

Carnegie Mellon
DATA PRIVACY LAB

Identity Angel Project (2) How the Activities Fit Together

"Identity Angel uses filtered search as a way to identify sensitive resumes."



Latanya Sweeney

privacy.cs.cmu.edu

Project: Identity Angel

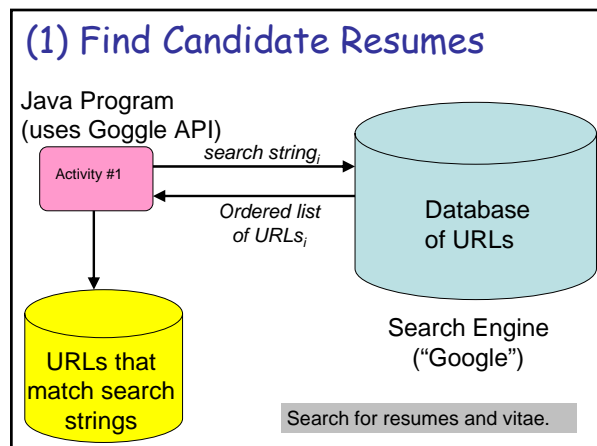
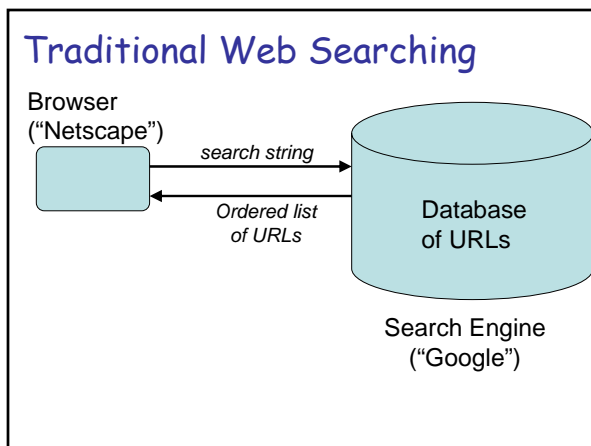
Imagine a benevolent program that locates resumes on the Web having Social Security Numbers, and then, for each resume found, sends an automated email message alerting the subject that they are needlessly placing themselves at risk to identity theft.

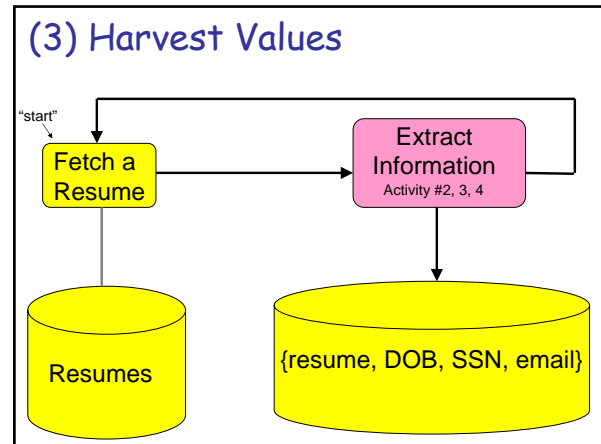
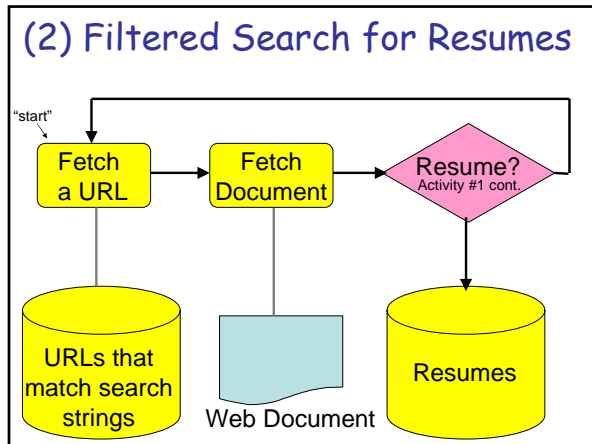
More on Assignment 1

- Identity Angel Architecture
- Coding and reporting specifics
- Who's doing what

Identity Angel: Major Steps

1. Conduct multiple searches.
Get candidate URLs from search engine.
2. Use a predicate function.
Determine whether retrieved documents are desired resumes.
3. Harvest sensitive values from resumes.
{SSN, DOB, email}





More on Assignment 1

Identity Angel Architecture

- Coding and reporting specifics
- Who's doing what

1. Identify "real" Resumes (A)

A. Your java code: operates from the command line [*main()*] accepting one argument *dirname*, which is the name of the subdirectory that will contain found resumes.

1. Identify "real" Resumes (B)

B. Make the subdirectory and execute your code.

Goal: have only resumes in *dirname*

Prefer resumes containing Social Security numbers.

1. Identify "real" Resumes (C)

C. Optional. Accept a second argument to *main()* named *rankfilename*.

Provide a text file named *rankfilename* in the same directory in which program operates.

The file contains the certainty your program gives each resume stored in *dirname*. Each line consists of: rank, filename where rank is in the range: 0 (not resume)... 1 (real resume with SSN)

1. Identify "real" Resumes (D)

D. Email your source (.java) files
and Javadoc for your code
to dp1staff@privacy.cs.cmu.edu

For more information on Javadoc, see
<http://lab.privacy.cs.cmu.edu/courses/java1/lectures/lecture14/sld036.htm>

2. 3. 4. Harvest Values (A)

A. Your java code: operates from the
command line [*main()*] accepting two
arguments *dirname* and *outfile*.

Resumes for your program to process can
be found in *dirname*.

You may assume the files are all text files
of somewhat appropriate content.

2. 3. 4. Harvest Values (B)

B. Provide a text file named *outfile* in the
same directory in which program operates.

Each line consists of:
filename, value₁, value₂, value₃

where filename is the name of the resume
file, value_i are extracted instances, up to 3.

2. 3. 4. Harvest Values (C)

D. Email your source (.java) files
and Javadoc for your code
to dp1staff@privacy.cs.cmu.edu

For more information on Javadoc, see
<http://lab.privacy.cs.cmu.edu/courses/java1/lectures/lecture14/sld036.htm>

5. 6. 7. Survey Related Areas (A)

A. Carefully select how you will conduct your
survey.

Gather information!
Conduct the survey!

Write a 3 page report summarizing your
method and findings.

5. 6. 7. Survey Related Areas (B)

B. Email your report and supporting materials
to dp1staff@privacy.cs.cmu.edu

8. 9. File Conversion (A)

A. Your java code: operates from the command line [*main()*] accepting two arguments *indir* and *outdir*.

Files for your program to convert can be found in *indir*, which is a subdirectory to where the program executes.

You may assume *indir* contains files of the appropriate type and content.

8. 9. File Conversion (B)

B. Make the subdirectory named *outdir* as a subdirectory to where the program executes.

Execute your code, placing converted files in *outdir*, bearing the same filename as the source file found in *indir*.

8. 9. File Conversion (C)

C. Optional. Accept a third argument to *main()* named *rankfilename*.

Provide a text file named *rankfilename* in the same directory in which program operates.

The file contains the certainty your program gives each converted file.

Each line consists of: rank, filename
where rank is in the range:

0 (not convert)... 1 (converted without error)

8. 9. File Conversion (D)

D. Email your source (.java) files and Javadoc for your code to dp1staff@privacy.cs.cmu.edu

For more information on Javadoc, see <http://lab.privacy.cs.cmu.edu/courses/java1/lectures/lecture14/sld036.htm>

10. Email Notification (A)

A. Your java code: operates from the command line [*main()*] accepting two arguments *mailinglist* and *message*.

Both arguments are the names of text files. Your program provides a "mail merge" function in which *message* contains a form and *mailinglist* contains the list of email addresses to mail.

10. Email Notification (B)

B. The first line of *mailinglist* consists of:

<sender>,<subject>,<info>,<signature>

<sender> is email address of message sender.

<subject> is the subject line of the message.

<info> is a URL for more information

<signature> will appear at the end of the message

Example:

ls@cmu.edu, Help, cmu.edu, LS

10. Email Notification (C)

C. The remaining lines of *mailinglist* consists of:

```
<email address>,<SSN>  
<email address> is email address of recipient.  
<SSN> is first few digits of the Social Security number.
```

10. Email Notification (D)

D. The file *message* contains the body of the email message. Any occurrences of:

```
<email address>,<SSN>  
<sender>,<subject>,<info>,<signature>
```

should be replaced in the email message with the value.

10. Email Notification (E)

E. Email your source (.java) files and Javadoc for your code to `dp1staff@privacy.cs.cmu.edu`

For more information on Javadoc, see <http://lab.privacy.cs.cmu.edu/courses/java1/lectures/lecture14/sld036.htm>

More on Assignment 1

- ✓ Identity Angel Architecture
- ✓ Coding and reporting specifics
 - Who's doing what