Database Management System for Monitoring the Effect of Climate on Wildlife Interactions
Progress Report 1

NCSU College of Veterinary Medicine
Dr. Suzanne Kennedy-Stoskopf
Dr. Lauren Charles-Smith

CSC 492 Team 12:

Bo Chulindra
Chris King
Scott Gentry

North Carolina State University
Department of Computer Science
Senior Design Center

September 28, 2011

I. Sponsor Background

Dr. Suzanne Kennedy-Stoskopf and Dr. Lauren Charles-Smith of the NCSU College of Veterinary Medicine are conducting research in order to determine the effect climate has on wildlife infectious diseases. They currently have collars on various wildlife in order to collect data such as the animal's locations throughout the day, their home range, the time spent in proximity to other collars, and more. This data is also combined with topographical maps and weather station data to give a more detailed picture of the collar's environment.

- Dr. Suzanne Kennedy-Stoskopf, Research Professor of Wildlife Infectious Diseases
  suzanne_stoskopf@ncsu.edu

- Dr. Lauren Charles-Smith, Fisheries, Wildlife, and Conservation Biology
  drlaurencharles@gmail.com

II. Problem Statement

The sponsors are collecting data from animal collars and weather stations, as well as maps.

This data presents the following problems:

- **The researchers don't have a way of automatically linking data together.** For example, interaction data, or data of where and for how long two collars have been within a certain range of each other, is important to the researchers. The researchers don't have a system that calculates interaction data.
- **The researchers can't share data with other researchers very well.** Data is retrieved from the collars in one format, and then may be saved in some other format. Because of this, two researchers working with different equipment will often have their data in different formats.
- **The researchers want to see their data visualized in ArcGIS.** In order to get their data in ArcGIS correctly, the researchers have to use some tool, or create on themselves, that imports their data.

III. Project Goals & Benefits

Our project goal is to create a system that manages the collar and weather station data, integrates it with map data and ArcGIS, and provides a UI for researchers to analyze portions of the data. This will save researchers time and facilitate analysis with less errors.

IV. Resources Needed

The following is a list of the hardware and software our team (and any follow-up team) needs for the project. The list has been separated into hardware and software we already have, and those that we need.

Resources we have:

- Dedicated Server
- ArcGIS 10
- Pivotal Tracker
- GitHub
- Linux OS
- Python
- Hudson

Resources we need:

- PyCharm Licensing
- Backup Storage for the database

VI. Methodology

Our team has chosen to develop using Agile, with two-week iterations. We are reporting to Dr. Lauren Charles-Smith every two weeks with our progress. For testing, we have a suite that is run automatically every time we commit.

VI. Requirements

The following are a list of our requirements, broken up into functional and non-functional, and then broken up into components and the requirements for each of those components.

Functional Requirements:

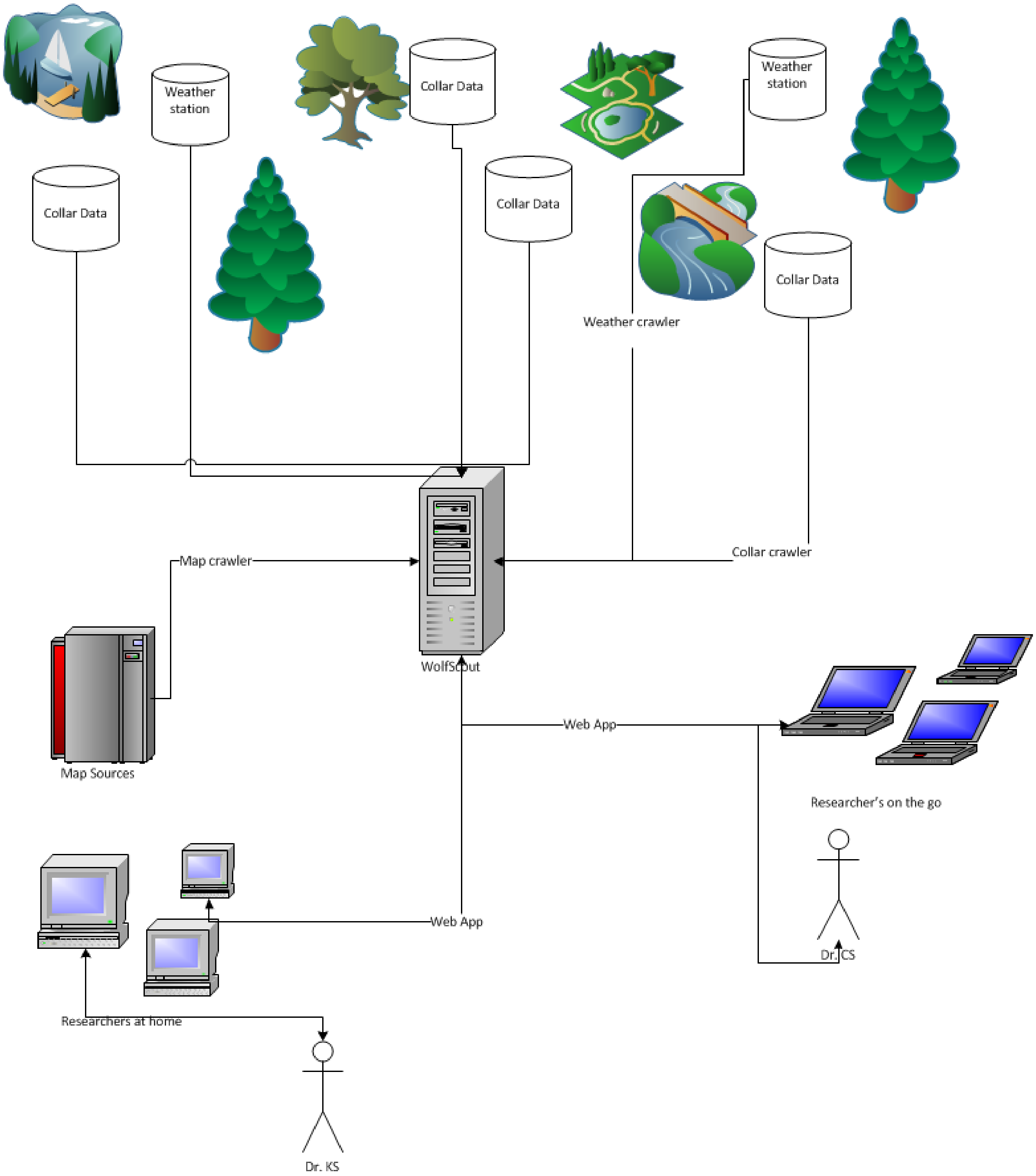
- Graphical UI
    - Web app or stand-alone GUI
    - Retrieves the following data from the database (via the server):
        - Collar data (temperature, time, location)
        - Climate data
    - Integrates with ArcGIS 10
    - Authenticates the user
- Database

    - Meets OpenGIS standards
    - Links data according to their time and locations
- Server
    - Reads the following data from a file:
        - Collar data
        - Weather station data
    - Stores data in and serves data from the database
    - Calculates the following:
        - Interaction data
        - Home-range

<u>Nonfunctional Requirements:</u>

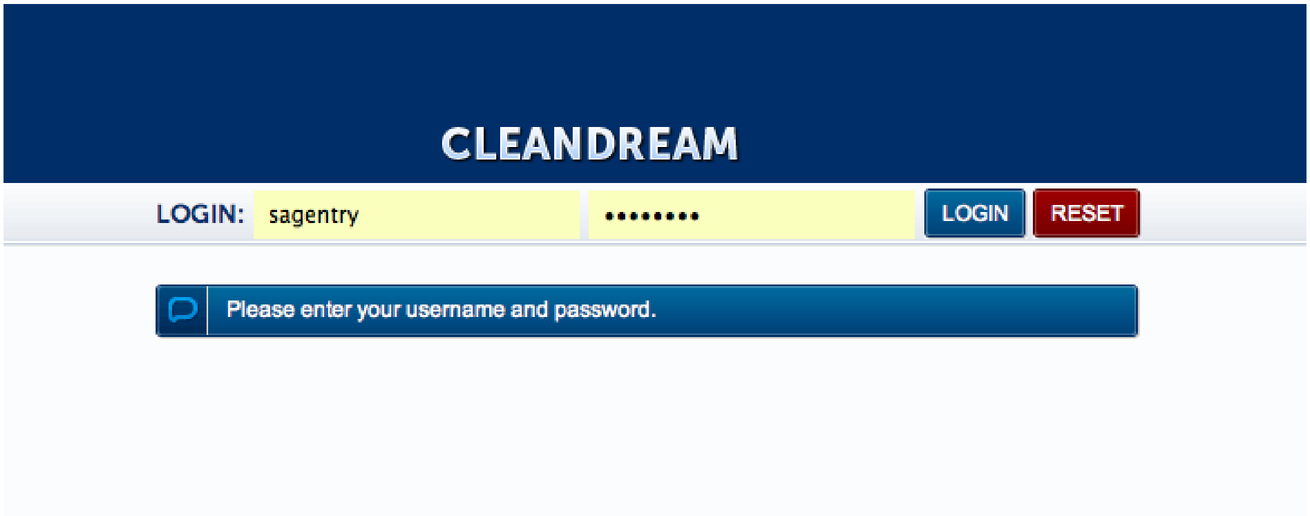
- Database
    - Can handle 2 million GPS data points per month
    - PostgreSQL 8.4
    - PostgreSQL-8.4-PostGIS
- Server
    - Can handle 50-60 users at once
    - If fails, can be brought back up within a day.
    - Ubuntu 11.04+
    - Python 2.7.x
    - Supporting Packages:
        - Django==1.3
        - Fabric==1.2.0
        - PIL==1.1.7
        - South==0.7.3
        - coverage==3.5.1b1
        - django-nose==0.2
        - gunicorn==0.13.1
        - logilab-astng==0.22.0
        - logilab-common==0.56.1
        - nose==1.1.2
        - nosexcover==1.0.7
        - paramiko==1.7.7.1
        - psycopg2==2.4.1
        - pycrypto==2.3
        - pylint==0.24.0
        - unittest2==0.5.1
        - wsgiref==0.1.2
- Documentation
    - Non-trivial code is documented
    - Administrative server management is documented

VII. Design



Above is a (overly simple) diagram of our overall architecture. The server, in the center and named WolfScout, grabs all the collar data, weather station data, and maps with crawlers. Users can then access that data via a web application.

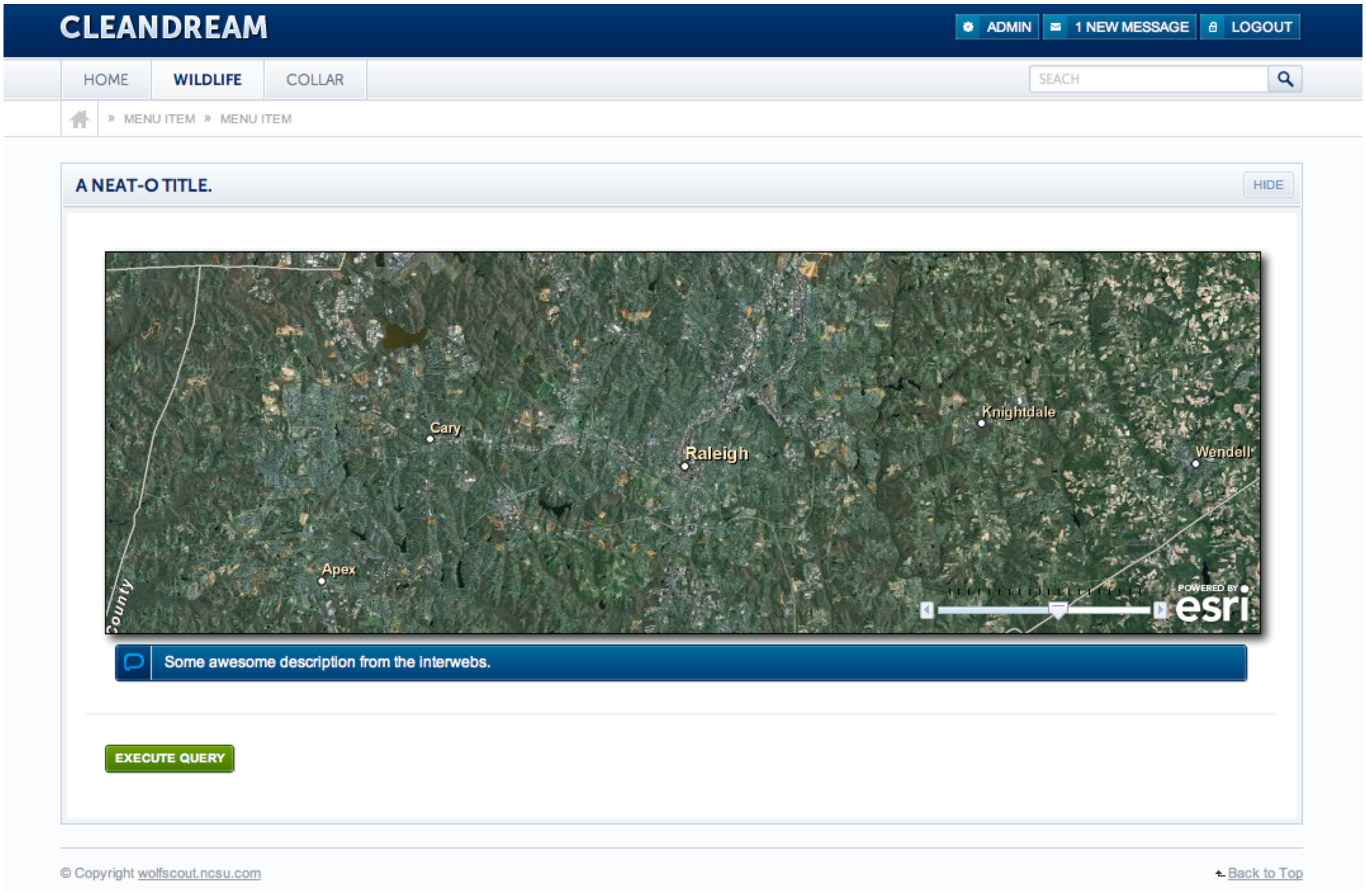
Below are early looks at the UI. Here's how the login page for the web app may look...



LOGIN: sagentry ●●●●●●●●●

Please enter your username and password.

And here's how the web app may end up interacting with maps.



Some awesome description from the interwebs.

© Copyright wolfscout.ncsu.com

Back to Top

VIII. Implementation:

Code:

- crawler/
  - **collar.py**: Reads in collar data from a text file, parses it, and inserts into DB
  - **views.py**: Pulls collar data from the DB and outputs it in HTML
- general/
  - **views.py**: Renders the login page to the user
- wildlife/
  - **model.py**: Provides a model for collar, species, and specimen data. This model can be used to pull data from the DB.
- **fabfile.py**: Provides methods to handle common source control actions such as commit, running tests, and updating local code.
- wolfscout/
  - **README.md**: Provides instructions for new developers to set up their environment. Can also be viewed by visiting https://github.com/NCSU-VSR/wolfscout and scrolling down until README.md is displayed.

Development environment:

- GitHub
  - Source code is stored on GitHub at *github.com/NCSU-VSR/wolfscout*
- Server
  - Deployed on Senior Design lab machine
- Database
  - On Senior Design lab machine

IX. Test Plan/Cases

Automated tests are written for collar.py. Our test plan is to have 80+% code coverage, which means that at least 80% of our code will have an automated test that steps through that code. All tests are run and must pass before any code can be committed or merged in the repository. We use a tool called Fabric to help enforce/facilitate this.

In addition to the automated tests, we have the following test scripts which will be manually run through periodically during development. We have not set a schedule for testing yet because the there is still not enough workable code to merit it.

Note that the test scripts below are not complete. As the system gains more function, more tests will be added.

Test ID	**Description**	**Expected Results**	**Actual Results**
login	Preconditions: You are not logged in to the UI. The server is up and running. 1. Open the UI home page.	You are shown the login page.	
login_validUser	Preconditions: You are not logged in to the UI. The server is up and running. User ‘user’ and password ‘password’ is registered in the system. 1. Open UI home page 2. Enter username ‘user’ and password ‘password’. 3. Submit.	You are logged in.	
login_invalidUser	Preconditions: You are not logged in to the UI. The server is up and running. User ‘bad’ is not registered in the system. 1. Open the UI home page. 2. Enter username ‘bad’ and password ‘bad’. 3. Submit.	You are not logged in. You are still shown the login page. You are shown a message that explains that login failed.	
login_CSS	Preconditions: You are not logged in to the UI. The server is up and running. 1. Open the UI home	An alert does not appear.	

	page. 2. Enter the following for the username and password: ‘<<SCRIPT>alert("XSS");//<</SCRIPT>’. 3. Submit.		
login_SQLInjection **Not yet implemented. Tests SQL Injection[2]**			
login_noScript **Not yet implemented. Tests the web UI with javascript turned off.**	Preconditions: You are not logged in to the UI. The server is up and running.		
createUser_CSS **Not yet implemented. Tests creating a user using cross-site scripting[1]**	Preconditions: You are logged in to the UI. The server is up and running. 1.		

[1] see http://en.wikipedia.org/wiki/Cross-site_scripting

[2] see http://en.wikipedia.org/wiki/SQL_injection

XI. Task Plan

The following is our task plan for Iteration 1 (ends 9/28) and 2 (ends 10/12). In addition to these task items, we have user stories in Pivotal Tracker. If you don’t have access to our project in Pivotal Tracker and would like it, please contact a member of our team and we will see about getting you access.

Item	**Owner(s)**	**Due Date**	**Status**

Iteration 1	All	09-18	Done
Crawl/Store collar data	CK	09-18	Done
* Encapsulate bad collar data	CK	--	Done
* Enable manually entering notes per collar & CK	CK	--	Done
* Integrate collar data with UI	CK	--	Done
* Write tests	BC	--	Done
Rough draft of WPR 1	All	09-16	Done
* Include tutorial	BC	--	Done
Final draft of WPR 1	All	09-28	Done
Learn HTML + Python (Scott)	SG	09-18	Done
Learn Python (Bo)	BC	09-18	Done
Contact wildlife folks	CK	09-14	Done
Contact CSC IT	CK	09-15	Done
Iteration 2	All	09-28	Started
Establish SQLite	CK	09-28	
* Postgres	CK	--	Started
* PostGIS	CK	--	Started
* GeoDjango	CK	--	
* 5 utilities	CK	--	
Hard code data into DB	CK	09-28	
* Write tests	BC	--	Started
Place Task Plan in Cubicle	BC	09-12	Started
UI (Scott)	SG	09-28	Started
* UI Prototype	SG	--	Started
* Standalone ArcGIS	SG	--	
* Web based (wildlife)	SG	--	

* Different ways to manipulate data	SG	--	
OPR 2	All	10-03	Started
* Slides	All	--	Started
* Presentation	Scott	--	