

Introduction to Data Science

Lecture 2

Data Preparation

Data Science is about checking assumptions

English Premier League

Scores & Schedule

<

Week 1

Week 2

Week 3

Week 4

Week 5

Week 6

>

Saturday, August 30

Burnley FC	0	Final	Man City	0	Final
Man United	0		Stoke City	1	
QPR	1	Final	Newcastle	3	Final
Sunderland	0		Crystal Palace	3	
Swansea City	3	Final	West Ham	1	Final
West Brom	0		Southampton	3	
Everton	3	Final			
Chelsea	6				

Sunday, August 31

Aston Villa	2	Final	Tottenham	0	Final
Hull City	1		Liverpool	3	
Leicester City	1	Final			
Arsenal	1				

All times are in Pacific Time

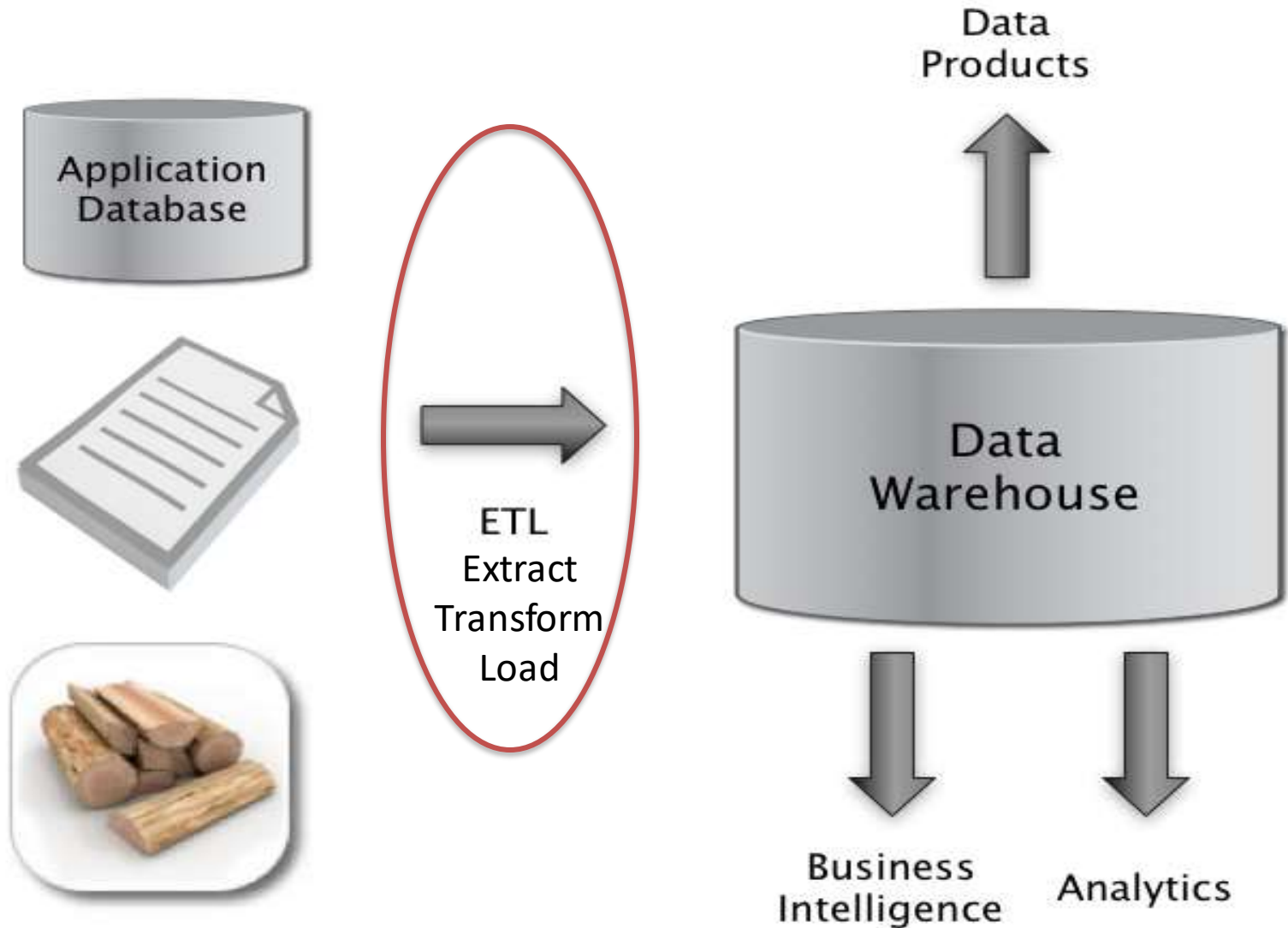
English Premier League

Scores & Schedule

< Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7 >
Saturday, September 13						
Arsenal		4:45 AM	Sunderland		7:00 AM	
Man City			Tottenham			
Stoke City		7:00 AM	Crystal Palace		7:00 AM	
Leicester City			Burnley FC			
Chelsea		7:00 AM	West Brom		7:00 AM	
Swansea City			Everton			
Southampton		7:00 AM	Liverpool		9:30 AM	
Newcastle			Aston Villa			
Sunday, September 14						
Man United		8:00 AM				
QPR						
Monday, September 15						
Hull City		12:00 PM				
West Ham						

All times are in Pacific Time

The Big Picture



Data Preparation overview

- ETL
 - We need to **extract** data from the **source(s)**
 - We need to **load** data into the **sink**
 - We need to **transform** data at the source, sink, or in a **staging area**
 - Sources: file, database, event log, web site, HDFS...
 - Sinks: Python, R, SQLite, RDBMS, NoSQL store, files, HDFS...

Data Preparation overview

- Process model
 - The construction of a new data preparation process is done in many phases
 - Data **characterization**
 - Data **cleaning**
 - Data **integration**
 - We must efficiently move data around in space and time
 - Data **transfer**
 - Data **serialization** and **deserialization** (for files or network)

Data Preparation overview

- Workflow
 - The transformation **pipeline** or **workflow** often consists of many steps
 - For example: Unix pipes and filters
 - `$ cat data_science.txt | wc | mail -s "word count" myname@some.com`
 - If the workflow is to be used more than once, it can be **scheduled**
 - Scheduling can be time-based or event-based
 - Use publish-subscribe to register interest (e.g. Twitter feeds)
 - Recording the execution of a workflow is known as capturing **lineage** or **provenance**

The Businessperson

- Data Sources
 - Web pages
 - Excel
- ETL
 - Copy and paste
- Data Warehouse
 - Excel
- Business Intelligence and Analytics
 - Excel functions
 - Excel charts
 - Visual Basic?!

The Programmer

- Data Sources
 - Web scraping, web services API
 - Excel spreadsheet exported as CSV
 - Database queries
- ETL
 - wget, curl, BeautifulSoup, lxml
- Data Warehouse
 - Flat files
- Business Intelligence and Analytics
 - Numpy, Matplotlib, R, Matlab

The Enterprise

- Data Sources
 - Application databases
 - Intranet files
 - Application server log files
- ETL
 - Informatica, IBM DataStage, Ab Initio, Talend
- Data Warehouse
 - Teradata, Oracle, IBM DB2, Microsoft SQL Server
- Business Intelligence and Analytics
 - Business Objects, Cognos, Microstrategy
 - SAS, SPSS, R

The Web Company

- Data Sources
 - Application databases
 - Logs from the services tier
 - Web crawl data
- ETL
 - Flume, Sqoop, Pig, Crunch, Oozie
- Data Warehouse
 - Hadoop/Hive, Spark/Shark
- Business Intelligence and Analytics
 - Custom dashboards: Argus, BirdBrain
 - R

Data Sources at Web Companies

- Examples from Facebook
 - Application databases
 - Web server logs
 - Event logs
 - API server logs
 - Ad server logs
 - Search server logs
 - Advertisement landing page content
 - Wikipedia
 - Images and video

Tabular Data

- What is a table?
 - A **table** is a collection of **rows** and **columns**
 - Each row has an **index**
 - Each column has a **name**
 - A **cell** is specified by an (index, name) pair
 - A cell may or may not have a **value**

Tabular Data

- Fortune 500

	A	B	C	D	E	F	G	H	I
1	rank	company	cik	ticker	sic	state_location	state_of_incorporation	revenues	profits
2	1	Wal-Mart Stores	104169	WMT	5331	AR	DE	421849	16389
3	2	Exxon Mobil	34088	XOM	2911	TX	NJ	354674	30460
4	3	Chevron	93410	CVX	2911	CA	DE	196337	19024
5	4	ConocoPhillips	1163165	COP	2911	TX	DE	184966	11358
6	5	Fannie Mae	310522	FNM	6111	DC	DC	153825	-14014
7	6	General Electric	40545	GE	3600	CT	NY	151628	11644
8	7	Berkshire Hathaway	1067983	BRKA	6331	NE	DE	136185	12967
9	8	General Motors	1467858	GM	3711	MI	MI	135592	6172
10	9	Bank of America Corp.	70858	BAC	6021	NC	DE	134194	-2238
11	10	Ford Motor	37996	F	3711	MI	DE	128954	6561
12	11	Hewlett-Packard	47217	HPQ	3570	CA	DE	126033	8761
13	12	AT&T	732717	T	4813	TX	DE	124629	19864
14	13	J.P. Morgan Chase & Co.	19617	JPM	6021	NY	DE	115475	17370
15	14	Citigroup	831001	C	6021	NY	DE	111055	10602
16	15	McKesson	927653	MCK	5122	CA	DE	108702	1263
17	16	Verizon Communications	732712	VZ	4813	NY	DE	106565	2549
18	17	American International Group	5272	AIG	6331	NY	DE	104417	7786
19	18	International Business Machines	51143	IBM	3570	NY	NY	99870	14833
20	19	Cardinal Health	721371	CAH	5122	OH	OH	98601.9	642.2
21	20	Freddie Mac	37785	FMC	2800	PA	DE	98368	-14025

Tabular Data

- Fortune 500

Fortune 500 with ticker and EDGAR ☆

File Edit View Insert Format Data Tools Help Last edit wa

Share...

New ▶

Open... %O

Rename...

Make a copy...

Import...

See revision history

Spreadsheet settings...

Download as ▶

Publish to the Web...

Email collaborators...

Email as attachment...

Print %P

CSV (current sheet)

HTML (current sheet)

Text (current sheet)

Excel

OpenOffice

PDF...

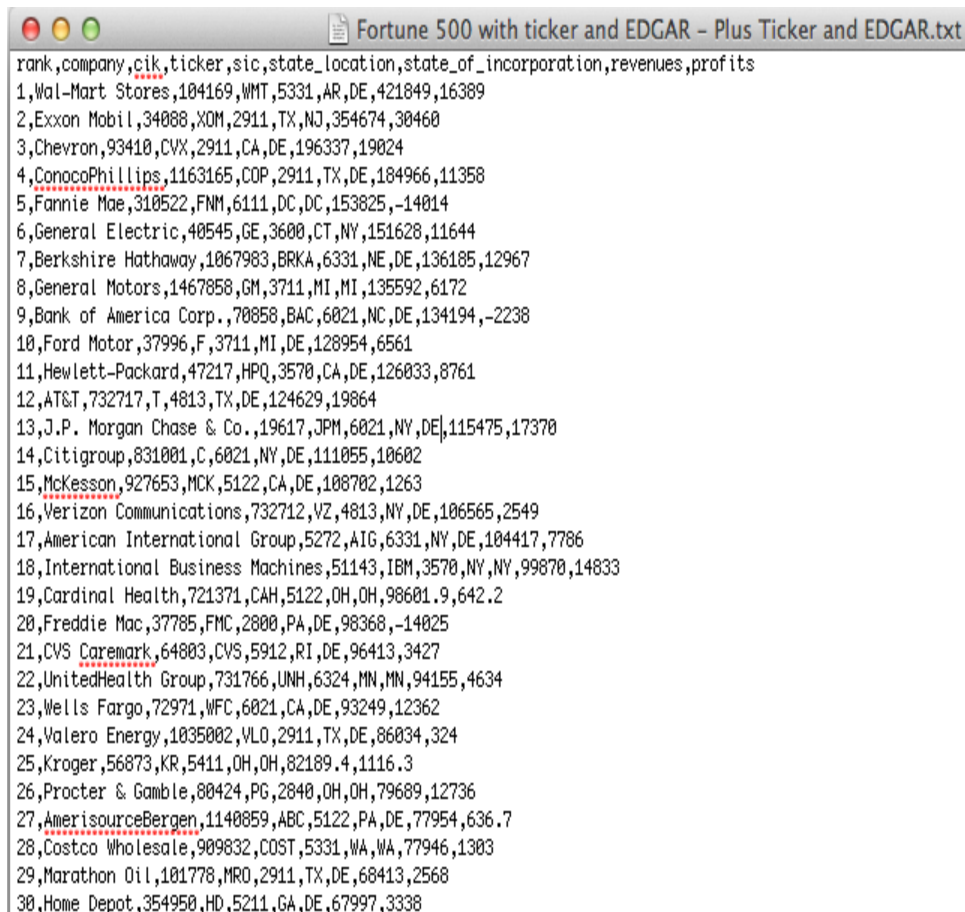
	C	D	E
	cik	ticker	sic
	104169	WMT	5331
	34088	XOM	2911
	93410	CVX	2911
	1163165	COP	2911
	310522	FNM	6111
	40545	GE	3600
	1067983	BRKA	6331
	1467858	GM	3711
	70858	BAC	6021
	37996	F	3711

20 Freddie Mac

21 CVS Caremark

Tabular Data (csv)

- Fortune 500

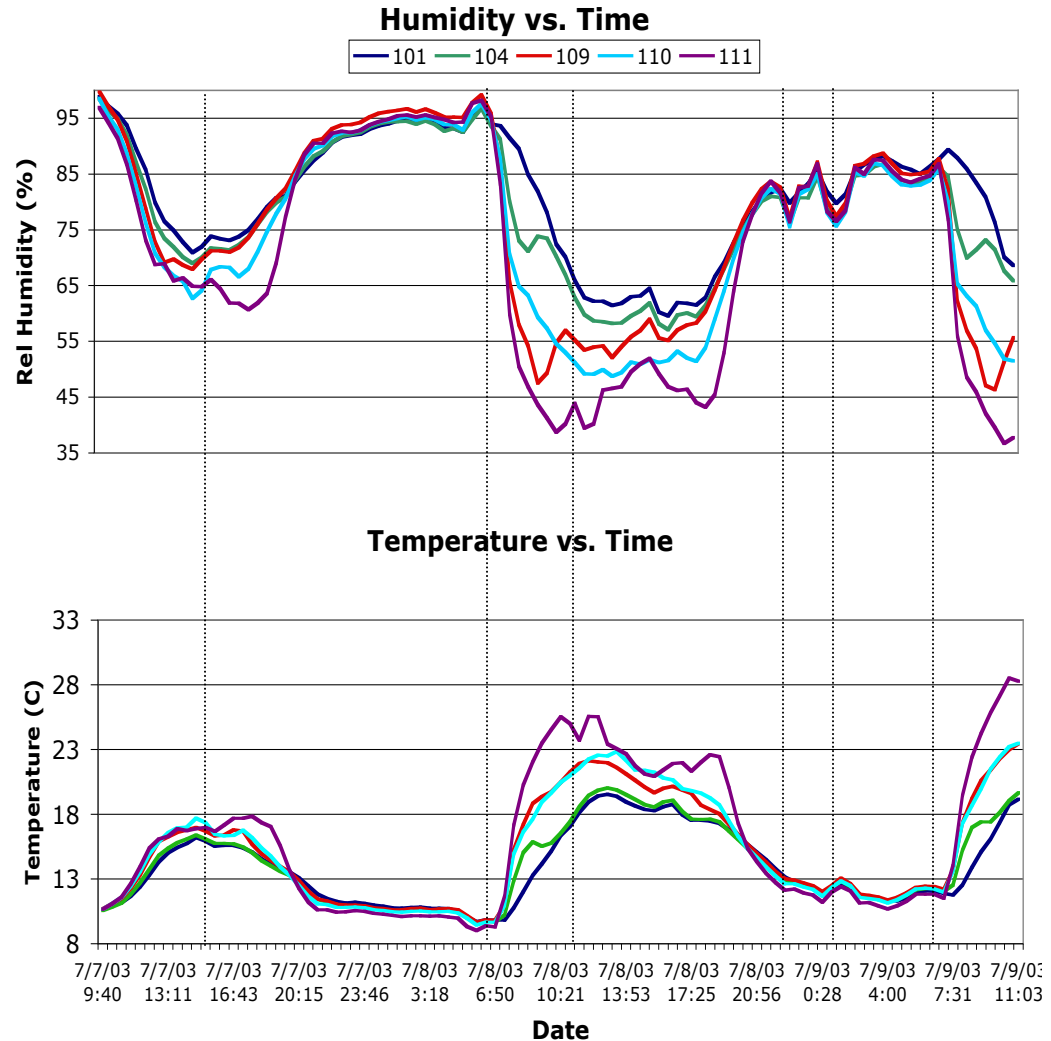
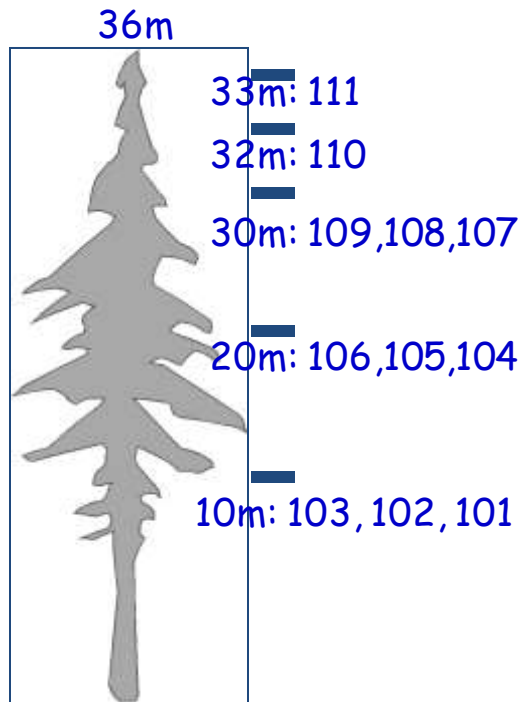


```
rank,company,cik,ticker,sic,state_location,state_of_incorporation,revenues,profits
1,Wal-Mart Stores,104169,WMT,5331,AR,DE,421849,16389
2,Exxon Mobil,34088,XOM,2911,TX,NJ,354674,30460
3,Chevron,93410,CVX,2911,CA,DE,196337,19024
4,ConocoPhillips,1163165,COP,2911,TX,DE,184966,11358
5,Fannie Mae,310522,FNM,6111,DC,DC,153825,-14014
6,General Electric,40545,GE,3600,CT,NY,151628,11644
7,Berkshire Hathaway,1067983,BRKA,6331,NE,DE,136185,12967
8,General Motors,1467858,GM,3711,MI,MI,135592,6172
9,Bank of America Corp.,70858,BAC,6021,NC,DE,134194,-2238
10,Ford Motor,37996,F,3711,MI,DE,128954,6561
11,Hewlett-Packard,47217,HPQ,3570,CA,DE,126033,8761
12,AT&T,732717,T,4813,TX,DE,124629,19864
13,J.P. Morgan Chase & Co.,19617,JPM,6021,NY,DE,115475,17370
14,Citigroup,831001,C,6021,NY,DE,111055,10602
15,McKesson,927653,MCK,5122,CA,DE,108702,1263
16,Verizon Communications,732712,VZ,4813,NY,DE,106565,2549
17,American International Group,5272,AIG,6331,NY,DE,104417,7786
18,International Business Machines,51143,IBM,3570,NY,NY,99870,14833
19,Cardinal Health,721371,CAH,5122,OH,OH,98601.9,642.2
20,Freddie Mac,37785,FMC,2800,PA,DE,98368,-14025
21,CVS Caremark,64803,CVS,5912,RI,DE,96413,3427
22,UnitedHealth Group,731766,UNH,6324,MN,MN,94155,4634
23,Wells Fargo,72971,WFC,6021,CA,DE,93249,12362
24,Valero Energy,1035002,VLO,2911,TX,DE,86034,324
25,Kroger,56873,KR,5411,OH,OH,82189.4,1116.3
26,Procter & Gamble,80424,PG,2840,OH,OH,79689,12736
27,AmerisourceBergen,1140859,ABC,5122,PA,DE,77954,636.7
28,Costco Wholesale,909832,COST,5331,WA,WA,77946,1303
29,Marathon Oil,101776,MRO,2911,TX,DE,68413,2568
30,Home Depot,354950,HD,5211,GA,DE,67997,3338
```

Protein Data Bank

HEADER APOPTOSIS 05-OCT-10 3IZA
TITLE STRUCTURE OF AN APOPTOSOME-PROCASPASE-9 CARD COMPLEX
COMPND MOL_ID: 1;
COMPND 2 MOLECULE: APOPTOTIC PROTEASE-ACTIVATING FACTOR 1;
COMPND 3 CHAIN: A, B, C, D, E, F, G;
COMPND 4 SYNONYM: APAF-1;
COMPND 5 ENGINEERED: YES
SOURCE MOL_ID: 1;
SOURCE 2 ORGANISM_SCIENTIFIC: HOMO SAPIENS;
SOURCE 3 ORGANISM_COMMON: HUMAN;
SOURCE 4 ORGANISM_TAXID: 9606;
SOURCE 5 GENE: APAF1, KIAA0413;
SOURCE 6 EXPRESSION_SYSTEM: SPODOPTERA FRUGIPERDA;
SOURCE 7 EXPRESSION_SYSTEM_TAXID: 7108;
SOURCE 8 EXPRESSION_SYSTEM_STRAIN: SF21;
SOURCE 9 EXPRESSION_SYSTEM_VECTOR_TYPE: INSECT VIRUS;
SOURCE 10 EXPRESSION_SYSTEM_PLASMID: PFASTBAC1
KEYWDS APOPTOSOME, APAF-1, PROCASPASE-9 CARD, APOPTOSIS
EXPDTA ELECTRON MICROSCOPY
AUTHOR S.YUAN,X.YU,M.TOPF,S.J.LUDTKE,X.WANG,C.W.AKEY
REVDAT 1 03-NOV-10 3IZA 0
SPRSDE 03-NOV-10 3IZA 3IYT
JRNL AUTH S.YUAN,X.YU,M.TOPF,S.J.LUDTKE,X.WANG,C.W.AKEY
JRNL TITL STRUCTURE OF AN APOPTOSOME-PROCASPASE-9 CARD COMPLEX
JRNL REF STRUCTURE V. 18 571 2010

Internet of Things: Example measurements



Tabular Data from Sensors

Challenges

- May be many missing fields (a particular sensor may not produce all types of output).
- Device may go offline for a while.
- Device may be damaged (permanently or intermittently).
- Timestamps usually critical but may not be accurate.
- Other meta-data (location, device ID) may have errors.

Log Files – Example Apache Web Log

Processes, usually daemons, create logs

e.g., httpd, mysqld, syslogd

- 66.249.65.107 - - [08/Oct/2007:04:54:20 -0400] "GET /support.html HTTP/1.1" 200 11179 "-" "Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
- 111.111.111.111 - - [08/Oct/2007:11:17:55 -0400] "GET / HTTP/1.1" 200 10801 "http://www.google.com/search?q=log+analyzer&ie=utf-8&oe=utf-8 &aq=t&rls=org.mozilla:en-US:official&client=firefox-a" "Mozilla/5.0 (Windows; U; Windows NT 5.2; en-US; rv:1.8.1.7) Gecko/20070914 Firefox/2.0.0.7"
- 111.111.111.111 - - [08/Oct/2007:11:17:55 -0400] "GET /style.css HTTP/1.1" 200 3225 "\"<http://www.loganalyzer.net/>\" "Mozilla/5.0 (Windows; U; Windows NT 5.2; en-US; rv:1.8.1.7) Gecko/20070914 Firefox/2.0.0.7"

Syslog – A Standard for System Messages

- Developed by Eric Allman (at Berkeley) as part of the Sendmail project
- Standardized by the IETF in RFC 3164 and RFC 5424
- Listens on port 514 using UDP
- Puts data in `/var/log/messages` by default
- Enables rich analysis:



Syslog

dhcp-47-129:DataScienceF14> syslog -w 10

Feb 3 15:18:11 dhcp-47-129 Evernote[1140] <Warning>: -[EDAMAccounting read:]: unexpected field ID 23 with type 8. Skipping.

Feb 3 15:18:11 dhcp-47-129 Evernote[1140] <Warning>: -[EDAMUser read:]: unexpected field ID 17 with type 12. Skipping.

Feb 3 15:18:11 dhcp-47-129 Evernote[1140] <Warning>: -[EDAMAuthenticationResult read:]: unexpected field ID 6 with type 11. Skipping.

Feb 3 15:18:11 dhcp-47-129 Evernote[1140] <Warning>: -[EDAMAuthenticationResult read:]: unexpected field ID 7 with type 11. Skipping.

Feb 3 15:18:11 dhcp-47-129 Evernote[1140] <Warning>: -[EDAMAccounting read:]: unexpected field ID 19 with type 8. Skipping.

Feb 3 15:18:11 dhcp-47-129 Evernote[1140] <Warning>: -[EDAMAccounting read:]: unexpected field ID 23 with type 8. Skipping.

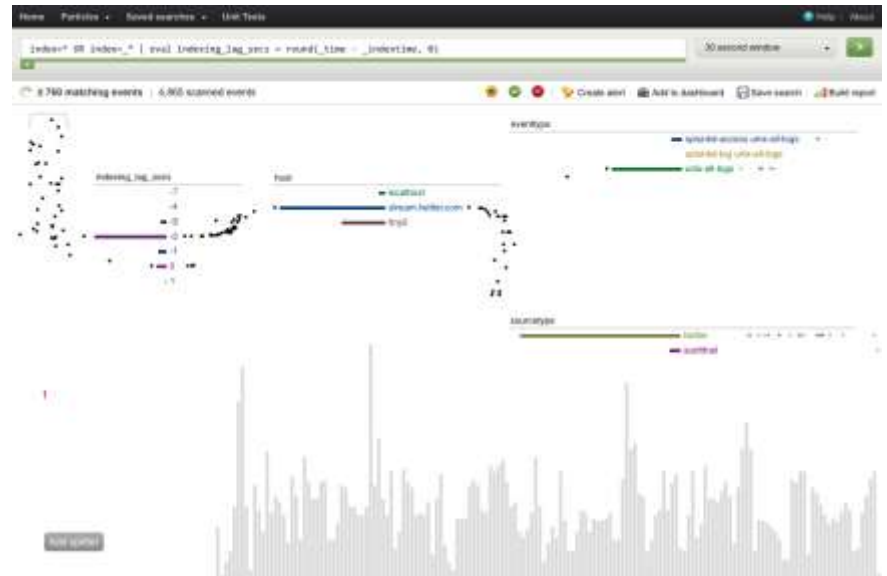
Feb 3 15:18:11 dhcp-47-129 Evernote[1140] <Warning>: -[EDAMUser read:]: unexpected field ID 17 with type 12. Skipping.

Feb 3 15:18:11 dhcp-47-129 Evernote[1140] <Warning>: -[EDAMSyncState read:]: unexpected field ID 5 with type 10. Skipping.

Feb 3 15:18:49 dhcp-47-129 com.apple.mtmd[47] <Notice>: low priority thinning needed for volume Macintosh HD (/) with 18.9 <= 20.0 pct free space

“Splunking”

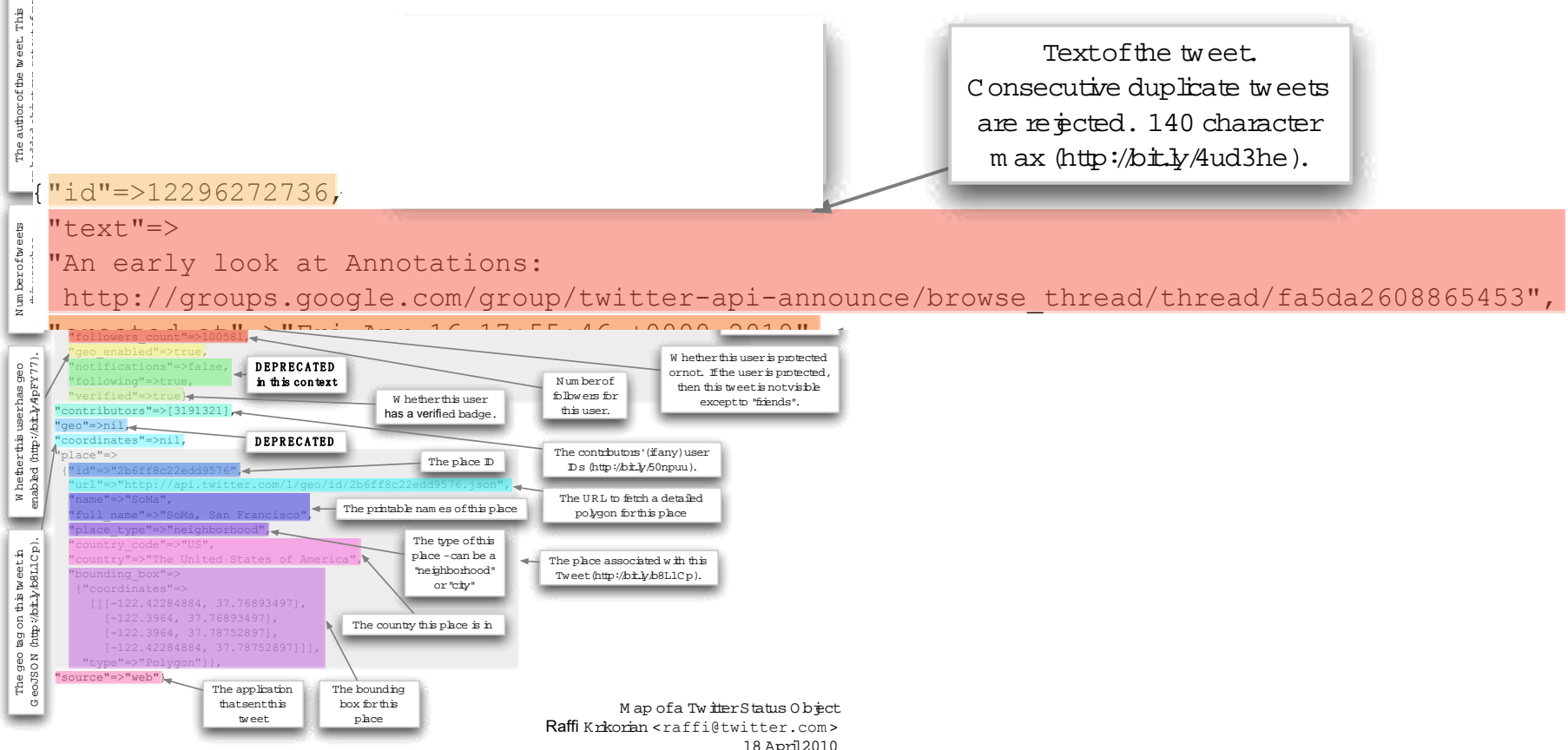
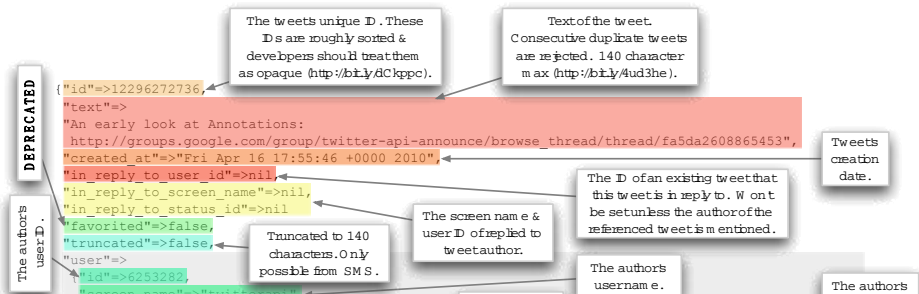
- Grab data from many machines
- Index it
- Check for unusual events:
 - Disk problems
 - Network congestion
 - Security attacks
- Monitor Resources
 - Network
 - Memory usage
 - Disk use, latency
 - Threads
- Dashboard for cloud administration.



Some Questions

- 1) How Many Characters are there in a Tweet?
- 2) How Many Bytes are there in a Tweet (msg)?

Tweet JSON Format



Tweet JSON Format

So how do we process the JSON from Twitter?

Stay tuned, but first some concepts from the XML world...

XML, DOM and SAX

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<!-- bookstore.xml -->
```

```
<bookstore>
```

```
  <book ISBN="0123456001">
```

```
    <title>Java For Dummies</title>
```

```
    <author>Tan Ah Teck</author>
```

```
    <category>Programming</category>
```

```
    <year>2009</year>
```

```
    <edition>7</edition>
```

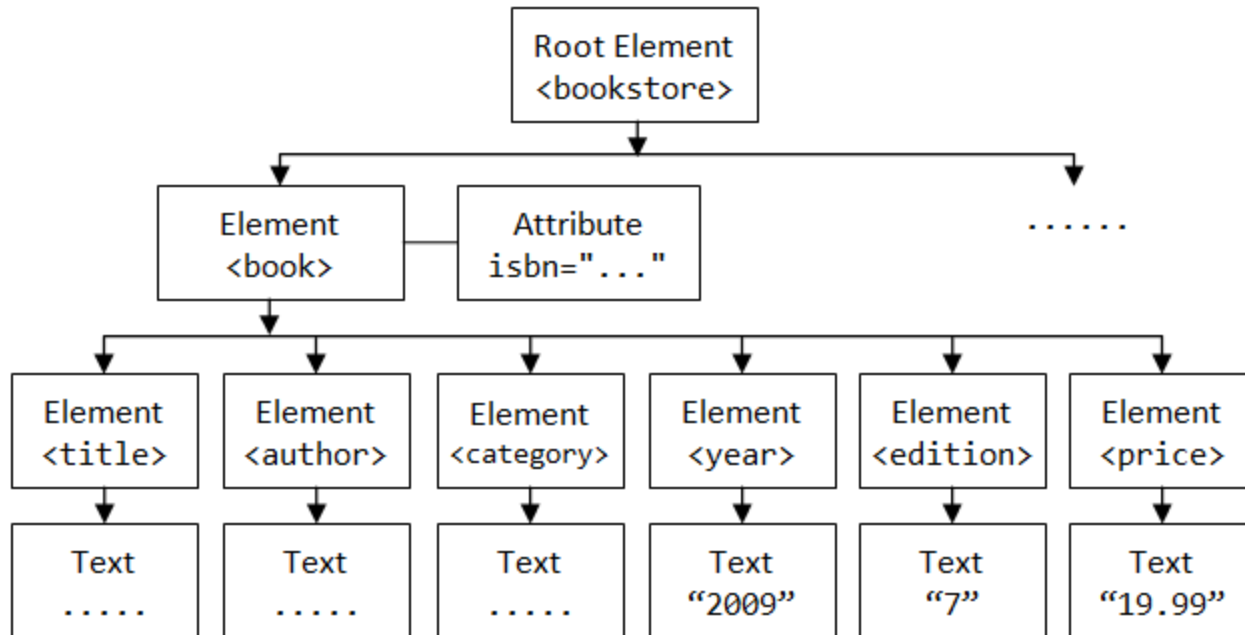
```
    <price>19.99</price>
```

```
  </book>
```

XML, DOM and SAX

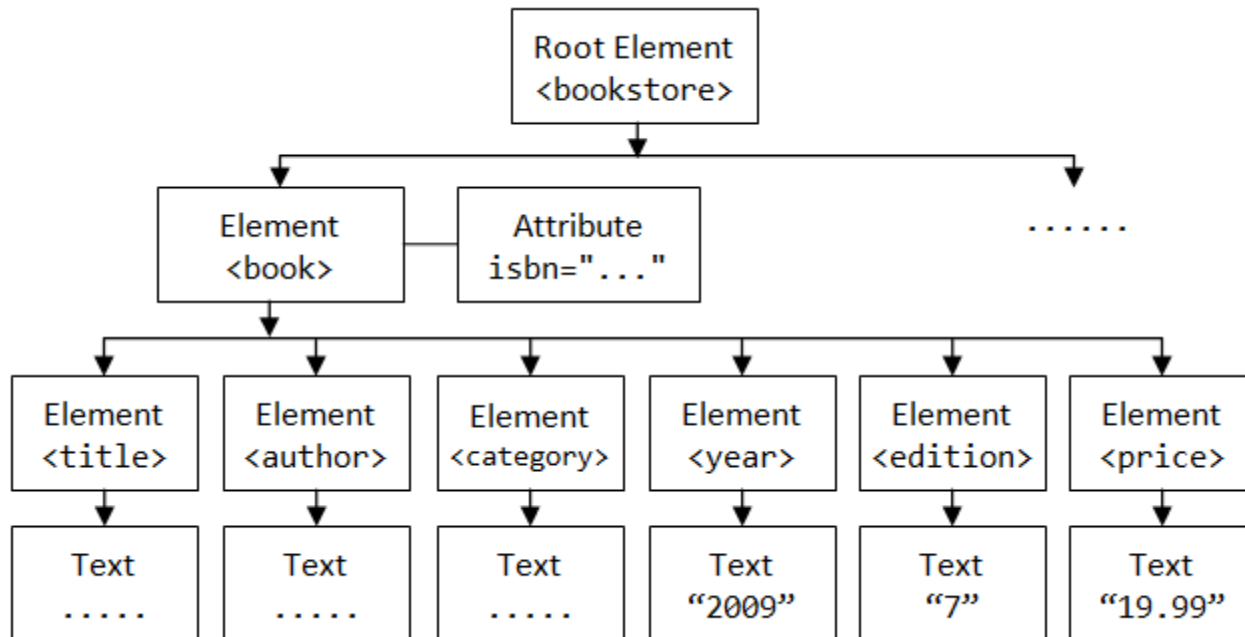
XML is a text format that encodes DOM (Document-Object Models) which is a data structure e.g. for Web pages.

The DOM is tree-structured:



XML, DOM and SAX

- The DOM is an easy object to work with: all the data in the object is accessible by links.
- The problem is that I might not care about most of the data, and I might not be able to fit the DOM for a large object in RAM.




SAX

SAX (Simple API for XML) is an alternative “lightweight” way to process XML.


A SAX parser generates a stream of events as it parses the XML file. The programmer registers handlers for each one.


It allows a programmer to handle only parts of the data structure.

SAX

`<?xml version="1.0" encoding="UTF-8"?>`  Document Header

`<!-- bookstore.xml -->`  Comment

`<bookstore>`  Start-element “bookstore”

`<book ISBN="0123456001">`  Start-element “book”

`<title>Java For Dummies</title>`  Start-element “title”


`<author>Tan Ah Teck</author>` End-element “title”

`<category>Programming</category>`

`<year>2009</year>`

`<edition>7</edition>`

`<price>19.99</price>`

`</book>`  End-element “book”

What about JSON?

Most JSON parsers construct the “DOM” directly.

But there are a few SAX-style parsers:

- Jackson
- JSON-simple

What about HTML?

- Common Crawl, **about 5 billion web pages**, between **0.2-0.5%** of Google's web crawl.
- 60 TB, hosted on Amazon S3, also available for download.
- Includes **link data, page rank**.
- In ARC (Internet Archive) File format.
- So there's plenty of data, and there are many crawlers for targeted exploration...
 - HTTrack, ...

HTML Tag Soup

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head><!-- types/widgets/pages/common/page.tmpl home/index_v3.html generated by index_v3 on Wed 29
Feb 2012 11:04:41 PM PST -->
<title>San Francisco Bay Area &mdash; News, Sports, Business, Entertainment, Classifieds: SFGate</title>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
<meta name="description" content="Find local news &amp; information, updated weather, traffic, classifieds,
sports scores, real estate, jobs, cars, food &amp; wine, travel, entertainment, events and more on SFGate.com.
Connect to the Bay Area community." />
<meta name="keywords" content="San Francisco, San Francisco Bay Area, news, local events, breaking news,
world news, San Francisco Chronicle, SFGate" />
<meta property="fb:page_id" content="105702905593" />
<meta property="fb:admins" content="653226748,658759748" />
<!-- /widgets/sitewide/css/all/inc.html widgets/pages/common/post_write_mtime/css_inc.tmpl -->
<!-- generated by sitewidecss on Thu 16 Feb 2012 10:41:53 AM PST -->
<link rel="stylesheet" type="text/css" title="SFGate" media="all"
href="http://imgs.sfgate.com/css1329417713/sitewide/css/sitewide.css" />
<!-- sitewide/css/all/inc.html end css_inc.tmpl -->
```

HTML Tools - Parsing

- “Beautiful Soup”
<http://www.crummy.com/software/BeautifulSoup/>
a Python API for handling real HTML. DOM or SAX interfaces.
- “TagSoup”
<http://ccil.org/~cowan/XML/tagsoup/>
provides a Sax interface, i.e. a streaming parse, to Java applications. Can transform to a format you want using XSLT.
- Taggle, part of the Arabica toolset
<http://www.jezuk.co.uk/cgi-bin/view/arabica/code>
is a version of TagSoup written in C++. You probably want to use this if you have a lot of data.

Web Services

Most large web sites today actively discourage screen-scraping to get their content, and provide Web Service APIs instead.

This is the “right” way to get data from online sources.

Web Services

W3C definition:

a "Web service" as "a software system designed to support interoperable machine-to-machine interaction over a network".

Two kinds:

- XML-based RPC-style messages: WSDL and SOAP
- REST-style stateless interactions, URLs encode state

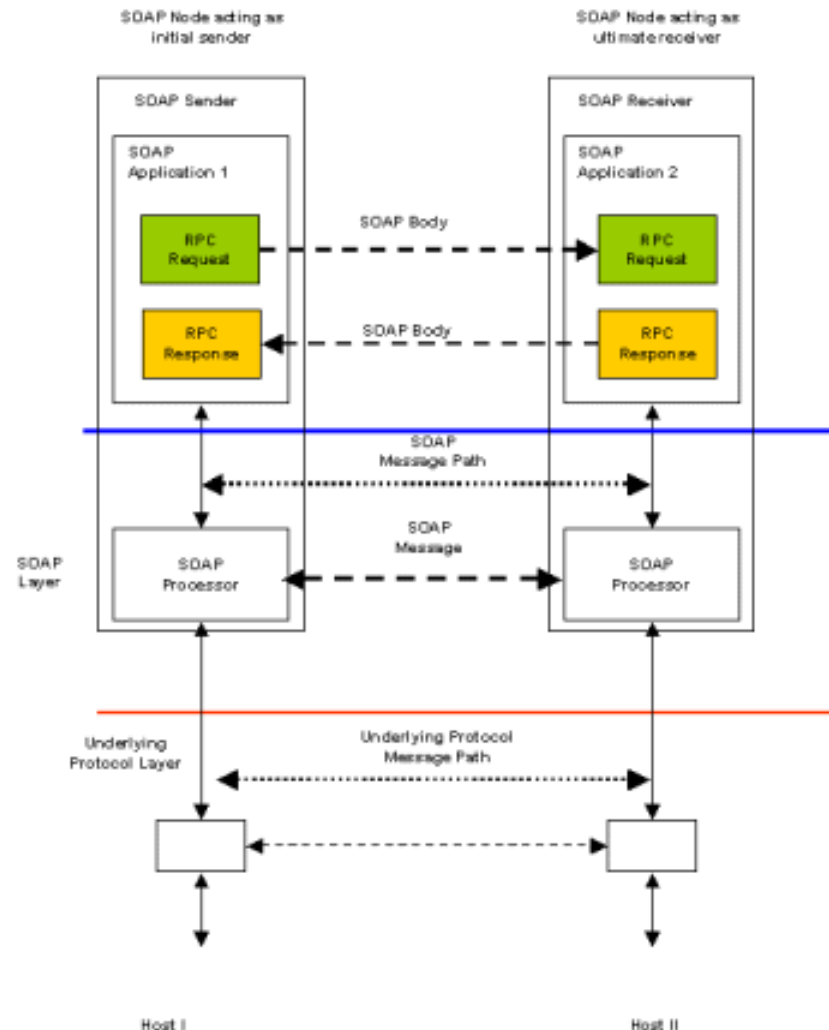
Can run over different transports, but usually HTTP

Examples

- Twitter:** REST API and streaming API with JSON content. Provides sampling, searching and filtering capabilities.
- Amazon:** has a “product advertising API” in XML with a WSDL spec. Includes product search, reviews etc.
- Livejournal:** RSS/Atom + custom XML/RPC. Search by keyword, topic, follow friend links.
- Netflix:** Javascript, Atom and REST interfaces.
- Ebay:** Many APIs for searching, buying and posting. WSDL descriptions, client code in Java and .NET
- Flickr:** Comprehensive API set, free for non-commercial use. REST, XML-RPC, SOAP, with client code in many languages.
- vBulletin:** REST interface, most actions supported

Web Services

XML-RPC, requires a request-response cycle. Often longer “conversations.”



WSDL and SOAP

- Conceptually a Remote-Procedure-Call system, like CORBA, Java RMI etc.
- But RPC often has to pass through multiple layers of firewalls, causing many problems as security increased in the 1990s.
- Web services typically use HTTP as a transport, which runs almost anywhere, provides security and simple GET-POST methods. So HTTP-based RPC was a natural choice.
- SOAP uses XML messages, is human-readable, easy to process from any programming language, and relatively robust to version slippage.

SOAP is

SOAP covers the following four main areas:

- A **message format** for one-way communication describing how a message can be packed into an XML document.
- A **description** of how a SOAP message should be transported using HTTP (for Web-based interaction) or SMTP (for e-mail-based interaction).
- A **set of rules** that must be followed when processing a SOAP message and a simple classification of the entities involved in processing a SOAP message.
- A **set of conventions** on how to turn an RPC call into a SOAP message and back.

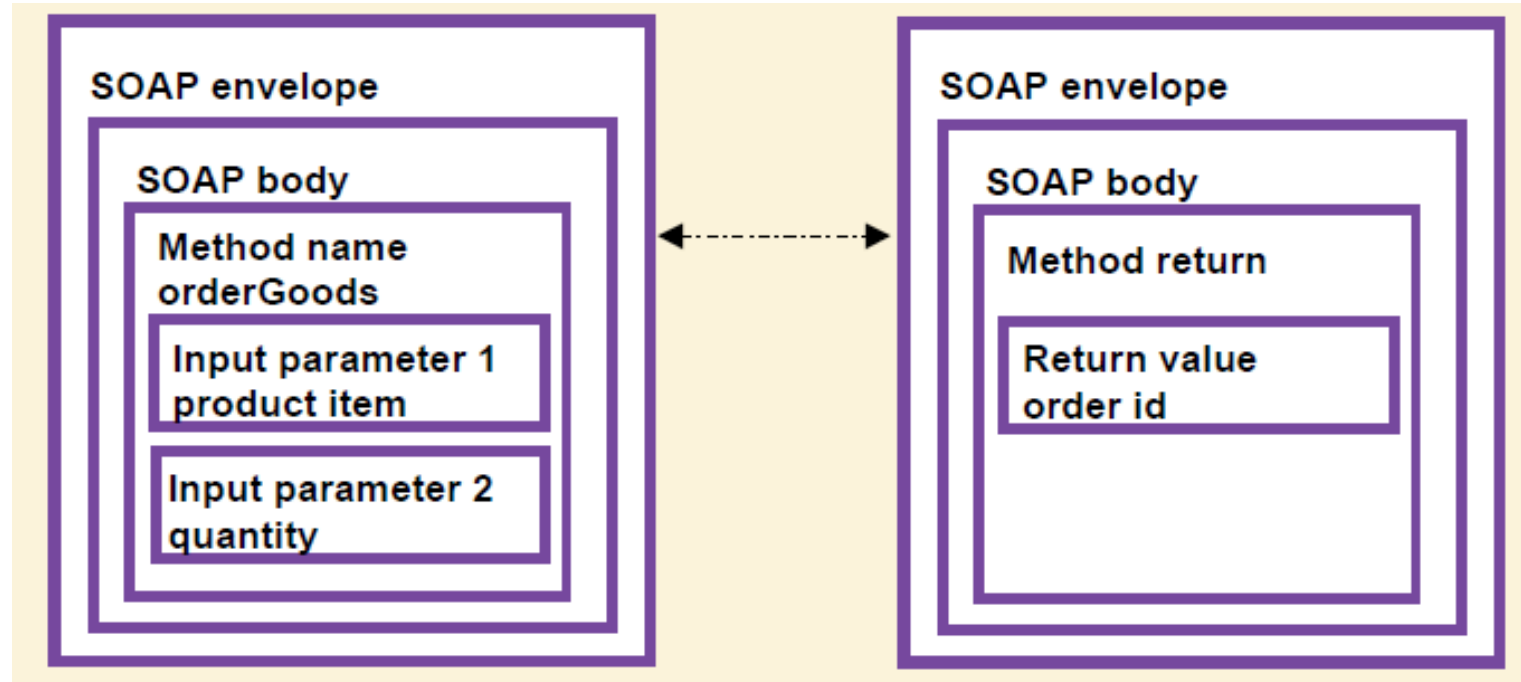
Soap Message

Typically an XML element containing header and body elements

```
<SOAP:Envelope xmlns:SOAP=
    "http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Header>
    <!-- content of header goes here -->
  </SOAP:Header>
  <SOAP:Body>
    <!-- content of body goes here -->
  </SOAP:Body>
</SOAP:Envelope>
```

SOAP RPC

SOAP RPC messages typically encode arguments that are presented to the calling program as parameters and return values:



Soap RPC

```
POST /travelservice
SOAPAction: "http://www.acme-travel.com/flightinfo"
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn

<SOAP:Envelope xmlns:SOAP=
  "http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Body>
    <m:GetFlightInfo
      xmlns:m="http://www.acme-travel.com/flightinfo"
      SOAP:encodingStyle=
        "http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi=
        "http://www.w3.org/2001/XMLSchema-instance">
      <airlineName xsi:type="xsd:string">UL
    </airlineName>
      <flightNumber xsi:type="xsd:int">506
    </flightNumber>
    </m:GetFlightInfo>
  </SOAP:Body>
</SOAP:Envelope>
```

Soap Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn

<SOAP:Envelope xmlns:SOAP=
  "http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Body>
    <m:GetFlightInfoResponse
      xmlns:m="http://www.acme-travel.com/flightinfo"
      SOAP:encodingStyle=
        "http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi=
        "http://www.w3.org/2001/XMLSchema-instance">
      <flightInfo>
        <gate xsi:type="xsd:int">10</gate>
        <status xsi:type="xsd:string">ON TIME</status>
      </flightInfo>
    </m:GetFlightInfoResponse>
  </SOAP:Body>
</SOAP:Envelope>
```

REST

REpresentation State Transfer

Stateless Client/Server Protocol: Principles

1. Each message in the protocol contains all the information needed by the receiver to understand and/or process it. This constraint attempts to “keep things simple” and avoid needless complexity
2. Set of Uniquely Addressable Resources
 - “Everything is a Resource” in a RESTful system
 - Requires universal syntax for resource identification (e.g. URI)

REST

3. Set of Well-Defined Operations that can be applied to all resources
 - In context of HTTP, the primary methods are
 - POST, GET, PUT, DELETE
 - these are similar (but not exactly) to the database notion of
 - CRUD (Create, Read, Update, Delete)

4. The use of Hypermedia both for Application Information and State Transitions
 - Resources are typically stored in a structured data format that supports hypermedia links, such as XHTML or XML

REST example

<user>

<name>Jane</name>

<gender>female</gender>

<location href="http://www.example.org/us/ny/new_york">

New York City, NY, USA</location>

</user>

This documentation is a representation used for the User resource

It might live at <http://www.example.org/users/jane/>

- If a user needs information about Jane, they GET this resource
- If they need to modify it, they GET it, modify it, and PUT it back
- The href to the Location resource allows savvy clients to gain access to its information with another simple GET request

Implication: Clients cannot be “thin”; need to understand resource formats

REST vs. RPC

In RPC systems, the design emphasis is on **verbs**

- What operations can I invoke on a system?
- getUser(), addUser(), removeUser(), updateUser(), getLocation(), updateLocation(), listUsers(), listLocations(), etc.

In REST systems, the design emphasis is on **nouns**

- User, Location
- In REST, you would define XML representations for these resources and then apply the standard methods to them

Files



- What is a file?
 - A **file** is a named sequence of **bytes**
 - Typically stored as a collection of pages (or blocks)
 - A **filesystem** is a collection of files organized within an hierarchical namespace
 - Responsible for laying out those bytes on physical media
 - Stores file metadata
 - Provides an API for interaction with files
 - Standard operations
 - `open()/close()`
 - `seek()`
 - `read()/write()`

Files

- Hierarchical namespace
 - / is known as the root of a filesystem
 - On Linux, the Filesystem Hierarchy Standard specifies which files live where
 - System executables in /usr/bin
 - Log files in /var/log
 - Permissions can be applied to all files beneath a directory
 - Files are not always arranged in a hierarchical namespace
 - Content-addressable storage (CAS)
 - Often used for large multimedia collections

File Formats

- Considerations for a file format
 - Data model: tabular, hierarchical, array
 - Physical layout
 - Field units and validation
 - Metadata: header, side file, specification, other?
 - Plain text or binary
 - Encoding: ASCII, UTF-8, other?
 - Delimiters and escaping
 - Compression, encryption, checksums?
 - Schema evolution

File Performance

Read/Write time (180 MB tabular file)

	Read Time (Text)	Write Time (Text)	Read Time (Binary)	Write Time (Binary)
Pandas (Python)	88 secs	107 secs	**	**
Scala/Java	12 secs	30 secs	0.5-2* secs	0.5-2* secs

** Pandas doesn't have a default binary file I/O library –
you can use Python, but performance depends on what you pick.

* 2 seconds is the time for sustainable read/write.
May be faster due to caching

File Performance - Compression

Read/Write time (180 MB tabular file, Scala/Java)

Binary File	Read Time	Write Time	File Size
Gzip level 6 (Java default)	1-2 secs	11 secs	35 MB
Gzip level 3	1-2 secs	4 secs	37 MB
Gzip level 1	1-2 secs	2 secs	38 MB
LZ4 fast	1-2 secs	1-2 secs	63 MB

Text File	Read Time	Write Time	File Size
Gzip level 6 (default)	1-2 secs	22 secs	32 MB
Gzip level 3	1-2 secs	14 secs	35 MB
Gzip level 1	1-2 secs	12 secs	39 MB
LZ4 fast	1-2 secs	11 secs	63 MB