Basic WWW Technologies

Olli-Pekka Rinta-Koski

October 29, 2003

WWW - What Is It?

A combination of:

- **Resources**
  - Web pages (HTML), images (PNG, JPEG), other files (PDF, MP3, software, Flash animations, ...)
- **Resource identifiers**
  - Addresses of resources (URI, URL)
- **Transfer protocols**
  - Communication conventions between client and server (HTTP, SMTP)

Resource Identifiers

- **URI** – Uniform Resource Identifier

  `http://www.hut.fi/`

  `fax:+39-055-2222222`

  `sms:+393301234567`

  `tv:rai.it`

- **URL** – Uniform Resource Locator (also a URI!)

  `ftp://ftp.ietf.org/rfc/rfc2396.txt`

  - Scheme – `ftp:`
  - Hierarchical part:
    * Authority – `//ftp.ietf.org`
    * Absolute path – `/rfc/rfc2396.txt`

Web Documents

- **Hypertext:**
  - Documents may contain (one-way) links pointing to other documents
- **Markup:**
  - Documents may contain structural information and meta-information

SGML and HTML

- **SGML** is a **meta-language** used for defining other languages
- **HTML** is a language that can be specified in SGML (with a DTD)
- **Descriptive markup:**
  - describes content *(chapter, section, paragraph)*
- **Procedural markup:**
  - describes appearance *(bold, large, blinking)*

HTML

- Standardized by the WWW Consortium
- Specifies:
  - Character set
  - Markup syntax and semantics
- **Tags:**
  - HTML markup syntactically distinguishable from other content

```
<A HREF="tatti.html">
<IMG SRC="sieni.png" ALT="Tatti"> Tatti</A>
```

Valid HTML Document

- Latest definition: HTML 4.01 (1999) – `http://www.w3.org/TR/html401/`
- **Version information** = "This document contains markup as defined in ..."
- **Header** = "This document is of type ... and contains ... and was written by ... and other metadata"
- **Body** = actual content

Example: Simple HTML Document

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<HTML>
<HEAD><!-- HTML demonstration page -->
<TITLE>Sieni&auml;</TITLE>
</HEAD>
<BODY> <H1>Sieni&auml;</H1>
<P><A HREF="Amanita_virosa.html">
<IMG SRC="av.png" ALT="A. virosa">
Valkok&#xE4;rp&auml;ssieni</A>
<A HREF="Russula_paludosa.html">
<IMG SRC="rp.png" ALT="R. paludosa">Isohapero</A></P>
</BODY>
</HTML>
```

HTTP (HyperText Transfer Protocol)

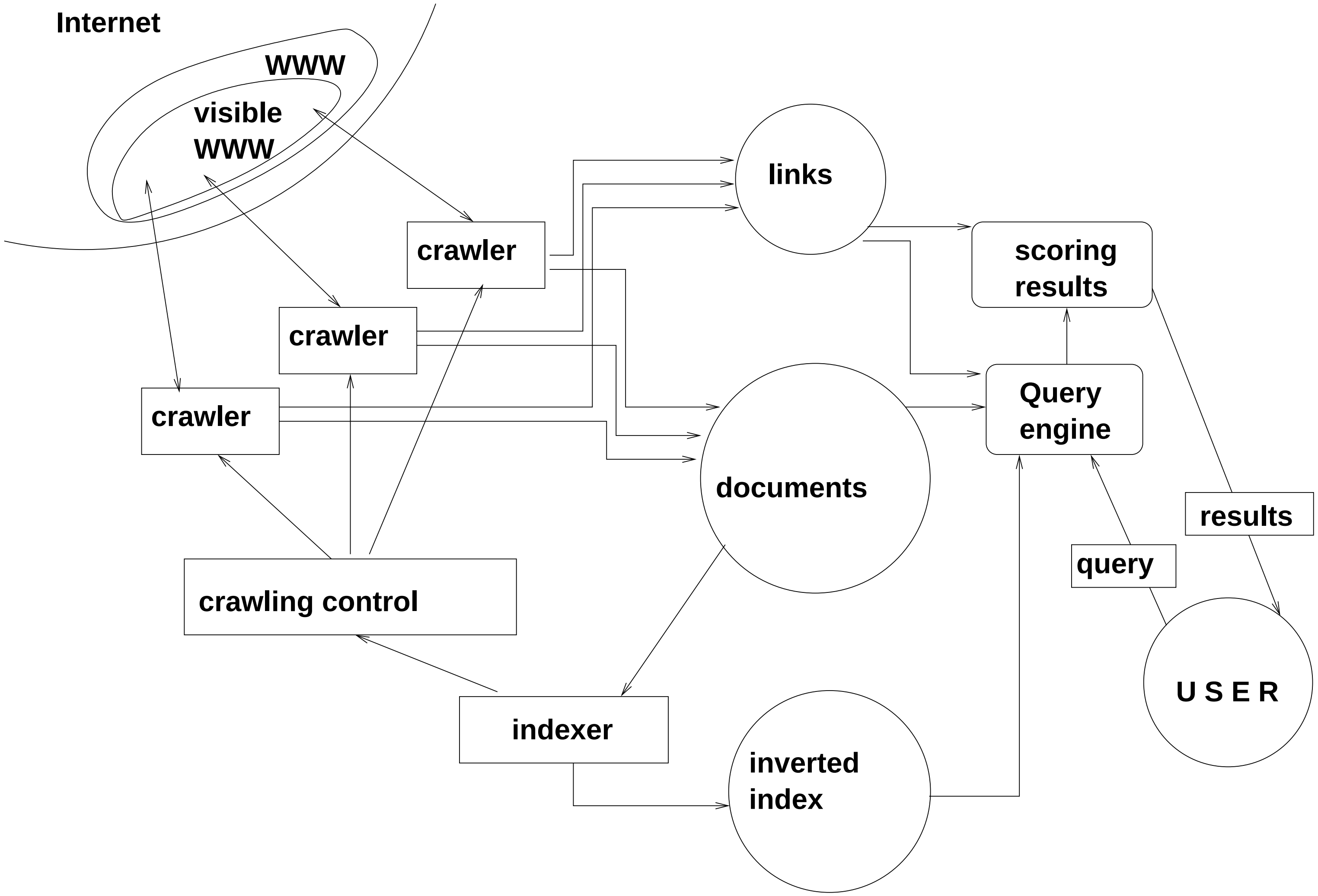
`ftp://ftp.isi.edu/in-notes/rfc2616.txt`

- Standardized set of query and response formats for client-server negotiation of (hypertext) documents
- HTTP Server = Web server
- HTTP Client = Web browser, crawler etc.
- HTTP/1.0 = 1 query/connection,
  HTTP/1.1 = multiple queries

Basic HTTP Commands

- **GET** - Retrieve entity identified by URI
- **HEAD** - Like GET, but for header only
- **POST** - Send enclosed entity to URI for handling
- **PUT** - Store enclosed entity under URI
- **DELETE** - Delete entity identified by URI

Search Engines



Search Engines (2)

- **Coverage** – the fraction of the visible Web searchable by the engine
- **Recency** *aka* freshness – the fraction of searchable pages that are up-to-date within the engine storage

Coverage

- **Overlap analysis**

  Estimating the size of the indexable Web:

  - W = set of Web pages
  - a, b = two independent search engines
  - $W_a \subset W$, $W_b \subset W$
  - $P(W_a), P(W_b)$ = probabilities that a page has been crawled by a/b
  - $P(W_a \cap W_b | W_b) = \frac{|W_a \cap W_b|}{|W_b|} = P(W_a)$
  - $\frac{|W_a|}{P(W_a)}$ = estimate for the size of the entire Web

Coverage (2)

- **November 1997:**
  $>$ 320 Mpages, 60% coverage by 6 major engines, maximum coverage for single engine: 1/3
- **February 1999:**
  $>$ 800 Mpages, 11 engines, 42% collective coverage, 16% maximum coverage
- **Current:**
  Google indexes 3 307 998 701 pages ($>$ 3300 Mpages) out of "several billion pages"

Basic Crawling

$Q \leftarrow S_0$	start with a set of seed URLs
while $Q \neq \emptyset$	while there are more URLs
do $u \leftarrow \mathrm{dequeue}(Q)$	get a URL from the queue
$\quad d(u) \leftarrow \mathrm{fetch}(u)$	fetch the corresponding document
$\quad \mathrm{store}(D, (d(u), u))$	store the document
$\quad L \leftarrow \mathrm{parse}(d(u))$	parse the URLs in the document
$\quad$ **for each** $v \in L$	for each of the URLs found
$\quad$ **do** $\mathrm{store}(E, (u, v))$	store the link
$\qquad$ **if** $\neg(v \in D \vee v \in Q)$	if target hasn't been stored or queued
$\qquad\quad$ **then** $\mathrm{enqueue}(Q, v)$	put it in the queue

Basic Crawling (2)

- **Practical considerations**
  - Storage resources are finite
  - The crawling process may be time-limited
  - A single crawling process may get stuck
  - Crawlers should respect `robots.txt` (see `http://www.robotstxt.org/`)
  - Crawlers should not overburden sites or network links
  - Finding out whether URL has been fetched or discovered may be expensive
  - The real world is more complex than a mathematical model (aliases, traps)

Homework Problems

1. Write a program that receives a HTML document as input and outputs the text (ie. non-tag content) of the document, followed by a list of outgoing hypertext links in the document. For reference, compare the output with that of
   ```
   lynx -dump <html document name>
   ```
2. Extend the program to retrieve the document from a Web server using HTTP 1.1.