Campus Maps

Colin Hunt, Tri Lai,

Edwin Chung, Shane Murnaghan,

Mark Galloway, Josh Stagg, Kris Kushniruk

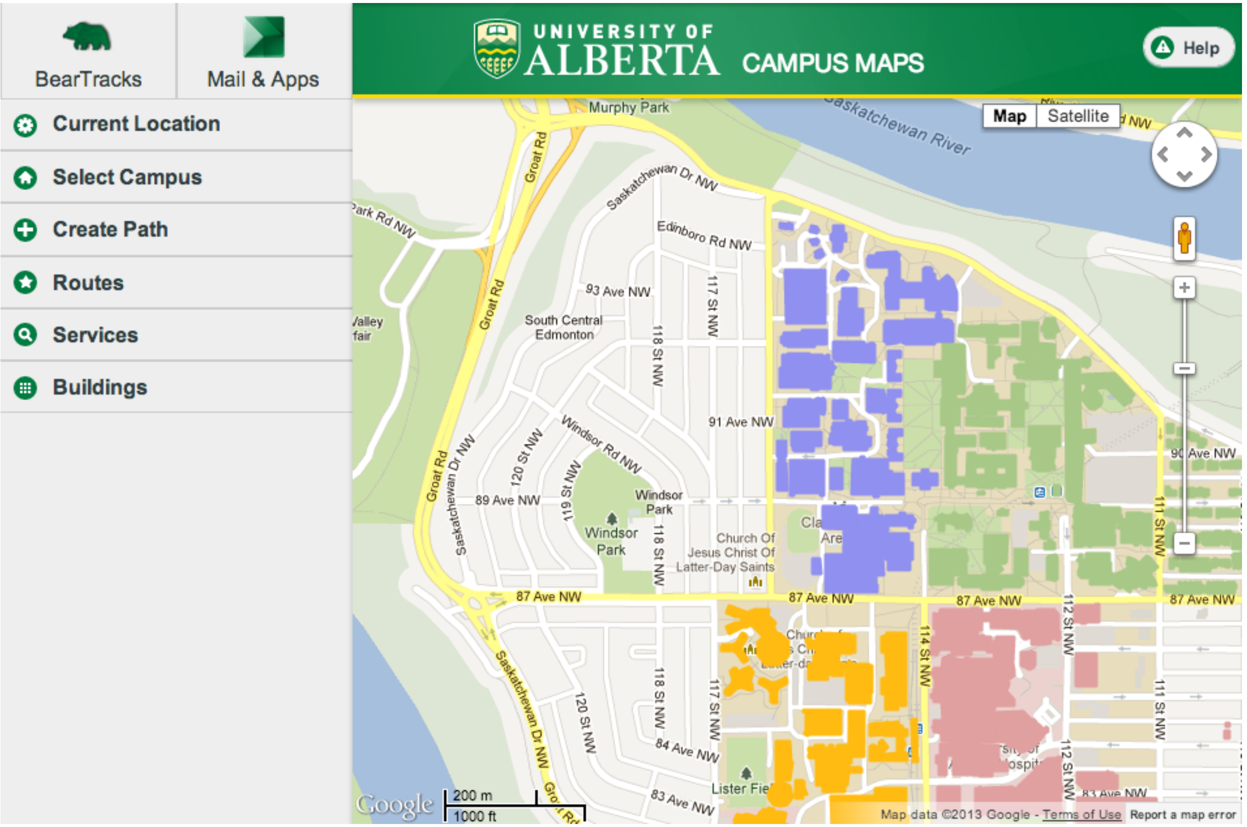
Product Overview

Summary

- The Campus Maps team is striving to provide a real time, user-friendly path finding experience across all of the University of Alberta campuses.
- Currently anyone who would like to find their way around using campus maps can do so, but the path provided does not give you directions directly to a door or specific service inside of a building.
- One will also be able to add waypoints to a specific path.
- Another example would be a disabled route, taking into account both stairs and elevators when planning the route

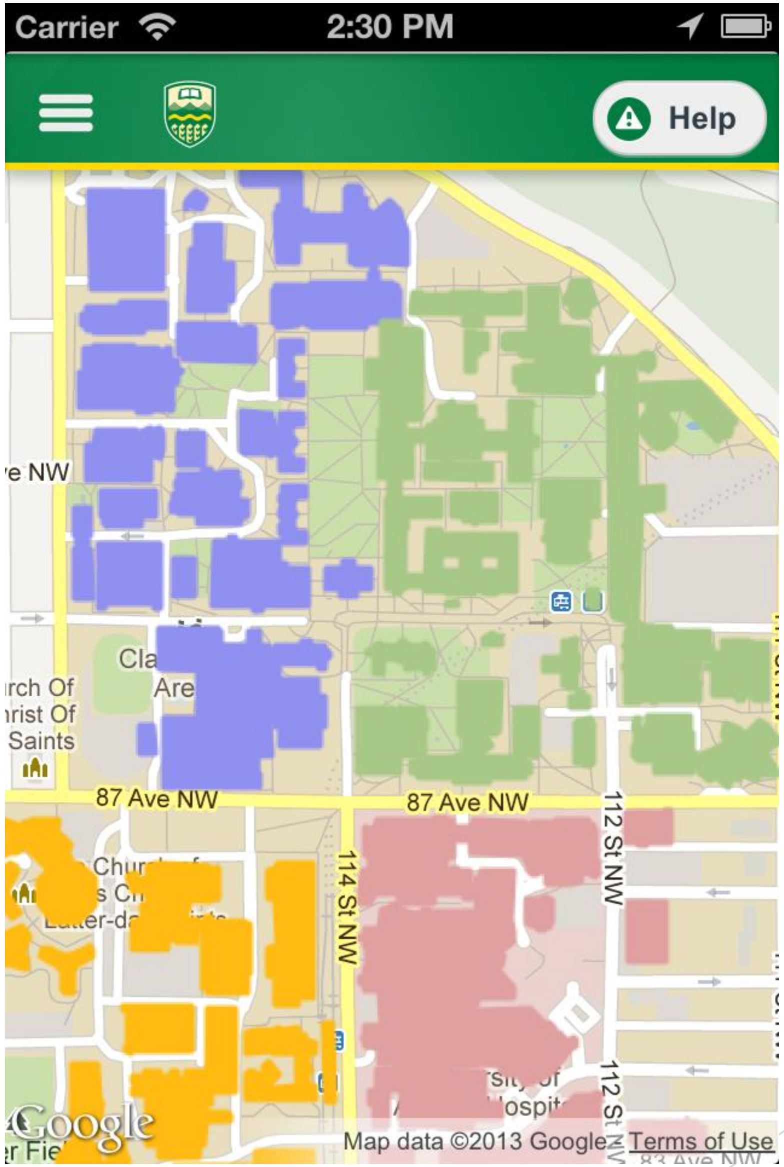
Desktop

We would like to see this service working on modern browsers, accessible by both HTML5 capable computers and mobile devices



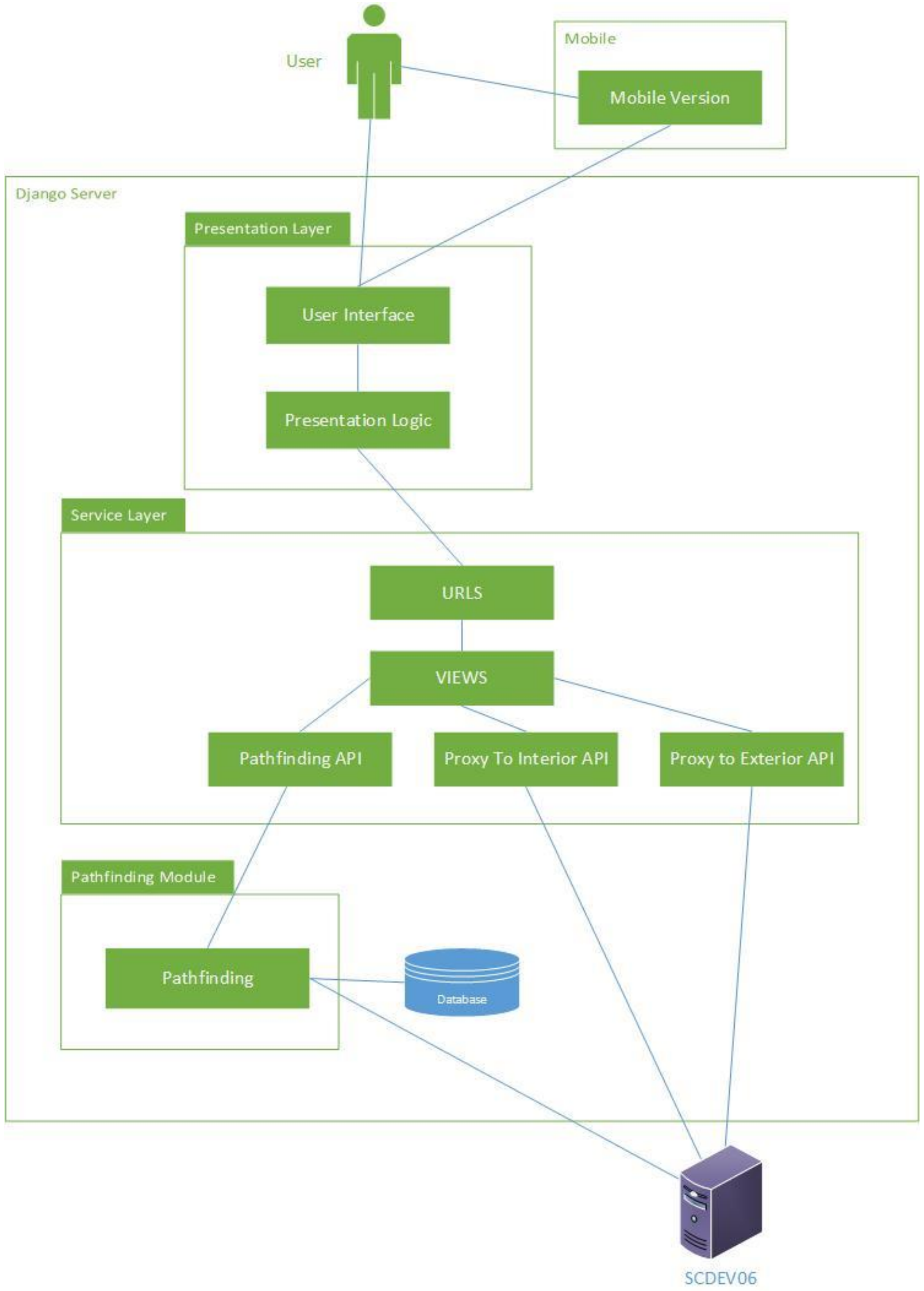
Mobile

Using phonegap the desktop version can be easily wrapped into native apps for both iOS and Android



Video and User Scenarios

System Architecture


User
Mobile
Mobile Version
Django Server
Presentation Layer
User Interface
Presentation Logic
Service Layer
URLS
VIEWS
Pathfinding API
Proxy To Interior API
Proxy to Exterior API
Pathfinding Module
Pathfinding
Database
SCDEV06

Implementation Issues

Frontend/Backend

IBM Worklight

- Initially attempted to use IBM Worklight to create an integrated HTML5/JS application.

- Pro: Recommended (strongly) by TA.

- Pro: Integrated solution.

IBM Worklight

- Con: First attempts at use of Eclipse IDE encountered bugs.
- Con: Really, just a fancy IDE wrapper and for rapid deployment of enterprise applications using web technology.
- Con: Most of IBM Worklight’s features are outside of the scope of our project, we really just need the Apache Cordova wrapper for mobile apps.
- Major bug: We had trouble deploying our Worklight code onto our development server.

Solution – Back to basics

- Our solution was to simply stop using this recommended technology, and start making a stand-alone HTML5/JS web application.
- This gave us the freedom to start implementing without the overhead of Worklight.
- We could still wrap our code in Apache Cordova, after implementation, and achieve our goals of cross-platform mobile apps. This made implementation simpler.

Conflicting Requirements

- We were given two sets of requirements.
- Our client, the University Digital Strategy, requested a simple indoor path finding addition to their current Campus Maps implementation.
- Eleni, however, wanted us to remake and improve the current Campus Maps as well.
- This caused thrashing with the initial development, as it was unclear which party we should be listening to.

Awkward requirements

- A major requirement from Eleni was to have natural language, turn by turn directions.

- A massive amount of work would be needed, after the pathfinding is complete, to create these turn by turn directions.

- This is a really tough task, and could be a project on its own.

- This requirement was cut due to time constraints.

Progress

- After meeting with Eleni at the end of February, we found the route cause of the conflicting requirements. Eleni had originally elicited requirements from the UDS manager, Jennifer, whom was not one of our clients.

- Therefore, we were able to just focus on Eleni’s requirements, and progress increased dramatically.

Implementation Issues

Pathfinding

API Limitations

- The client wanted interior to exterior pathfinding, but the API did not support exterior door locations.
- The client wanted paths between different floors, but stairwell coordinates did not line up.
- The client wanted paths to use elevators, but again, Elevators coordinates did not match.
- Furthermore, API docs contained syntax errors.

Overall pathfinding challenge

Challenge

- We have hallway drawings (or navigable area) but they are just CAD drawings, we cannot do path finding on them. We need to produce an A*-friendly navigable graph.
- We have room drawings with names but without doors.

Solution

- We ended up using Voronoi diagram since this potentially allows us to produce paths that go in the middle of and along the hallways.
- There is no door data returned from the API, so we decided to use the centre of rooms as room node to add to the navigable graph.

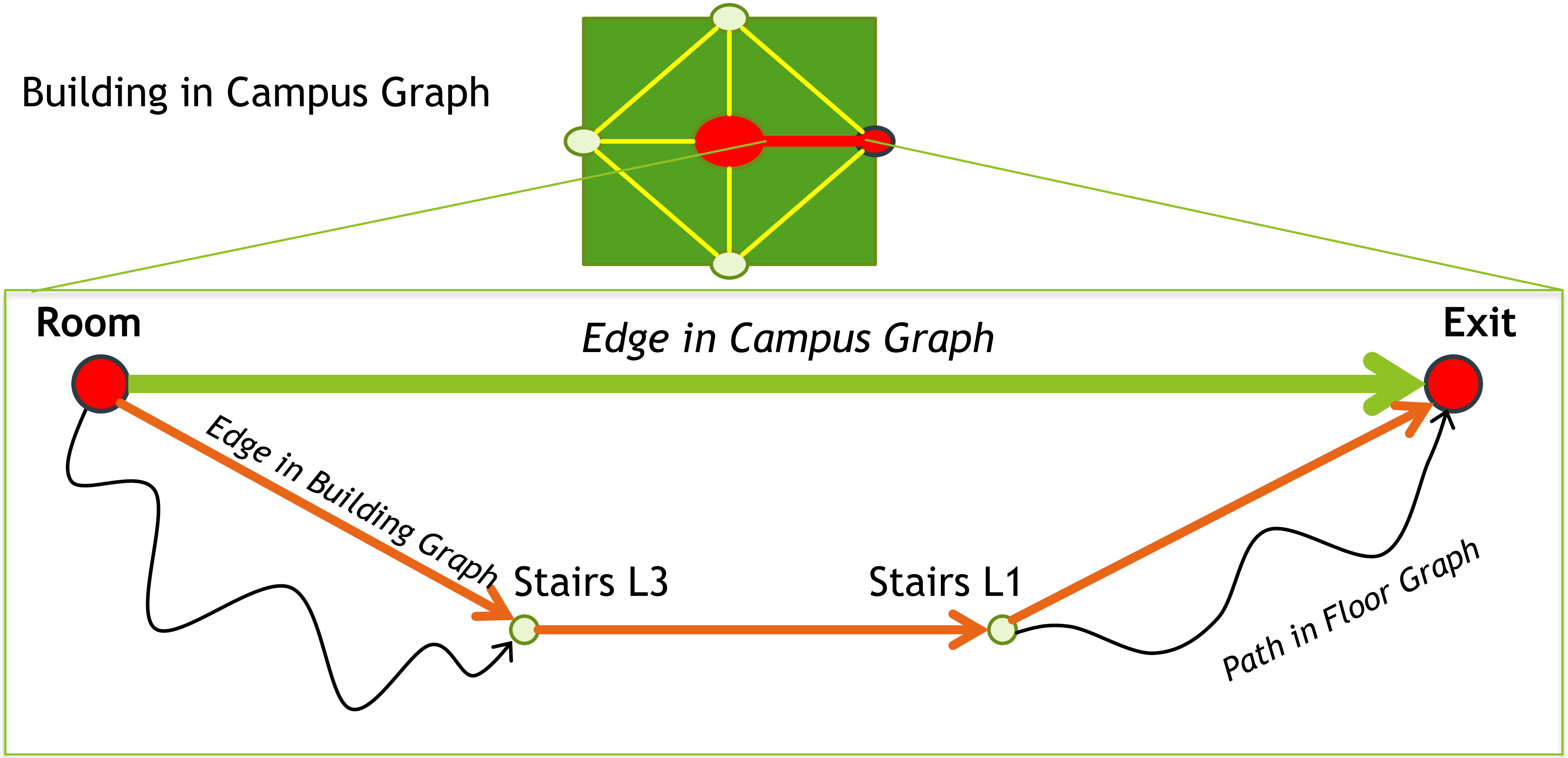
Path finding across floors of the same building

Challenge

- We must find a path using stairs and elevators, and have the ability to switch between the two.
- No stairs or elevator height data (distance weight), which is an issue in shortest path problem.
- Again, lack of sufficient data.

Solution

- We compute a building connection graph for each building.
- We asked the client to temporarily provide a reference id to determine the connection between portals.



Pathfinding is done in a hierarchal way over three abstraction layers:

- *Campus Graph* represents how doors on campus are connected
  - Edges represent paths in a building
- *Building Graph* represents how rooms, stairs, elevators, and entrances are connected
  - Edges represent paths over a floor or between floors
- *Floor Graph* represents how to navigate over a given floor in a building

Path finding across buildings on campus

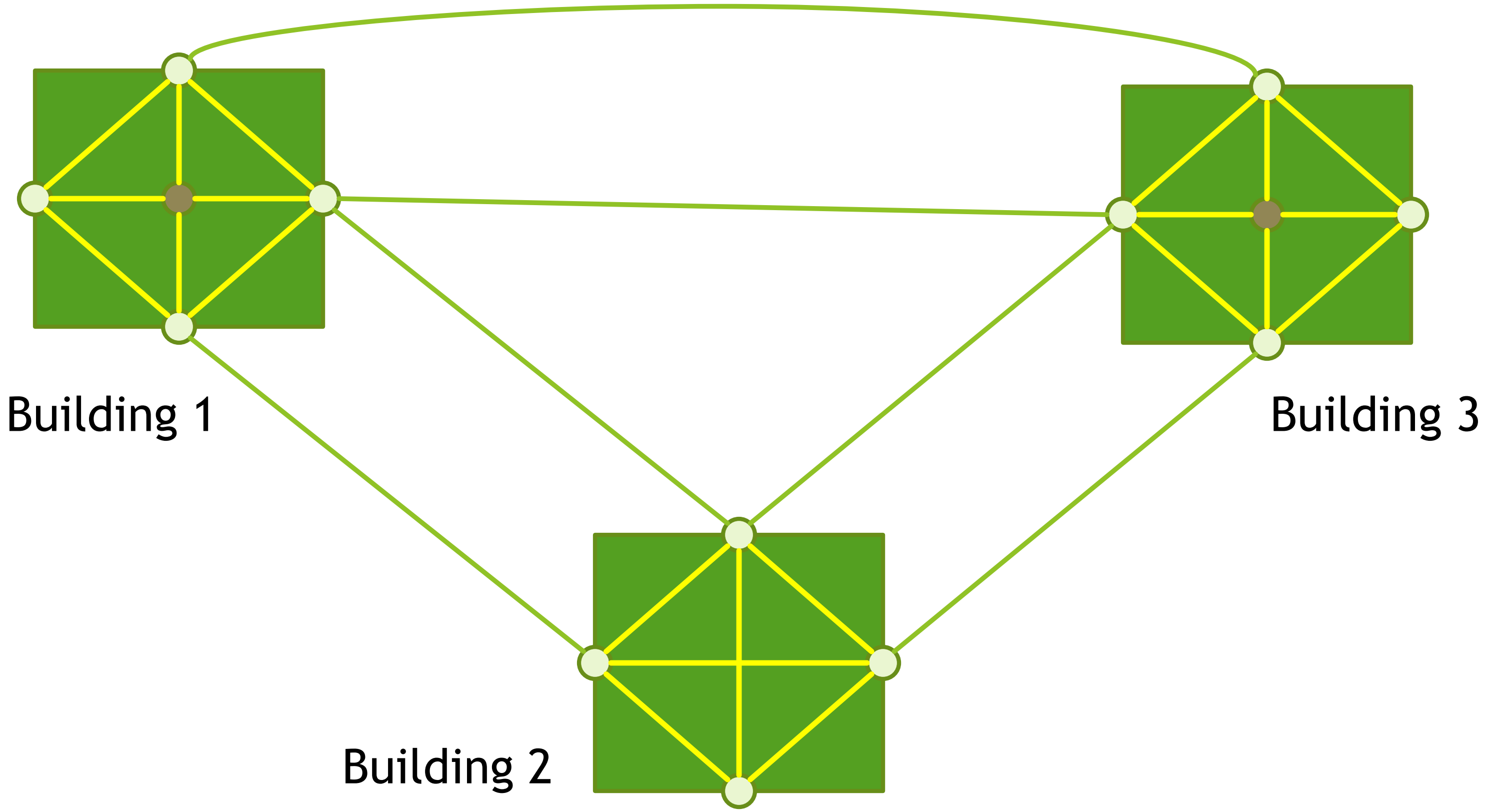
Challenge

- We must find the path between two rooms in two different building that consists of going up and down floors, getting in and out of buildings before reaching the destination room.
- Interior distance is not scaled to the exterior.
- Missing data.

Solution

- For each entrance of a building, we find paths to every entrance of other buildings.
- We find the shortest interior path between entrances of the same building using path finding between rooms in the same building algorithm.
- We manually mapped some entrances with lat/lng coordinates.

Sample campus connection graph



To be managed, or not to be managed?

Handling multiple clients

The plan

The team will be organized as a egoless, self organizing team. This allows for each member to work with collective ownership in mind while we are learning new languages and development methods. It also allows us to work by consensus so that the best ideas and solutions can be implemented while **guaranteeing** we meet our management requirements.

The goals

- Story points
- Weekly sprints
- Simple design
- Collective ownership
- Introduce any required redesigns and re-factorings constantly.
- Test first driven development
- Create and follow code reviews through GitHub
- Continuous integration using the server, non-stop build/testing
- Code commenting, simple, functional.
- No overtime
- Beer (team building)

Multiple clients

Conflicting requirements

Client

- We want an interior pathfinding algorithm
- Single path
- It has to use our data
- We don’t need a user interface
- This is only a prototype

CMPUT 401

- We need a visual interface to demo
- Path with waypoints
- Mobile applications
- Worklight
- This is a product
- Turn-by-turn natural language directions

Not to be managed

The result

- Conflicting requirements caused confusion and lack of organization
- Not having access to the client’s data stalled progress
- Nobody knew what to do
- Paperwork combined with the above scenario lead to thrashing
- Lack of leadership, and direction
- Needed help to get focused and gain access to required resources

Feb 20th

Code start

- Have API access!
- Set the server up and acquire data for pathfinding
- IBM Worklight
  - Used up valuable time attempting to implement
  - In reality it was the wrong tool for the problem
- Rotating product manager
  - New manager each sprint

March 1st

Houston we have a problem

- “I'm unsure what to do next and cannot imagine how we'll finish the project”
- “I feel I am not contributing as much as I should be”
- “I think some people lost a lot of steam”
- “I personally don't have any experience with anything we are doing”

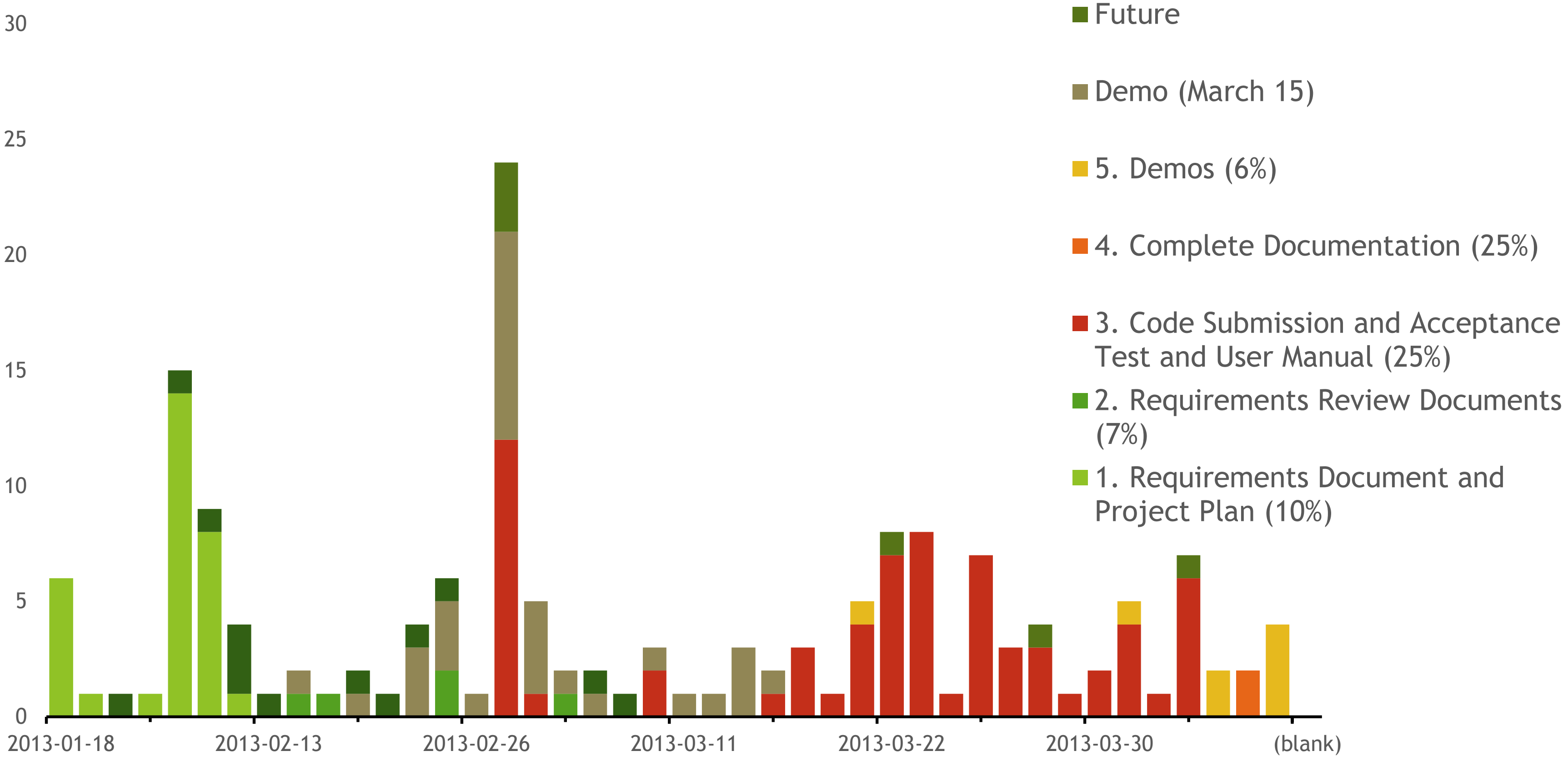
To be managed

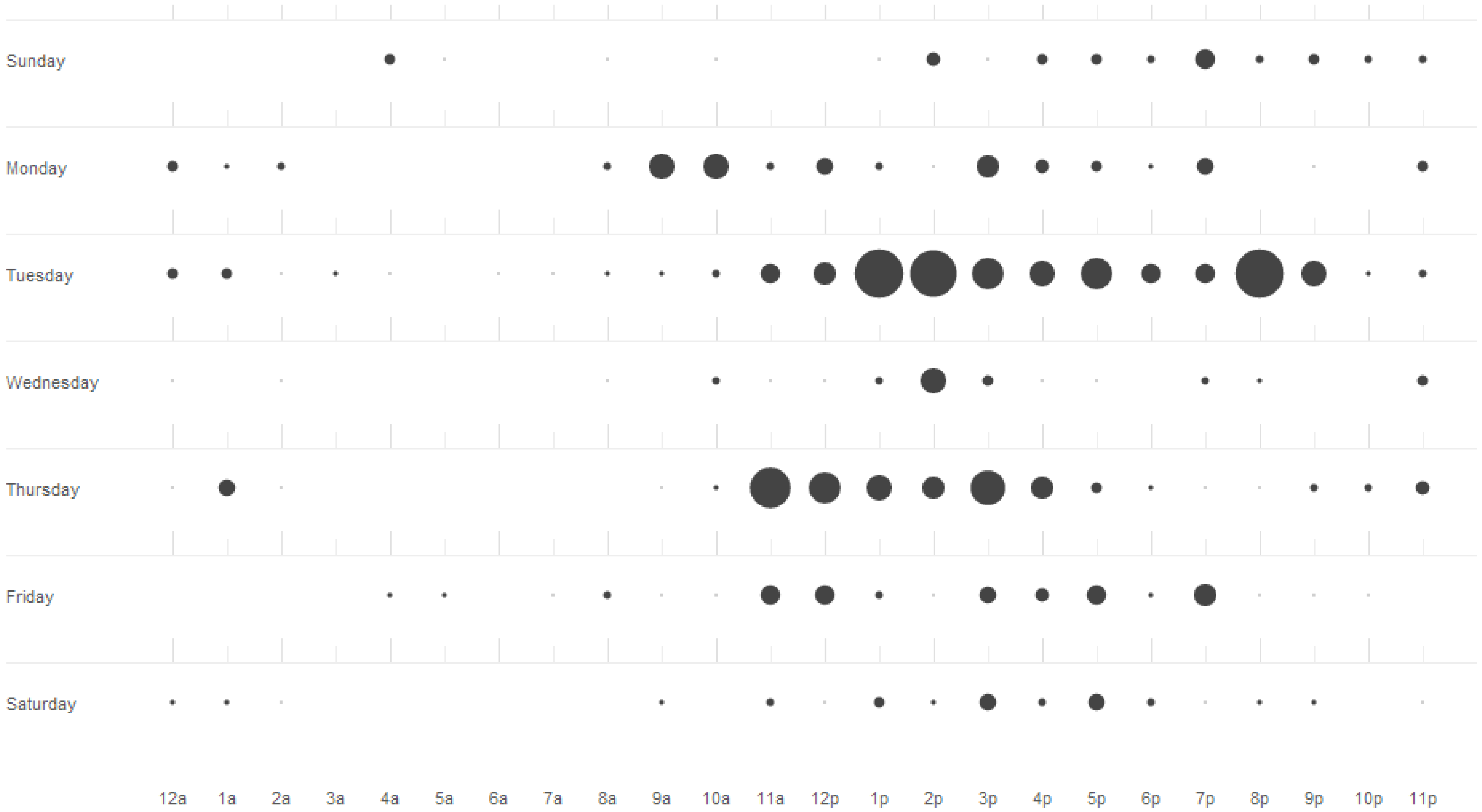
Actually starting, March 2nd to April 2nd

- Self organization isn’t working:
  - Split the team into three ‘focused’ teams
- We don’t have any issues or tasks:
  - Issues were created and assigned to individuals
- Dedicated manager:
  - Keep the project going, stay organized
- Coding session every week:
  - Worked most Tuesday's and Thursday’s

Issue creation

Getting started



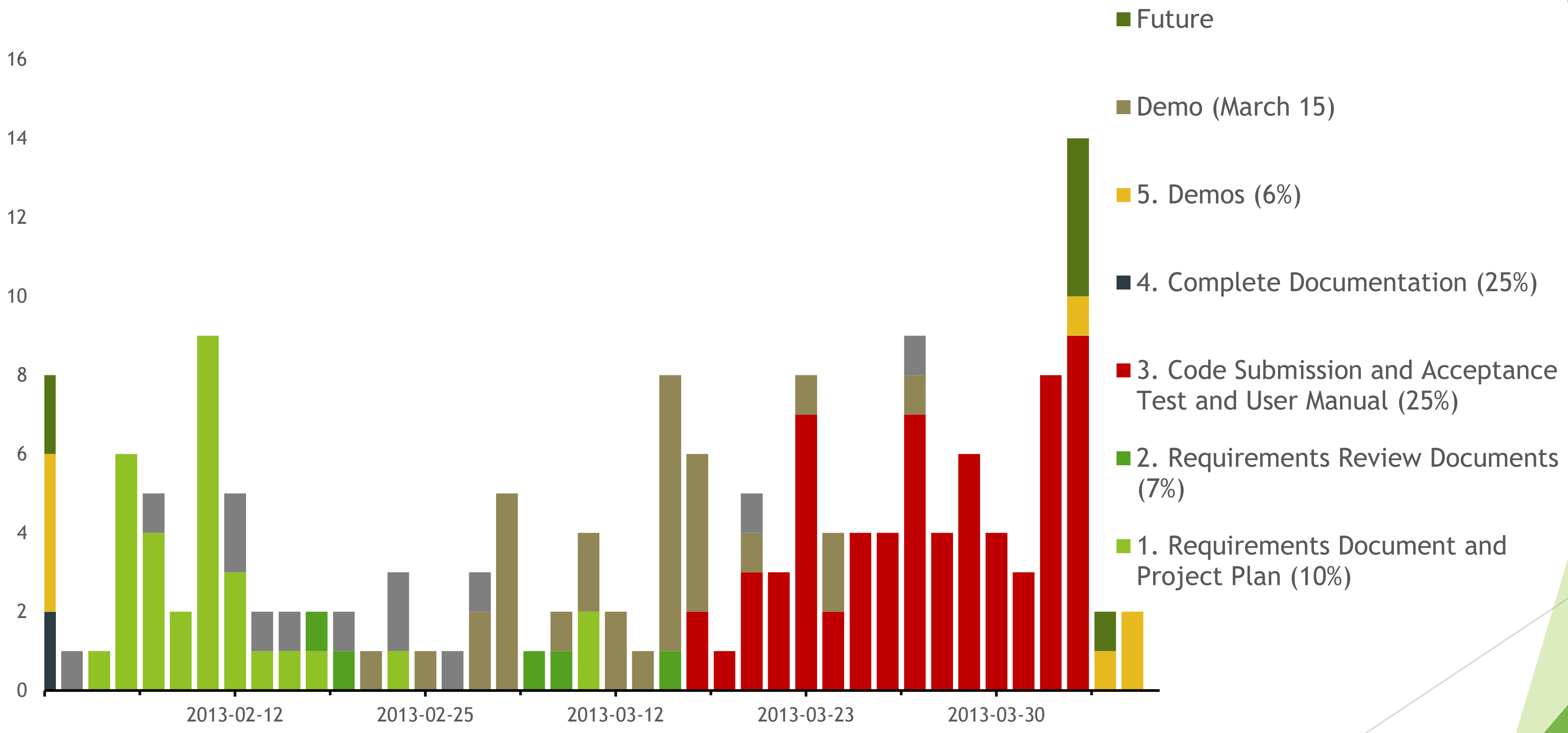


Tuesdays and Thursdays

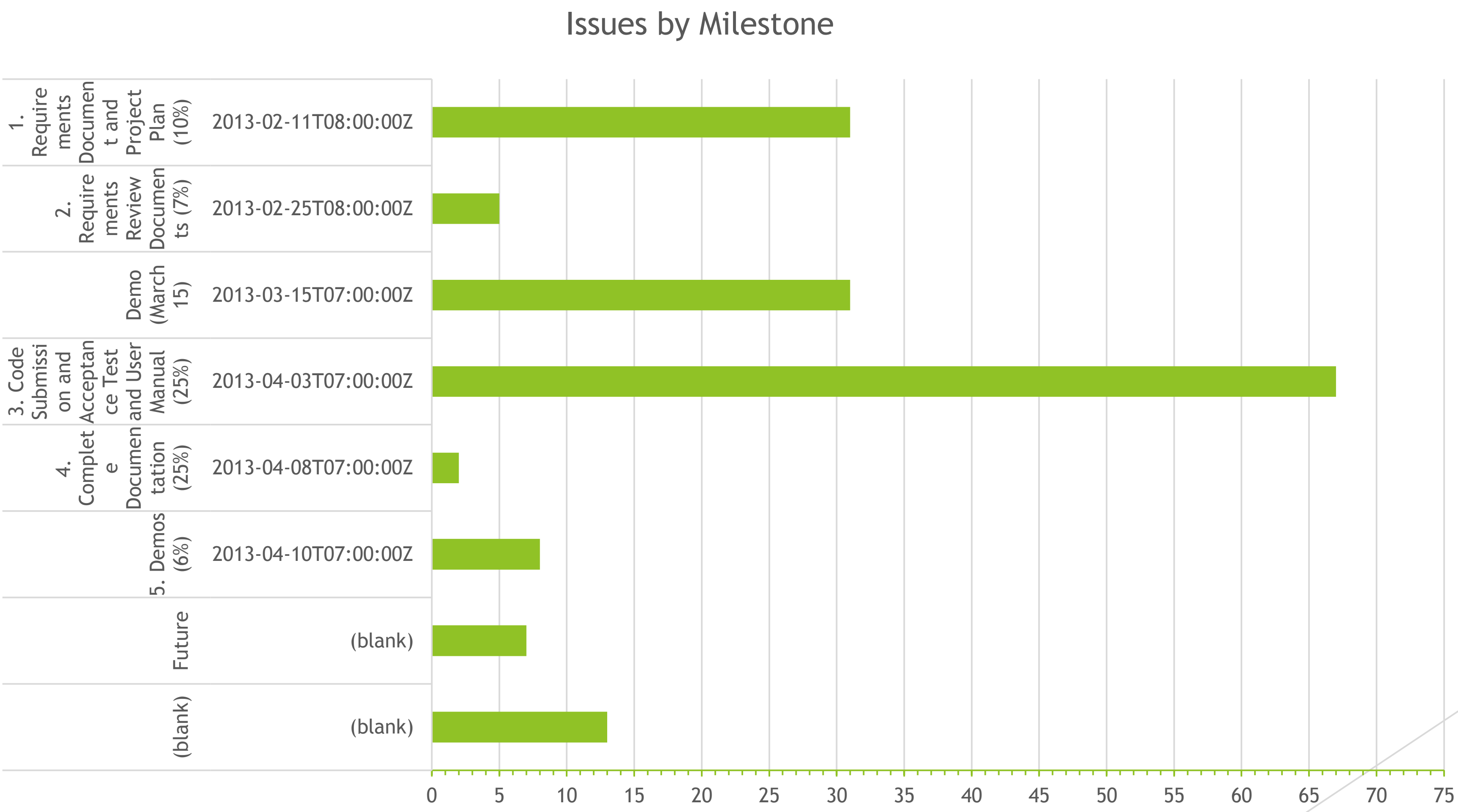
Teamwork, lots of hours to catch-up

Closing issues

A representation of progress



Sprints and Milestones



The goals (Recap)

- Story points
- Weekly sprints
- Simple design
- Collective ownership
- Re-factorings constantly.
- Test first driven development
- Create and follow code reviews through GitHub
- Continuous integration using the server, non-stop build/testing
- Code commenting, simple, functional.
- No overtime
- Beer (team building)

The goals we didn’t accomplished

- ~~Story points~~
- Weekly sprints – ~2 week sprints
- ~~Simple design~~
- ~~Collective ownership~~
- Re-factorings constantly – Where was the time?
- Test first driven development – Did not have time for the learning curve
- Create and follow code reviews through GitHub – We need it working now!
- ~~Continuous integration using the server, non-stop build/testing~~
- Code commenting, simple, functional. – Post-implementation commenting
- No overtime – Right.
- ~~Beer (team building)~~

Lessons Learned

The Bad…

- Python can be slow
  - Should perhaps have written some things in C++
- Do not assume client provided data to always be correct
- Python environment runtime issues
- more up-front design
  - have a design meeting
  - Documents
  - Diagrams
- think ahead a little more to the upcoming challenges
  - some things ended up surprising us and causing problems near the end
- Better planning using use case points or story points in a chart (gantt or burndown)

The Good…

- Learned how to use CSS3, HTML5, and JavaScript
- Learned how to wrap HTML5/JQuery code into a cross-platform app.
- Learned about responsive design.
- Learned about the importance of management firsthand
- Learned How to apply esoteric algorithms to solve real world problems
  - Voronoi diagrams for interior pathfinding
- Learned Python
  - using a mixture of OO and Procedural styles

The Awkward

- Its shorter to go down the stairs and take the elevator up than just going directly to the elevator.
- There’s no enforcement about booking the same breakout room for the entire day of the entire week.
- Tim Horton’s is not opened on Sunday.
- 50% student discount at Dominos Pizza.
- RATT is cheap for using plastic cups.

Future

Future Implementations

- Mostly indoor paths
- Mostly outdoor paths
- Using Lat/Long coordinates in outdoor paths
- Turn by turn natural language directions