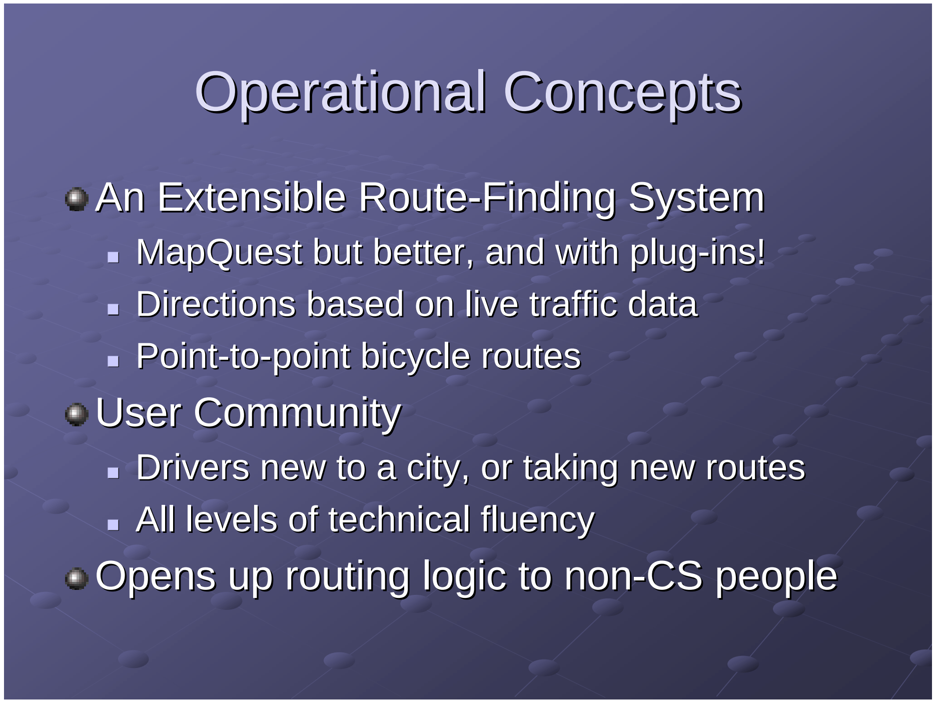
uRoute

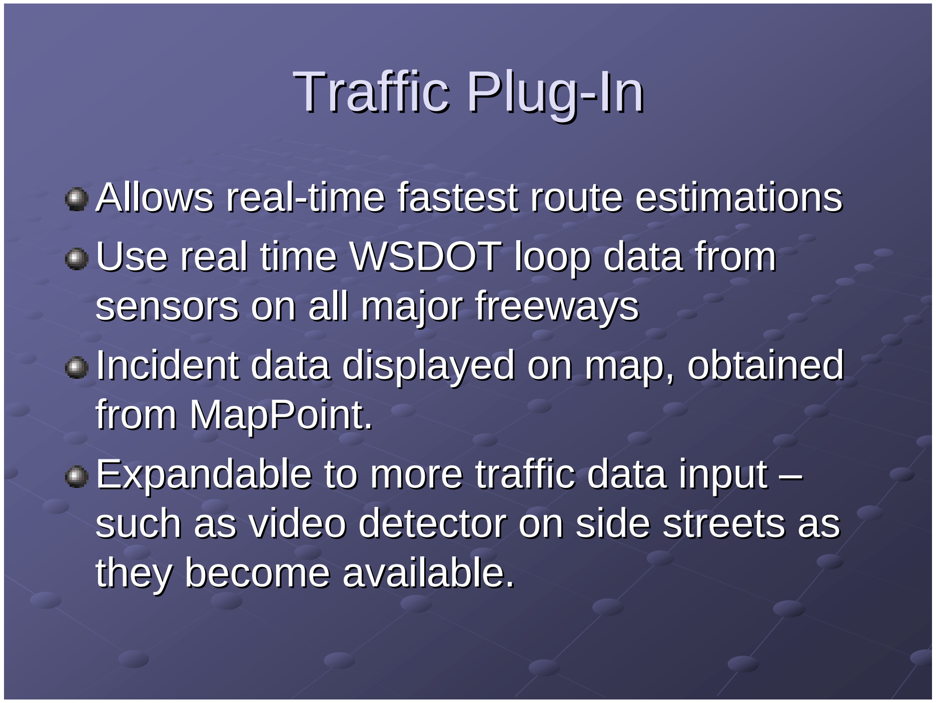
Map Team
LCA

Functional Specifications



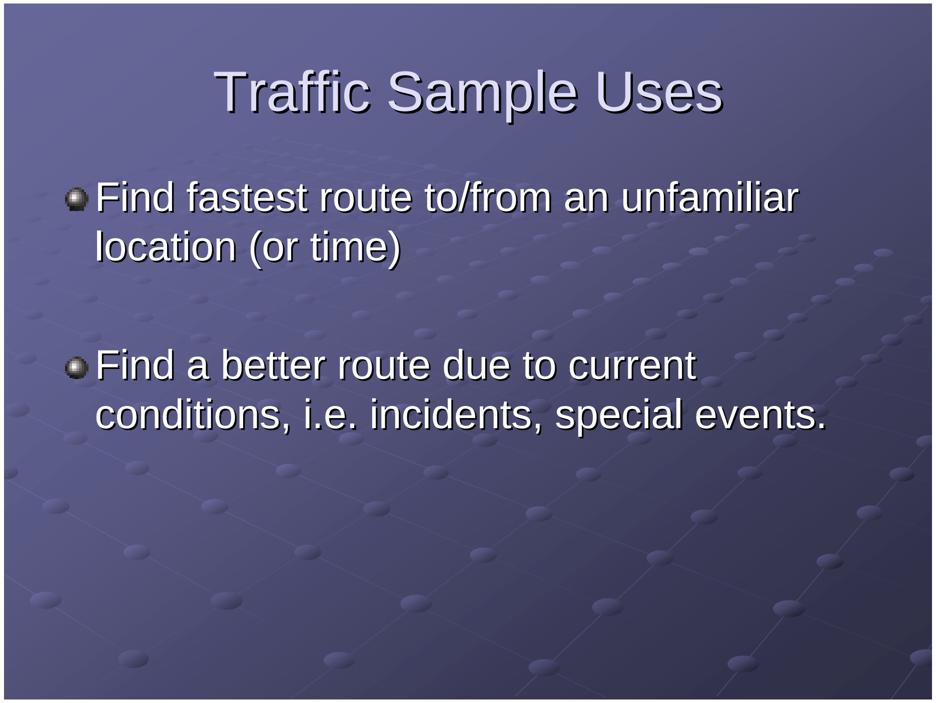
Operational Concepts

- An Extensible Route-Finding System
  - MapQuest but better, and with plug-ins!
  - Directions based on live traffic data
  - Point-to-point bicycle routes
- User Community
  - Drivers new to a city, or taking new routes
  - All levels of technical fluency
- Opens up routing logic to non-CS people



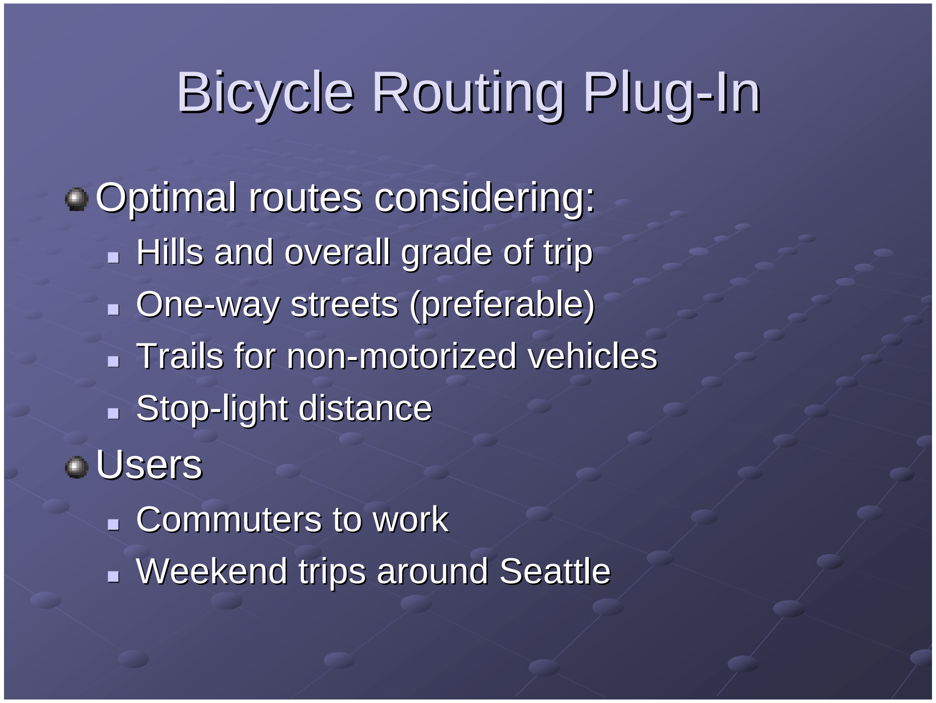
Traffic Plug-In

- Allows real-time fastest route estimations
- Use real time WSDOT loop data from sensors on all major freeways
- Incident data displayed on map, obtained from MapPoint.
- Expandable to more traffic data input – such as video detector on side streets as they become available.



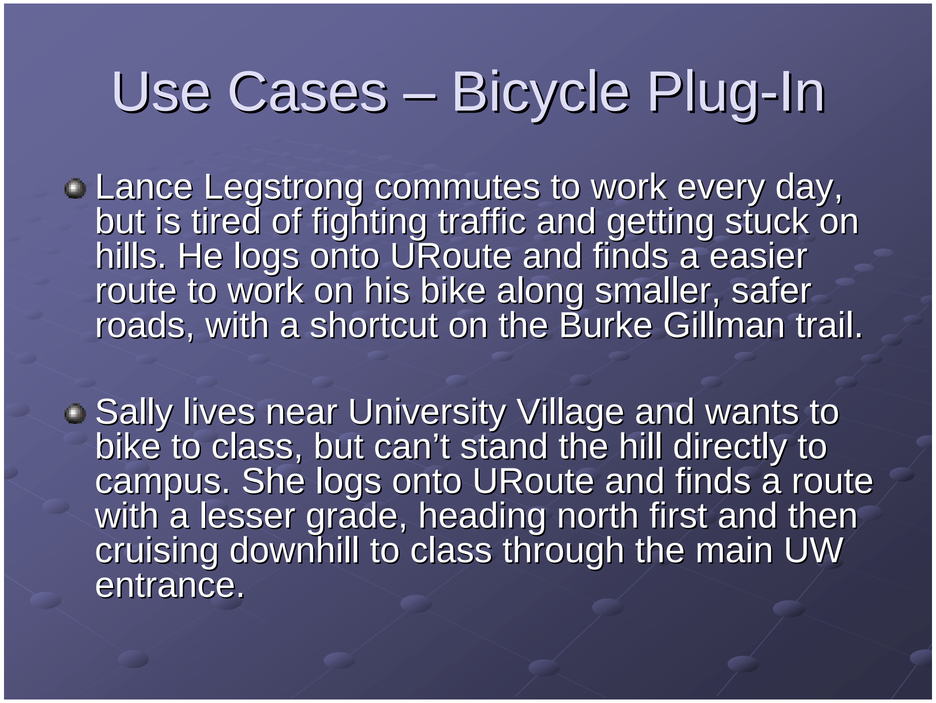
Traffic Sample Uses

- Find fastest route to/from an unfamiliar location (or time)
- Find a better route due to current conditions, i.e. incidents, special events.



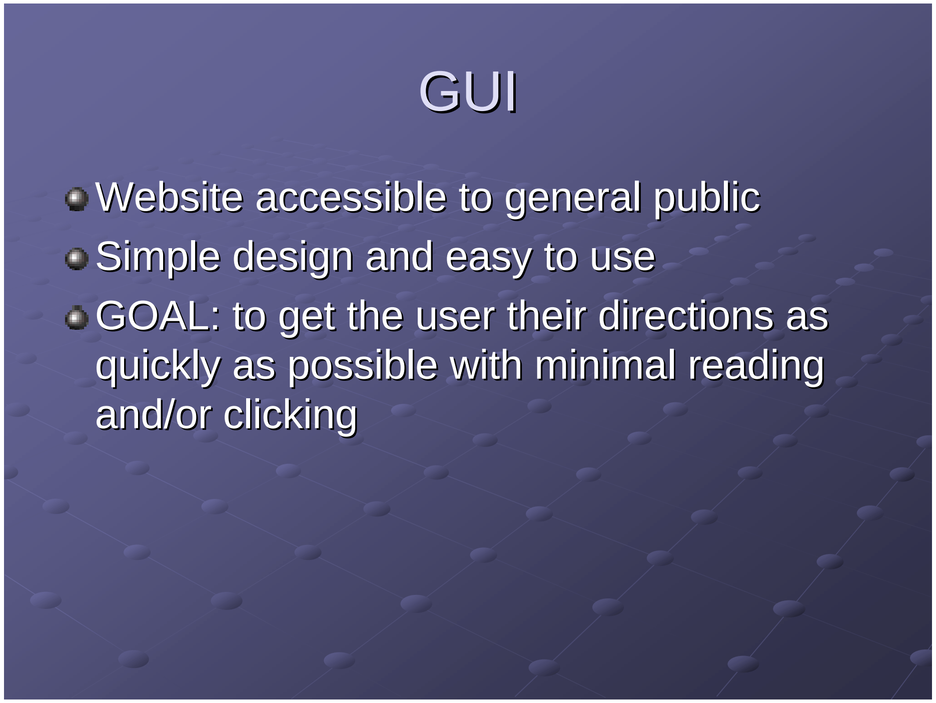
Bicycle Routing Plug-In

- Optimal routes considering:
  - Hills and overall grade of trip
  - One-way streets (preferable)
  - Trails for non-motorized vehicles
  - Stop-light distance
- Users
  - Commuters to work
  - Weekend trips around Seattle



Use Cases – Bicycle Plug-In

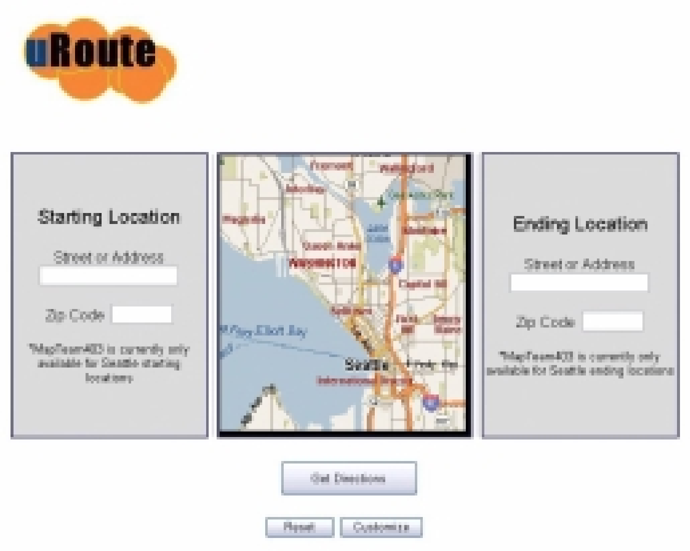
- Lance Legstrong commutes to work every day, but is tired of fighting traffic and getting stuck on hills. He logs onto URoute and finds a easier route to work on his bike along smaller, safer roads, with a shortcut on the Burke Gillman trail.
- Sally lives near University Village and wants to bike to class, but can't stand the hill directly to campus. She logs onto URoute and finds a route with a lesser grade, heading north first and then cruising downhill to class through the main UW entrance.



GUI

- Website accessible to general public
- Simple design and easy to use
- GOAL: to get the user their directions as quickly as possible with minimal reading and/or clicking

Home Page



Starting Location

Street or Address

Zip Code

*MapTeam403 is currently only available for Seattle starting locations

Ending Location

Street or Address

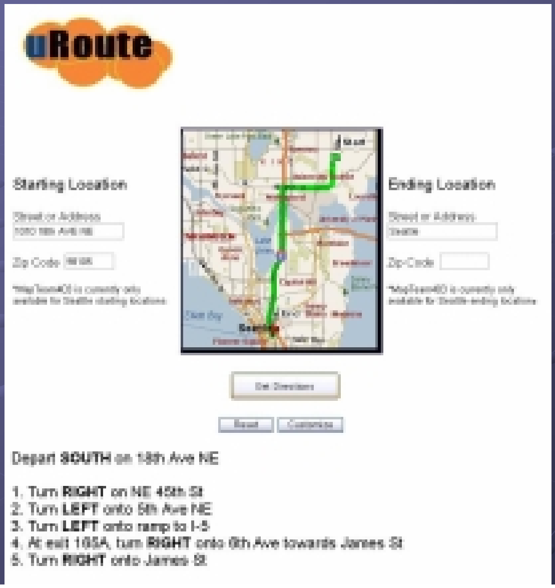
Zip Code

*MapTeam403 is currently only available for Seattle ending locations

GUI

- Two Ways to Input Locations
  - Type in Starting and/or Ending address
  - Click on Map to populate address fields
- Route is generated in real time
- Written Directions appear with one click
- User then has the option to go to a customization page to narrow their search

GUI: after “Get Directions”



Starting Location

Street or Address: 1050 18th AVE NE

Zip Code: 98105

*MapTeam403 is currently only available for Seattle starting locations

Ending Location

Street or Address: Seattle

Zip Code

*MapTeam403 is currently only available for Seattle ending locations

Depart **SOUTH** on 18th Ave NE

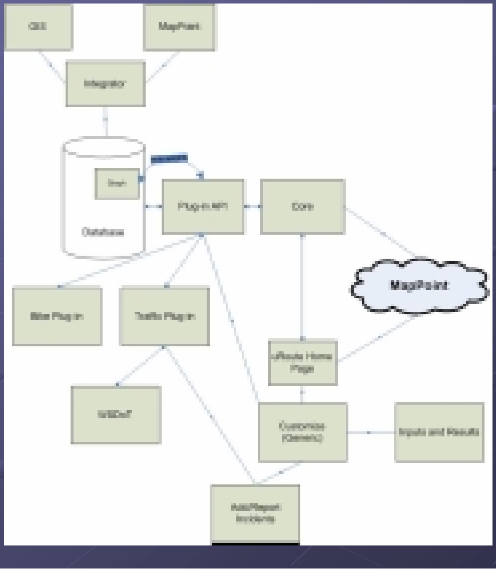
1. Turn **RIGHT** on NE 45th St
2. Turn **LEFT** onto 5th Ave NE
3. Turn **LEFT** onto ramp to I-5
4. At exit 165A, turn **RIGHT** onto 6th Ave towards James St
5. Turn **RIGHT** onto James St

Plug-In Engine

- Serves as interface between plug-ins and our system
- Plug-ins can both view and submit changes to the data model
  - Example: traffic plug-in submits current traffic data

Technical Specification and System Architecture

Flow Diagram



Core Routing Algorithm

- Using an implementation of A*.
- Heuristic state space search using f(n) = g(n) + h(n) to select the next state.
- Guaranteed to find the shortest path.
- Will run faster than Dijkstra's algorithm.

Plug-In Engine

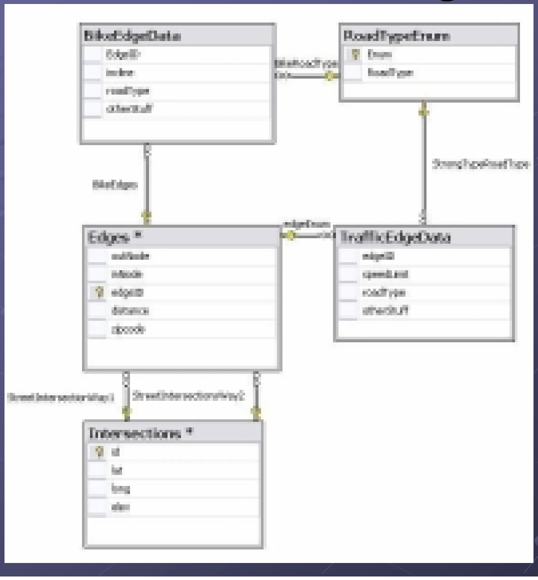
- Plugin API:
  - Plugins can access these primary methods:

```
XmlMapSection GetData(string regionName);
bool SubmitUpdate(string regionName,
                  XmlMapSection newData);
bool AddRegion(string regionName, string
                  parentRegion
```

  - Data between plugins and URoute formatted in XML

```
<node ID="??" location="???" region="???" moreStuff="???">
     <edge ID="???" endNode="endNodeID" weight="???" moreWeights="???" type="pluginType"/>
</node>
```

Database Design



Lifecycle Plan

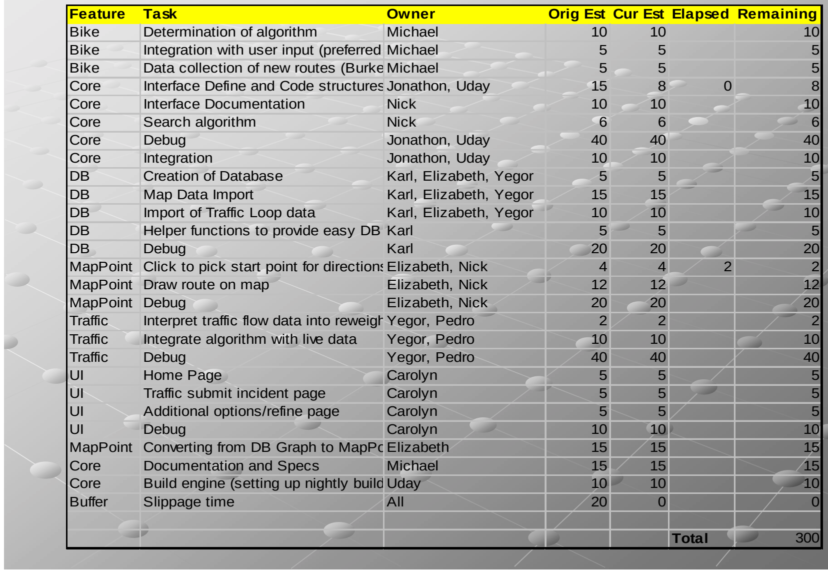
Objectives

- Spiral Development Model
  - Spiral 1 : Define interfaces and connections between major system components
  - Spiral 2 : Implement core with simple bike plug-in
  - Spiral 3 : Implement core with both plug-ins …and ship!

Rough Schedule

- **Week 1 (end Spiral 1)**: LCA Due Tuesday, present Wednesday. Finish specifications, server setup, database setup, and interface code by Sunday.
- **Week 2:** Coding of Search Algorithm, Map data import, Traffic data retrieval, basic UI done, bike plug-in.
- **Week 3 (end of Spiral 2):** Debug to Beta 1 from week 2
- **Week 4:** Integration of traffic Plug-in (with live data), full UI done with MapPoint
- **Week 5:** Beta 2 – Working as fully as possible
- **Week 6 (end of Spiral 3):** Debug to Final

Responsibilities



Feature	Task	Owner	Orig Est	Cur Est	Elapsed	Remaining
Bike	Determination of algorithm	Michael	10	10		10
Bike	Integration with user input (preferred	Michael	5	5		5
Bike	Data collection of new routes (Burke	Michael	5	5		5
Core	Interface Define and Code structures	Jonathon, Uday	15	8	0	8
Core	Interface Documentation	Nick	10	10		10
Core	Search algorithm	Nick	6	6		6
Core	Debug	Jonathon, Uday	40	40		40
Core	Integration	Jonathon, Uday	10	10		10
DB	Creation of Database	Karl, Elizabeth, Yegor	5	5		5
DB	Map Data Import	Karl, Elizabeth, Yegor	15	15		15
DB	Import of Traffic Loop data	Karl, Elizabeth, Yegor	10	10		10
DB	Helper functions to provide easy DB	Karl	5	5		5
DB	Debug	Karl	20	20		20
MapPoint	Click to pick start point for directions	Elizabeth, Nick	4	4	2	2
MapPoint	Draw route on map	Elizabeth, Nick	12	12		12
MapPoint	Debug	Elizabeth, Nick	20	20		20
Traffic	Interpret traffic flow data into reweigh	Yegor, Pedro	2	2		2
Traffic	Integrate algorithm with live data	Yegor, Pedro	10	10		10
Traffic	Debug	Yegor, Pedro	40	40		40
UI	Home Page	Carolyn	5	5		5
UI	Traffic submit incident page	Carolyn	5	5		5
UI	Additional options/refine page	Carolyn	5	5		5
UI	Debug	Carolyn	10	10		10
MapPoint	Converting from DB Graph to MapPo	Elizabeth	15	15		15
Core	Documentation and Specs	Michael	15	15		15
Core	Build engine (setting up nightly build	Uday	10	10		10
Buffer	Slippage time	All	20	0		0
					Total	300

Resources and Support

- Operations
  - Database Server
  - Webserver – IIS
  - Operations Staff post-launch
- Dependencies
  - GIS Data
  - Washington DOT data
  - MapPoint

Feasibility

- Pros:
  - Schedule of man hours is reasonable
  - Existing demand for driving directions
  - Extensibility increases chances of success
- Challenges:
  - Integration of different data sources
  - Public acceptance and our ability to attract developers
  - Actually finding a better route (around traffic, for bikes, etc.)

Future Extensions

- Display on mobile devices
- Integration with in-car GPS systems
- Expansion beyond greater Seattle area
- Route-finding for hikers in national parks
  - Plus route tracking via GPS to add to graph of park area