CSE 416, Section 1

Project Components

2

Team Activities

Be active with Piazza

- Be sure to read the project description page on the class web site (link in the Links box)
  - Become familiar with the references that describe the measures in detail
  - Review the project deliverable calendar and note the next two (use case list and GUI prototype) that are due
- Identify the technical skills that your team will need to develop
- Start on a client/server prototype if you have not done a J2EE system before

3

Session Objectives

- Understand data requirements
- Start to think about your GUI
- Understand options for accessing and processing currently available data
- Define the functions of the system

4

Data Requirements

- 3 states
- Boundary data (e.g., precinct, congressional district)
- Demographic data (e.g., population, voting age population, population by racial/ethnic group)
- Election result data (not for required use cases)

5

Preprocessing

- Goal – Do as much processing as possible in advance to lessen the server processing load
- Sample preprocessing tasks
  - Break out precinct boundary data if your data source groups it together
  - Determine precinct neighbors
  - Map some data identifiers to a canonical name (e.g., precinct name)
  - Combine multiple data sources (e.g., census) to generate complete precinct data
  - Write data to tables in your DB and to sequential files for use in the SeaWulf

Use a data source that has already done much of the preprocessing

6

Precinct Graph Formation

- Goal – form the graph of all precincts
- Graph
  - Each precinct is a node in the graph
  - Physically adjacent precincts identify edges in the graph
- There may be some issues with the precision of the geometry
  (self-intersecting edges, gaps, etc.) – you can relax some precision as long as you can generate a reasonable graph of the precincts

7

Precinct Adjacency Problem

- Complexity
  - Up to 25,000 precincts (polygons) in a state
  - Up to 50 edges (line segments) in a precinct boundary polygon
  - Up to 1.25M line segments (25,000 * 50)
  - Every pair of line segments can be compared to identify adjacency (up to 1.6T comparisons)

You will need to avoid n^2 comparisons by defining some limited set of search spaces

8

Precinct Adjacency Approach

- Determine a "search space" to avoid the n^2 edge comparisons
- Identify the polygons in the search space
- For a given precinct (i.e., polygon), iterate through the other polygons in the search space
- Compare polygons using a library function for polygon adjacency
- Use a library function to determine minimum line adjacency (200 feet)

Some Python libraries will allow you to define tree structure bounding areas for search

9

Data Combining

- Your precinct objects should contain
  - Precinct identifier / county identifier
  - Boundary data
  - Election results (if required for one of your use cases)
  - Demographic data (total population and voting age population)
- You might find multiple data sources with common precinct identifiers – combination will be easy
- You might need to get demographic data from US Census – combination will be more difficult

10

Election and Demographic Data Issues

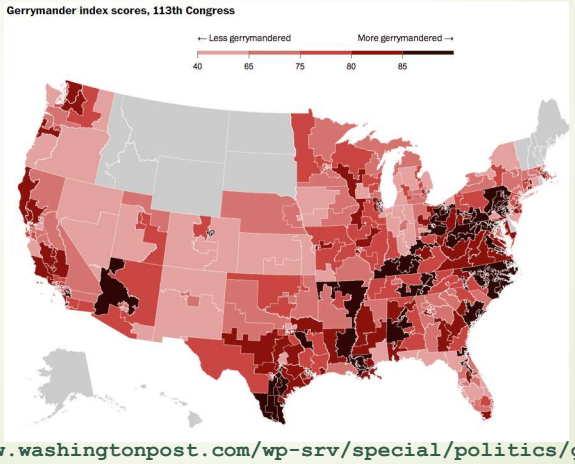
- Election results and demographic data originate from different sources (e.g., statewide tabulations and US Census Bureau)
- Census Bureau reports in various levels (blocks, groups, tracts, counties, and states), but possibly not precincts
- You need to identify a census block with a precinct, then accumulate demographic data into the precinct
- Average precinct is about 60 times larger than average census block
- Census Bureau attempts to coordinate with voting data through Voting Tabulation Districts (VTDs)

11

Web GUI

A prototype GUI is very helpful in driving out your requirements

Top-level page will likely be something like this



Be sure to use a mapping tool (e.g., Leaflet, Google Maps)

http://www.washingtonpost.com/wp-srv/special/politics/gerrymandering/

12

User Input

- Think of all the information a user might input, and design a part of your GUI
- Most likely you will
  - Use tabs for input/results
  - Plan on AJAX for dynamic GUI updates

13

OO Structure

- How do you identify the classes you need?
- How do you identify the attributes of each class?
- How do you determine the best type for an attribute?
- What is the best relationship among your classes

Remember to use encapsulation in your OO design

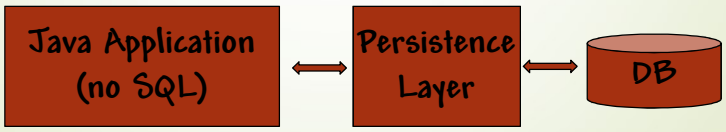
14

Persistence Layer

A major design issue is your DB, sequential files, and API access to existing data

- Your design will be object-first
- Database design will allow you to persist much of your OO data
- You will implement a layer (or use a tool) to
  - Retrieve DB data
  - Persist OO data

Requires OO-Relational mapping



JPA is helpful

15

GeoJSON

- Open standard format for representing simple geometric features
- Based on JSON
- Types – Point, LineString, Polygon, MultiPolygon
- Supported by Leaflet, Google Maps, et al
- Position information expressed as longitude, latitude


```
{
"type":"FeatureCollection",
"name":"precincts",
"description":"Minnesota Congressional District 1
"title":"Minnesota Congressional District 1 Votin
"publisher":"Office of the Minnesota Secretary of
"date":"July 1,2019",
"features":[
{"type":"Feature","properties":{"Precinct":"Amboy
Earth","CountyID":"7","CongDist":"1","MNSenDist":
[[[-94.1585,43.8916],[-94.1651,43.8915],[-94.1651
[-94.1657,43.8879],[-94.1665,43.8879],[-94.1665,4
[-94.1664,43.8868],[-94.1664,43.8862],[-94.1582,4
[-94.1583,43.8856],[-94.1585,43.8856],[-94.1585,4
[-94.159,43.8848],[-94.159,43.8849],[-94.1585,43.
[-94.1577,43.8861],[-94.1575,43.8861],[-94.1575,4
[-94.157,43.8842],[-94.157,43.8843],[-94.1574,43.
[-94.1537,43.8828],[-94.153,43.8829],[-94.153,43.
[-94.1529,43.8862],[-94.1529,43.8867],[-94.153,43
[-94.1485,43.8903],[-94.157,43.8902],[-94.157,43.
[-94.153,43.8887],[-94.153,43.8884],[-94.1536,43.
{"type":"Feature","properties":{"Precinct":"Beauf
Earth","CountyID":"7","CongDist":"1","MNSenDist":
[[[-93.8884,44.0222],[-93.9085,44.0221],[-93.9286
[-94.0084,43.964],[-94.0084,43.9349],[-93.9685,43
```


Be alert for MultiPolygon data

16

Shapefiles

- Geospatial vector data format
- Developed and maintained by ESRI
- Introduced in early 1990s
- Collection of files
  - Usually stored as a zip file
  - Mandatory files (.shp, .shx, and .dbf) and other files
- Represents points, lines, polygons
- Formatted as fixed length header, followed by one or more variable length records

Dominant format for geographic data due to the market dominance of ESRI

17

Precinct Boundary and Voting Data

- Possible sources
  - Harvard Election Data Archive – link in project page
  - OpenElections
  - States (e.g., https://www.sos.state.mn.us/election-administration-campaigns/data-maps)

Sources of Data

13. The MIT Election Data Scie
14. The Harvard Election Data
15. The Public Mapping Project
16. The Open Elections Project
17. A githb repository that might
18. Partisan Gerrymandering Hist
19. US Supreme Court Blog for G
    Contains links to many docum

Within a state, the office of the Secretary of State is usually responsible to provide election data

Keep checking the project page for new suggestions on data sources

18

Scope

Review the fall 2020 CSE416 use case list for style and scope

- Scope of the system is defined by your set of use cases
- A master set of use cases will be given to you following the requirements phase
- The list will include required use cases and optional use cases
- Use cases relating to standard system operation (e.g., change password) will not be in master use case list

Use cases are not a great fit for this project, but will be used as a way of normalizing units of work

There will likely be about 60-70 use cases, and you will have a target of 40 use cases

19

Comments on Project Use Case List

- Project not a great fit for use cases since actor driven scenarios create many complex use cases
- List of use cases is a 2-step process
  - Teams develop their list of requested use cases
  - Recommendations of each team will be considered in the generation of a master list of about 60-70 use cases (you will complete about 40 of these)
- Use cases the for project will
  - Provide balanced units of work
  - Allow for final demo grading based on completed use cases
- Use cases will consist of required, preferred, and optional
- Project grading will emphasize required and preferred use cases

20

How do You Assign Responsibilities to Team?

- Some parts of the project are standard SW development
- Major risk/unknown areas
  - What is the user interface? How do you display maps?
  - What data is associated with your requirements?
  - What are the best sources of that data?
  - How will you extract that data?
  - What analysis of the data is meaningful?
  - How do you test your system? To what do you compare your results?

Remember, the project is much more than just coding

21

High Priority Project Tasks

- Understand terminology and concepts in problem domain (read background references)
- Search for data and think about a starting OO structure that includes the graph components
- Think about the components in your GUI, along with Ajax updates
- Build a simple system prototype to help understand SW design issues (especially client/server interface)

22

Did You Achieve The Session Objectives?

- Understand data requirements
- Start to think about your GUI
- Understand options for accessing and processing currently available data
- Define the functions of the system