avoid any of the licensing fees attached with using Knime server in a production environment. In addition, Python showed an increase in speed while allowing us an added degree of flexibility working with analyzing the data.

Future Work

The next step for the CANA project is to apply for partners and funding to support further development. This will allow proper testing to be conducted in a live hospital environment. In addition, this would allow for more members to be brought into the project to aid the development and optimization process. The project will require work to be done to implement an interface for the hospital's patient record database. This will also require that diligent steps be taken to provide a secure environment to protect patient's privacy.

More work will need to be done to improve the accuracy of the predictions provided by the application. This will include optimization of the current decision tree or the implementation of a completely new approach such as a neural network classifier.

In the future, connection with the machine learning service will be based on a WCF client. This will allow all server communication to be asynchronous. Utilizing WCF will also ensure secure data connections to further protect patient data as only users with appropriate access rights will be granted access to the service.

References

A. Dirin and M. Nieminen, “mLUX :Usability and User experience development framework for m-learning,” iJIM, Int. J. Interact. Mob. Technol., 2015.

A. Ng, “1. Supervised learning,” Mach. Learn., pp. 1–30, 2012.

C. M. C. C. M. Bishop, “Pattern recognition and machine learning,” Pattern Recognit., vol. 4, no. 4, p. 738, 2006.

C. Yang and H. Shao, “WiFi-based indoor positioning,” IEEE Commun. Mag., vol. 53, no. 3, pp. 150–157, 2015.

E. Alpaydın, “Introduction to machine learning,” Methods Mol. Biol., vol. 1107, pp. 105–128, 2014.

F. Evennou and F. Marx, “Advanced integration of WiFi and inertial navigation systems for indoor mobile positioning,” EURASIP J. Appl. Signal Processing, vol. 2006, pp. 1–11, 2006.

G. Madlmayr, J. Langer, C. Kantner, and J. Scharinger, “NFC devices: Security and privacy,” ARES 2008 - 3rd Int. Conf. Availability, Secur. Reliab. Proc., pp. 642–647, 2008.

H. Zhang, “The Optimality of Naive Bayes,” Proc. Seventeenth Int. Florida Artif. Intell. Res. Soc. Conf. FLAIRS 2004, vol. 1, no. 2, pp. 1 – 6, 2004.

J. Cannan and H. Hu, “Human-Machine Interaction (HMI): A Survey,” Gesture, pp. 1–16, 2011.

J. Su and H. Zhang, “A Fast Decision Tree Learning Algorithm,” in 21st national conference on Artificial intelligence - Volume 1, 2006, vol. 5, no. Quinlan 1993, pp. 500–505.

L. Ojeda and J. Borenstein, “Personal Dead-reckoning System for GPS-denied Environments,” in IEEE International Workshop on Safety, Security and Rescue Robotics, 2007, pp. 1–6.

M. Dirin, A. Dirin, and T. H. Laine, “User-Centered Design of a Context-Aware Nurse Assistant (CANA) at Finnish Elderly Houses,” in The 9th International Conference on Ubiquitous Information Management and Communication, 2015.

M. E. Tipping, “Sparse Bayesian Learning and the Relevance Vector Machine,” J. Mach. Learn. Res., vol. 1, pp. 211–245, 2001.

M. Prgomet, A. Georgiou, and J. I. Westbrook, “The Impact of Mobile Handheld Technology on Hospital Physicians’ Work Practices and Patient Care: A Systematic Review,” J. Am. Med. Informatics Assoc., vol. 16, no. 6, pp. 792–801, 2009.

P. Domingos, “A few useful things to know about machine learning,” Communications of the ACM, vol. 55. p. 78, 2012.

P. Van Bogaert, H. Meulemans, S. Clarke, K. Vermeyen, and P. Van De Heyning, “Hospital nurse practice environment, burnout, job outcomes and quality of care: Test of a structural equation model,” J. Adv. Nurs., vol. 65, no. 10, pp. 2175–2185, 2009.

S. B. Kotsiantis, “Supervised Machine Learning : A Review of Classification Techniques,” Informatica, vol. 31, pp. 249–268, 2007.

S. Beauregard and H. Haas, “Pedestrian dead reckoning: A basis for personal positioning,” POSITIONING, Navig. Commun., pp. 27–35, 2006.

T. M. Mitchell, Machine Learning, no. 1. 1997.

V. Coskun, B. Ozdenizci, and K. Ok, “A survey on near field communication (NFC) technology,” Wirel. Pers. Commun., vol. 71, no. 3, pp. 2259–2294, 2013.

W. S. Noble, “What is a support vector machine?,” Nat. Biotechnol., vol. 24, no. 12, pp. 1565–1567, 2006.

Y. Baştanlar and M. Özuysal, “Introduction to machine learning,” Methods Mol. Biol., vol. 1107, pp. 105–128, 2014.

Z. Ghahramani, “Unsupervised Learning BT - Advanced Lectures on Machine Learning,” Adv. Lect. Mach. Learn., vol. 3176, no. Chapter 5, pp. 72–112, 2004.