

# Patterns of Use

KAN Min-Yen

# Patterns of Use

- Users often find other patterns of use than the software designer's original intent
- Best practice: Release early and often; correct fixes and add features as needed

# An Urban Legend



... says that the architects of the walking paths of Cornell University ...

... let them evolve by themselves by observing how people walked between buildings and then finally paving those that were popular and functional.

# What we are going to do today

Review the technologies in the past to the present:

- The search engine
- The scholarly paper
- The web browser

And project this to the future from the present:

- The mobile web browser
- The scholarly paper

# Google<sup>SM</sup>

[About Google](#)

[Jobs@Google](#)

Enter your search terms...

Google Search

I'm Feeling Lucky

...or [browse web pages](#) by category.



Google Search

I'm Feeling Lucky

Google.com.sg offered in: [中文\(简体\)](#) [Bahasa Malaysia](#) [தமிழ்](#)



hurricane

- hurricane
- hurricane irene
- hurricane katrina
- hurricane lyrics



Advanced search

Search

**Everything**

Images

Maps

Videos

News

More

**Singapore**

Change location

**The web**

Pages from Singapore

**Any time**

Past hour

Past 24 hours

Past 2 days

Past week

Past month

Past year

Custom range...

**All results**

Sites with images

Timeline

[Tropical cyclone - Wikipedia, the free encyclopedia](#)

[en.wikipedia.org/wiki/Tropical\\_cyclone](http://en.wikipedia.org/wiki/Tropical_cyclone)

Jump to [Hurricane or typhoon](#): A **hurricane** or typhoon (sometimes simply referred to as a tropical cyclone, as opposed to a depression or storm) is a ...

[Hurricane \(disambiguation\)](#) - [Scales](#) - [Tropical cyclones portal](#) - [Extratropical cyclone](#)

[Hurricane Katrina - Wikipedia, the free encyclopedia](#)

[en.wikipedia.org/wiki/Hurricane\\_Katrina](http://en.wikipedia.org/wiki/Hurricane_Katrina)

**Hurricane** Katrina of the 2005 Atlantic **hurricane** season was the costliest ...

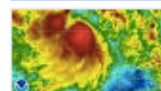
[Hurricane Irene \(2011\) - Wikipedia, the free encyclopedia](#)

[en.wikipedia.org/wiki/Hurricane\\_Irene\\_\(2011\)](http://en.wikipedia.org/wiki/Hurricane_Irene_(2011))

**Hurricane** Irene was a large and powerful Atlantic **hurricane** that left ...

[+](#) [Show more results from wikipedia.org](#)

[Hurricane Jova Gains Strength in Eastern Pacific](#)



CTV.ca

Fox News - 2 hours ago

AP The US National **Hurricane** Center in Miami said Sunday evening that Jova had maximum sustained winds of 90 mph and was centered about 305 miles southwest ...

[733 related articles](#)

[Hurricane Jova strengthens off Mexico's west coast - Reuters](#)

[More news for hurricane »](#)

[Images for hurricane](#) - Report images



What's the difference?



# What's the difference?

- Regional
- Auto suggest / Query Suggest
- Auto search
- Page Previews
- Universal Search
- Contextual Advertising

# Patterns of use in scientific articles

“What better contribution could a scholar make than an article which could ... provide a clear, but vivid argument to the [secondary school student] but which, if unraveled, could provide the rigor demanded by the most crusty specialist?” Crane (of the Perseus DL)

- Question: How do DL designers support this in terms of HCI?
- Answer: Creating different document layers. Allow users to “fold” the document to see the only the relevant portions.

Fisheye

## Executable Object Modeling with Statecharts

David Harel & Eran Gery, Computer, JULY 1997, 30 no. 7, 31-42

Statecharts, popular for modeling system behavior in the structural analysis paradigm, are part of a fully executable language set for modeling object-oriented systems. The languages form the core of the emerging Unified Modeling Language.

For the development of object-oriented systems should be behaviorally expressive and rigorous as well as intuitive and well structured. Thus, any modeling approach must be detailed and precise enough to produce fully executable models and permit the automatic synthesis of efficient code in languages such as C++.

Most OO modeling methodologies specify a model through graphical notations. Entity-relationship-like diagrams typically specify object classes and their interrelationships, and there is some way to describe what objects do and how they interact. Most methodologies also adopt a state-based formalism to specify behavior, using statecharts[1] or some sublanguage thereof.

However, many methodologies fail to rigorously define the semantics of the languages. Without a rigorous semantic definition, precise model behavior over time is not well defined and full executability and automatic code synthesis is impossible. Adopting a richly expressive behavioral language like statecharts makes modeling easier, but requires great care in defining the way it integrates with the other parts of the model. Statecharts must capture not only the state of the object as a precondition to service requests, but also the dynamics of the object's internal behavior in responding to those requests and in maintaining relationships with other objects.

### RAILCAR SYSTEM

To explain the properties of our language set, we use the automated railcar system in Figure 1, inspired by Yehud Gafni. Six terminals are located on a cyclic path. Each pair of adjacent terminals is connected by two rail tracks, one for clockwise and one for counterclockwise travel. Several railcars are available to transport passengers between terminals. A control center receives, processes, and sends system data to various components.



Our current implementation framework is based on C++, which is natural given its status in the OO language community. However, this is more a matter of convenience, so that models contain actions and operations written directly in the implementation language. This, in turn, makes it relatively easy to plug in a framework based on another language, such as Ada, Smalltalk, Java, or even on a set-based language [5]. However, what programming language is chosen as the implementation framework has little bearing on our modeling and analysis approach. Rhapsody supports the modeling process in its entirety, so once we chose C++ for our initial implementation, it became natural to use it for the detail level of the model, too.



## Executable Object Modeling with Statecharts

David Harel & Eran Gery, Computer, JULY 1997, 30 no. 7, 31-42

Statecharts, popular for modeling system behavior in the structural analysis paradigm, are part of a fully executable language set for modeling object-oriented systems. The languages form the core of the emerging Unified Modeling Language.

Models for the development of object-oriented systems should be behaviorally expressive and rigorous as well as intuitive and well structured. Thus, any modeling approach must be detailed and precise enough to produce fully executable models and permit the automatic synthesis of efficient code in languages such as C++.

Most OO modeling methodologies specify a model through graphical notations. Entity-relationship-like diagrams typically specify object classes and their interrelationships, and there is some way to describe what objects do and how they interact. Most methodologies also adopt a state-based formalism to specify behavior, using statecharts[1] or some sublanguage thereof.

However, many methodologies fail to rigorously define the semantics of the languages. Without a rigorous semantic definition, precise model behavior over time is not well defined and full executability and automatic code synthesis is impossible. Adopting a richly expressive behavioral language like statecharts makes modeling easier, but requires great care in defining the way it integrates with the other parts of the model. Statecharts must capture not only the state of the object as a precondition to service requests, but also the dynamics of the object's internal behavior in responding to those requests and in maintaining relationships with other objects.

These issues are complicated and go beyond recommending a modeling approach or methodology—they are language design concerns, requiring rigorous mathematical underpinnings. Both syntax and semantics must be fully worked out, any possible combination of constructs must be clearly characterized as syntactically legal or illegal, and each legal combination must be given a unique and formal meaning.

To address these needs, we embarked on an effort to develop an integrated set of diagrammatic languages for object modeling, built around statecharts, and to construct a supporting tool that produces a fully executable model and allows automatic code synthesis. The language set includes two constructive modeling languages (languages containing the information needed to execute the model or translate it into executable code).

- Object model diagrams specify system structure by identifying object classes and their multiplicities, object relationships and roles, and subclassing relationships
- Statecharts describe system behavior. A statechart attached to a class specifies all behavioral aspects of the objects in that class

Overview +  
Details

- Overview + Details shown as best (Hornbaek & Frokjaer 01)
  - Fisheye distortion unsatisfactory
  - Shown better for QA but not for whole document understanding

# The scientific article

How *do* we use articles?

Answer these in groups:

- Do we use scientific articles as a whole? Or specific *components*?
- How do *you* (personally) determine the relevance of an article?
- When do you decide to read an article?
- (Harder) What parts of an article do you use, and for what purpose / task?
- How do you categorize or label the articles that you read?

Typical critical reading patterns:

1. Read the title and the abstract  
If you still don't know what this paper is about, then this is a poorly-written paper.
2. Read the conclusion  
Are you now sure you know what this paper is about? If not, throw it away.
3. Read the introduction
4. Read the section headings
5. Read tables and graphs and captions

# Usage lifecycle of an article

- Being found as relevant
- Assessing relevance
- Document surrogate
- “Information finding”
  - Browsing for exploration
  - Searching for specific bits
- Conveying knowledge not easily rendered in words

# Being found as relevant

- Advanced features of search not often used
  - “Just to be safe”, use full text
  - Common and well-understood UI (legacy effect)
  - When features failed, users often don't try them again
- Features thus need:
  - To be properly introduced / understood (scaffolding)
  - To have well-understood error messages

# Searching for specific bits

One-shot queries rare:

– Tip of the larger iceberg of an information seeking pattern

- I look for specific surface tensions, experimental measurements
- Looking for best efficiency of electric motors.
  - Ended up reading tons of documents for electric motor
- I sometimes want to look specifically at other's methods and theories
- I often need multiple copies of a specific piece, like a table, for class
- I need to keep up to date on my research area

# Browsing

- Why do people browse?
  - Semi-directed / Undirected learning
  - Initial Exploration
- Collection Evaluation
  - What's in this collection? Is it relevant to my objectives?
- Subject Exploration
  - How well does this collection cover my area of interest?
- Query Exploration
  - What kind of queries will succeed in this area? How can I access this collection?



# Using the article

- Reading has different purposes too:
  - General Learning
  - Identification
  - Skimming
  - Answer questions
  - Defend position
  - Cross-Reference
  - Editing or critical review

# Using the article (2)

- Biased to particular user and task
  - Current researcher's work as “lens” to view the work
  - Different workflow for different users
    - Beginning researchers
    - Seasoned veterans
    - E.g., when to do annotation? Read references?
- Writing goes hand in hand with reading:
  - Three levels: Creating, note-taking and annotation
  - Annotation serves not so much to add to an article:
    - But to extract / filter important nuggets from an article (e.g., highlighting)
    - Adding a “document layer” to be used to view the document
    - Also inter-document annotation (e.g., labeling)

# Patterns of use of the web browser

- How do people use query the web?
- How do they use the web browser?
- How can we build a better web browser?

# Web query types (revisited)

What features are best for differentiating web search intent?

- Discriminate using *mutual information* for 2+ word queries  
 $P(x,y) / P(x) P(y)$  – collocation corrected for chance  
High MI corresponds to navigational task
- Navigational (Known item, Home page finding)
  - Relevant pages are mostly entry (root) pages
  - Anchor text and URL information
- Informational (Topic relevance)
  - Relevant pages are mostly nested pages
  - Content information (e.g., TF × IDF)

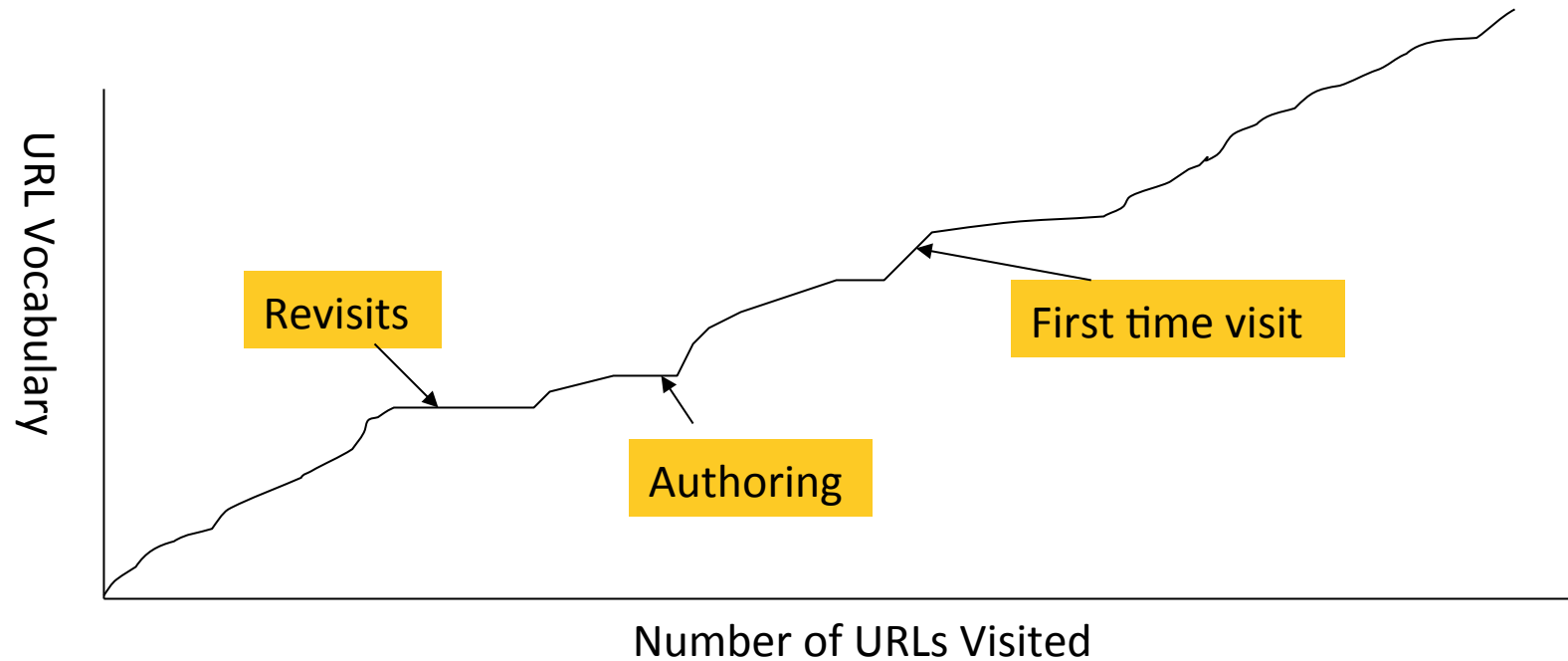
# User behavior

- Users tend not to use monitoring steps
  - Sign up for email alerts, RSS, create a channel
- Even in a formal search mode
  - Users use simple keyword search, not advanced
  - Don't revise their queries often (75% of all searches)
  - Don't access help
- Users don't seem to have strongly repetitive patterns within a cluster of pages
  - No consistent paths
  - Longest repeated sequence analysis fails
- Larger volume of queries
  - Higher percentage of repetition
  - Caching is a good strategy

# Page navigation types

- ~40% by following hyperlinks
- ~20-50% by back button navigation
- 11% new window
- 10% other (pop-ups count here)
  - Should be counted in hyperlink following
- 2.5% by bookmarks
- 0.8% by history

# URL Vocabulary



- Observed linear growth, not power law
  - Why?

# Modes of web browsing

Tauscher and Greenberg (1997):

- *First time visit*: new URLs observed
- *Revisits*: reading in depth (e.g., course notes), flicking to previous page(s)
- *Authoring of pages*: `reload` heavily used
- *Using web-based applications*: form submissions
- *Hub-and-spoke*: central page  $\Rightarrow$  specific page and back
- *Guided Tour*: Viewing a many-page article



# Scenario

- You went to a website this afternoon to do some fact-finding for a project that you're working on.

After going through many sites, some reading you're currently doing reminds of a link that would be useful to visit on a page that you visited sometime in the last hour or two.

How would you go about finding it?

Your answers:

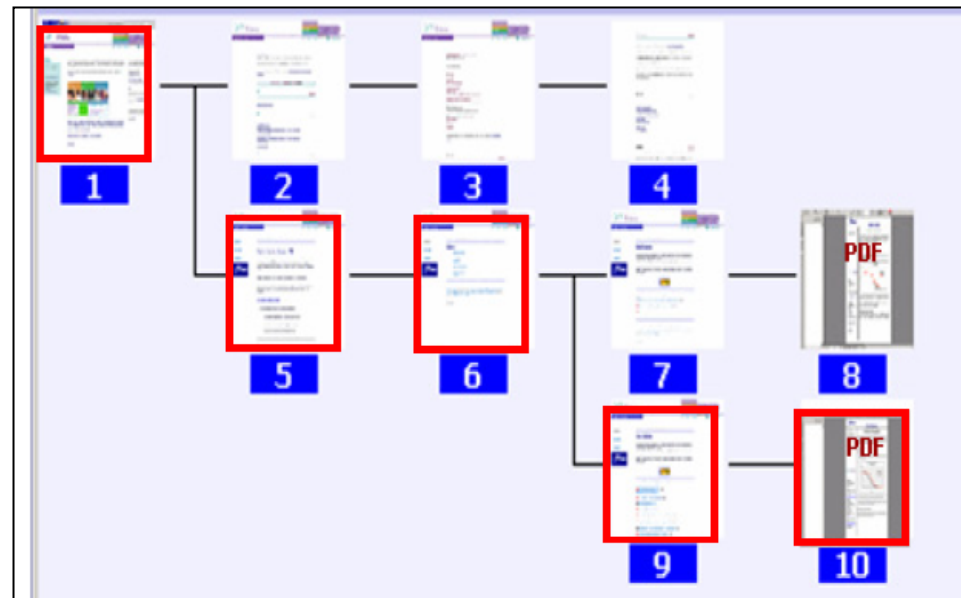
A large, empty rectangular box with a black border, intended for the user to write their answers to the question above.

# The Back Button

- Takes you to the previous page
  - With a reverse-order of chronological pages; i.e. a stack
  - Extremely simple and easy to use
- How would you improve upon this?
- A UI feature of web browsers that have made it into operating systems

# Temporal model of revisiting

- Promote a previously visited page to the top of the stack if:
  - I go back to visit it and
  - I take a different hyperlink from there



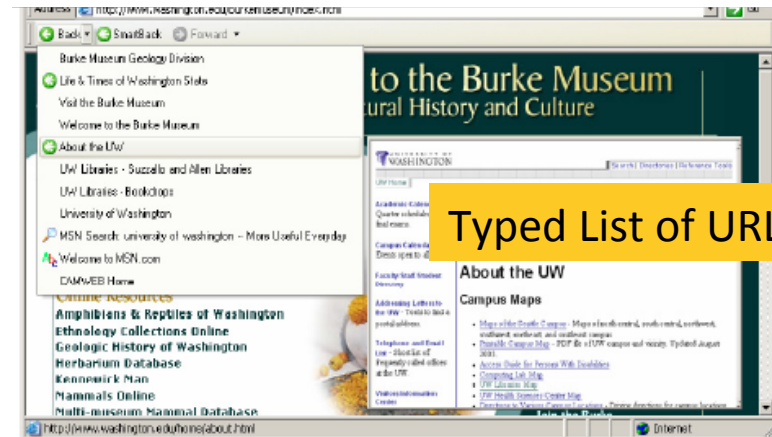
# The navigation hub

- Hub: a page that was promoted in the previous algorithm
- Study shows hubs revisited 1.8 times
- Ideally, predict which pages would be revisited

# Algorithm for finding hubs

- Safari Browser: Search Engine and typed URLs as hubs
- Previous revisit of a page indicates hub
  - Even across sessions (“new window” commands)
  - Points to per-user customization

- SmartBack



# Where does this lead us to?



What are the usage patterns of mobile web browsers? What tasks do they need to support?

What are the device characteristics and the characteristics of the environment?

# Mobile Web Browsers (ca. 2011)

- Geospatial, temporally aspects of search
- I/O difficult. Small screen, keyboard, leading to the possibility of NUI
- Bandwidth limited, power-draining websites, scripting

## Environment

- Introduction of HTML 5 for desktops
- One of many (proliferating) apps on mobile devices
- The playing field is still being defined

Your thoughts?



# Conclusions

- Patterns of use collaborate with the user interface of a system
  - And also with Web 2.0, with the authoring (mobile web site design)
- Now at a crux where much improvement coming due to changes in devices
- Look back and check your predictions 5 years from now