Information Fractal Is

Information Fractal Is

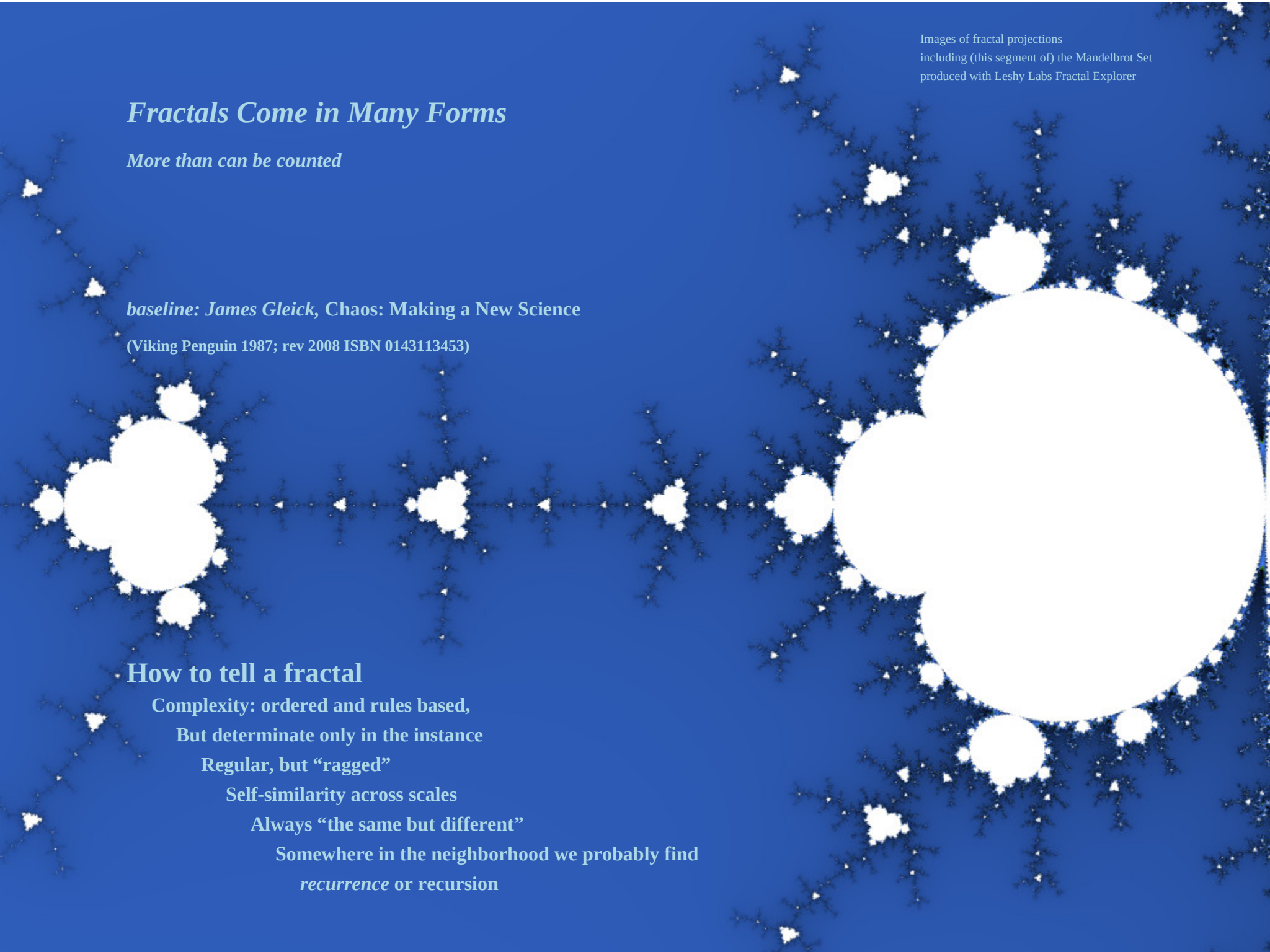
Information Fractal Is

Information Fractal Is

Information Fractal Is

Information Fractal Is

Wendell Piez
Balisage 2018
Rockville, Maryland



Images of fractal projections
including (this segment of) the Mandelbrot Set
produced with Leshy Labs Fractal Explorer

Fractals Come in Many Forms

More than can be counted

***baseline: James Gleick,* Chaos: Making a New Science**

(Viking Penguin 1987; rev 2008 ISBN 0143113453)

How to tell a fractal

Complexity: ordered and rules based,

But determinate only in the instance

Regular, but “ragged”

Self-similarity across scales

Always “the same but different”

Somewhere in the neighborhood we probably find

***recurrence* or recursion**

We *conceive* fractals with mathematics

We *observe* fractals in nature

Cultural productions also exhibit fractal features

... whether by “accident” or “design”

Cultural production - the archive!

Documentary production (or: the written word)

Electronic/documentary media

Non-proprietary, open, standards-based media

Text-based formats

Formalisms, formal languages, programming languages

Markup languages and data description syntaxes

IANAM! (Mathematician)

It's all about regularity

Or (contrary-wise) scope definition and exception handling

“What do you mean by that, exactly?” is like “How long is your coastline?”

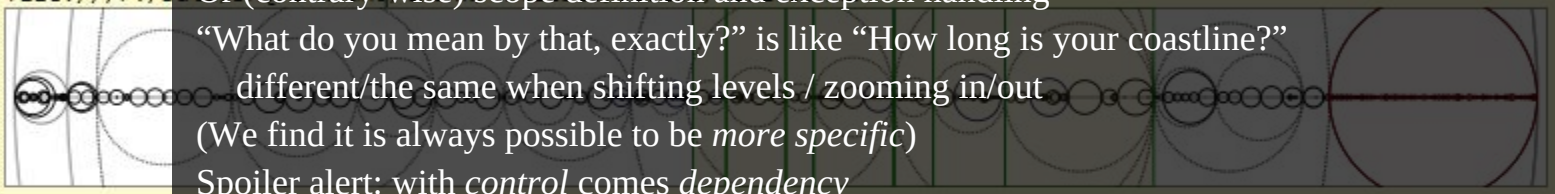
different/the same when shifting levels / zooming in/out

(We find it is always possible to be *more specific*)

Spoiler alert: with *control* comes *dependency*

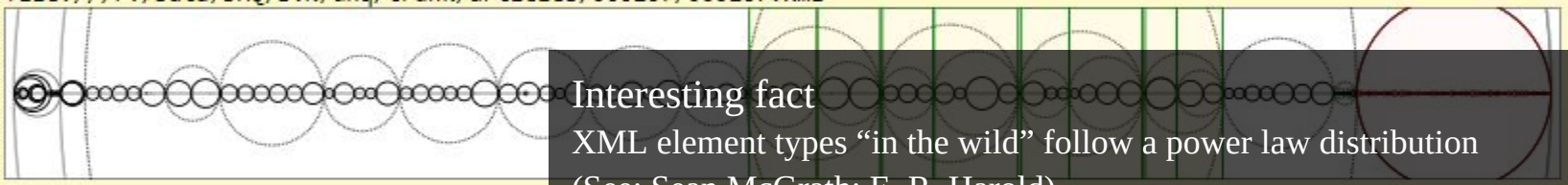
Information access in the art history domain: Evaluating a federated search engine for Rembrandt research [Verberne, Boves and van den Bosch]

file:///F:/Da



The App-Maker Model: An Embodied Expansion of Mobile Cyberinfrastructure [Oppegaard and Rabby]

file:///F:/Data/DHQ/SVN/dhq/trunk/articles/000267/000267.xml



Interesting fact

XML element types “in the wild” follow a power law distribution

(See: Sean McGrath; E. R. Harold)

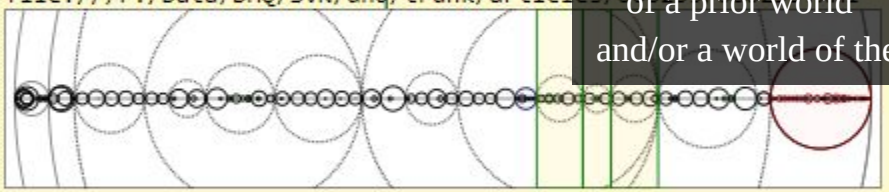
This is interesting if/as we take XML tags to be descriptive

of a prior world

and/or a world of their own

Exploring Citation Networks to Study Intertextuality in Classics [Romanello]

file:///F:/Data/DHQ/SVN/dhq/trunk/articles/000255/000255.xml



TaDiRAH: a Case Study in Pragmatic Classification [Borek, Dombrowski, Perkins and Schöch]

file:///F:/Data/DHQ/SVN/dhq/trunk/article

Disclaimer: this is not about Information Theory

Entropy = Information?

Entropy “measures the state description we don't have yet” (paraphrasing Shannon)

In this view, information is what is *not* known to us (yet), not what is ...

Information exists only in transmission (and it's not always what you think it is)

Setting aside complexity theory ...

... let's talk about *parsing* ...

Sound and Digital Humanities: reflecting on a DHSI course [Barber]

file:///F:/Data/DHQ/SVN/dhq/trunk/articles/000239/000239

<soCalled>Plain text</soCalled>

Example of plain text

Example of "plain text"

Example of *plain text*

Example of “plain text”

“plain text”

Example of <q>plain text</q>

Example of plain text

```
{\rtf1\ansi\deff3\adeflang1025
{\fonttbl{\f0\froman\fprq2\fcharset0 Times New Roman;}{\f1\froman\fprq2\fcharset2 Symbol;}{\f2\fswiss
\fprq2\fcharset0 Arial;}{\f3\froman\fprq2\fcharset0 Liberation Serif{\*\falt Times New Roman};}
{\f4\fswiss\fprq2\fcharset0 Liberation Sans{\*\falt Arial};}{\f5\fnil\fprq2\fcharset0 WenQuanYi Micro
 Hei;}{\f6\fnil\fprq2\fcharset0 Lohit Devanagari;}{\f7\fnil\fprq0\fcharset128 Lohit Devanagari;}}
{\colortbl;\red0\green0\blue0;\red0\green0\blue255;\red0\green255\blue255;\red0\green255\blue0;
\red255\green0\blue255;\red255\green0\blue0;\red255\green255\blue0;\red255\green255\blue255;
\red0\green0\blue128;\red0\green128\blue128;\red0\green128\blue0;\red128\green0\blue128;
\red128\green0\blue0;\red128\green128\blue0;\red128\green128\blue128;\red192\green192\blue192;}
{\stylesheet{\s0\snext0\widctlpar\hyphpar0\cf0\kerning1\dbch\af5\langfe2052\dbch
\af6\afs24\alang1081\loch\f3\fs24\lang1033 Normal;}
{\s15\sbasedon0\snext16\sb240\sa120\keepn\dbch\af5\dbch\af6\afs28\loch\f4\fs28 Heading;}
{\s16\sbasedon0\snext16\sl288\slmult1\sb0\sa140 Text Body;}
{\s17\sbasedon16\snext17\sl288\slmult1\sb0\sa140\dbch\af7 List;}
{\s18\sbasedon0\snext18\sb120\sa120\noline\i\dbch\af7\afs24\ai\fs24 Caption;}
{\s19\sbasedon0\snext19\noline\dbch\af7 Index;}
}{\*\generator LibreOffice/5.1.4.2$Linux_X86_64 LibreOffice_project/10m0$Build-2}
{\info{\creatim\yr2018\mo7\dy23\hr13\min55}{\revtim\yr2018\mo7\dy23\hr13\min56}{\printim
\yr0\mo0\dy0\hr0\min0}}\deftab709
\viewscale100
{\*\pgdsctbl
{\pgdsc0\pgdscuse451\pgwsxn12240\pghsxn15840\marglsxn1134\margrsxn1134\margtsxn1134\margbsxn1134\pgdscnx
 Default Style;}}
\formshade\paperh15840\paperw12240\margl1134\margr1134\margt1134\margb1134\sectd\sbknone
\sectunlocked1\pgndec
\pgwsxn12240\pghsxn15840\marglsxn1134\margrsxn1134\margtsxn1134\margbsxn1134\ftnbj
\ftnstart1\ftnrstcont\ftnnar\aenddoc\aftnrstcont\aftnstart1\aftnnrlc
{\*\ftnsep\chftnsep}\pgndec\pard\plain \s0\widctlpar\hyphpar0\cf0\kerning1\dbch\af5\langfe2052\dbch
\af6\afs24\alang1081\loch\f3\fs24\lang1033{\rtlch \ltrch\loch
Example of }{\i\ai\rtlch \ltrch\loch
plain text}
\par }
```

```
<p>Example of <em>plain text</em></p>
```

```
{
   "p": {
      "em": "plain text",
      "__text": "Example of"
   }
}
```

```
<p>Example of <em>plain text</em> but with <em>even more fun mixed content</em></p>
```

```
{
   "p": {
      "em": [
         "plain text",
         "even more fun mixed content"
      ],
      "__text": "Example of \n but with"
   }
}
```

<w:document xmlns:wpc="http://schemas.microsoft.com/office/word/2010/wordprocessingCanvas" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:o="urn:schemas-microsoft-com:office:office" xmlns:r="http://schemas.openxmlformats.org/officeDocument/2006/relationships" xmlns:m="http://schemas.openxmlformats.org/officeDocument/2006/math" xmlns:v="urn:schemas-microsoft-com:vml" xmlns:wp14="http://schemas.microsoft.com/office/word/2010/wordprocessingDrawing" xmlns:wp="http://schemas.openxmlformats.org/drawingml/2006/wordprocessingDrawing" xmlns:w10="urn:schemas-microsoft-com:office:word" xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main" xmlns:w14="http://schemas.microsoft.com/office/word/2010/wordml" xmlns:wpg="http://schemas.microsoft.com/office/word/2010/wordprocessingGroup" xmlns:wpi="http://schemas.microsoft.com/office/word/2010/wordprocessingInk" xmlns:wne="http://schemas.microsoft.com/office/word/2006/wordml" xmlns:wps="http://schemas.microsoft.com/office/word/2010/wordprocessingShape" mc:Ignorable="w14 wp14"><w:body><w:p w:rsidR="00866D2D" w:rsidRDefault="00DA293E"><w:r><w:t xml:space="preserve">Example </w:t></w:r><w:proofErr w:type="gramStart"/><w:r><w:t xml:space="preserve">of </w:t></w:r><w:bookmarkStart w:id="0" w:name="_GoBack"/><w:r w:rsidRPr="00DA293E"><w:rPr><w:i/></w:rPr><w:t>plain</w:t></w:r><w:proofErr w:type="gramEnd"/><w:r w:rsidRPr="00DA293E"><w:rPr><w:i/></w:rPr><w:t xml:space="preserve"> text</w:t></w:r><w:bookmarkEnd w:id="0"/></w:p><w:sectPr w:rsidR="00866D2D"><w:pgSz w:w="12240" w:h="15840"/><w:pgMar w:top="1440" w:right="1440" w:bottom="1440" w:left="1440" w:header="720" w:footer="720" w:gutter="0"/><w:cols w:space="720"/><w:docGrid w:linePitch="360"/></w:sectPr></w:body></w:document>

Semantic Slope depiction, 2012 (Brown University workshop)

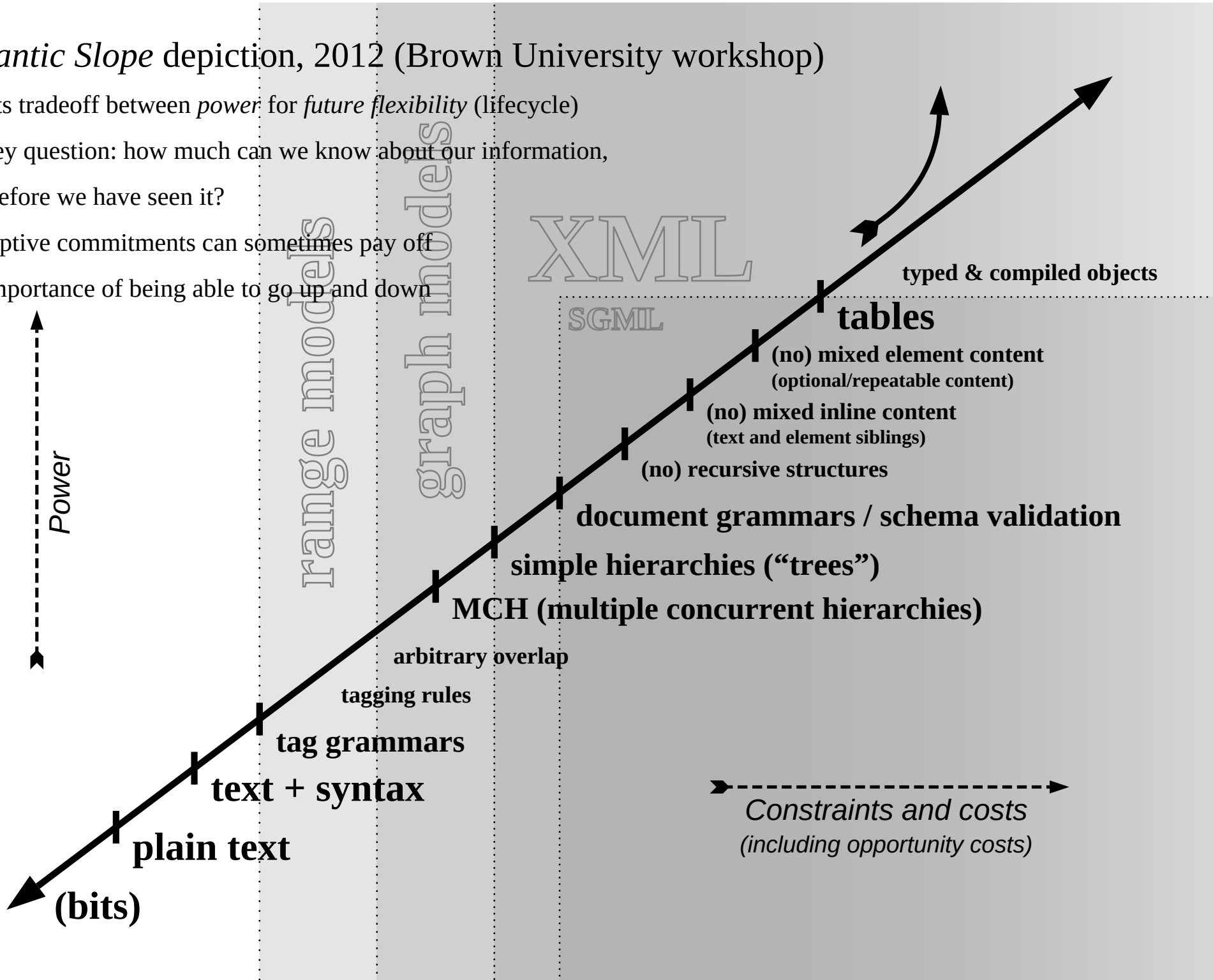
Depicts tradeoff between *power* for *future flexibility* (lifecycle)

The key question: how much can we know about our information,

before we have seen it?

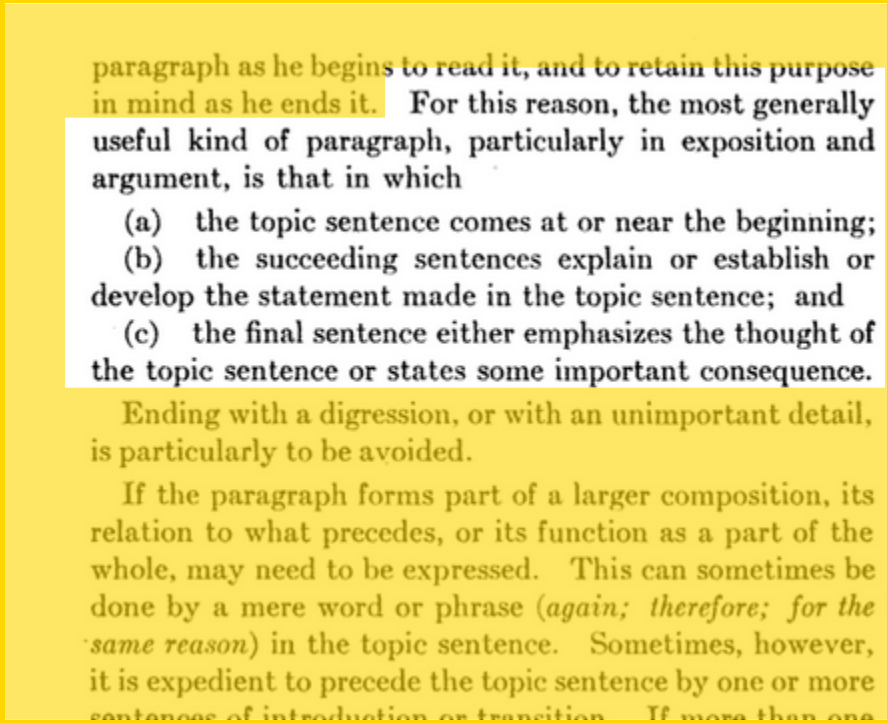
Preemptive commitments can sometimes pay off

The importance of being able to go up and down



What do we mean when we say “markup”?

or “encoding” or “information”



paragraph as he begins to read it, and to retain this purpose in mind as he ends it. For this reason, the most generally useful kind of paragraph, particularly in exposition and argument, is that in which

(a) the topic sentence comes at or near the beginning;

(b) the succeeding sentences explain or establish or develop the statement made in the topic sentence; and

(c) the final sentence either emphasizes the thought of the topic sentence or states some important consequence.

Ending with a digression, or with an unimportant detail, is particularly to be avoided.

If the paragraph forms part of a larger composition, its relation to what precedes, or its function as a part of the whole, may need to be expressed. This can sometimes be done by a mere word or phrase (*again; therefore; for the same reason*) in the topic sentence. Sometimes, however, it is expedient to precede the topic sentence by one or more sentences of introduction or transition. If more than one

From Strunk and White, *Elements of Style*, 1908 ed., p 16

(Screenshot from Google Books **`id=TadLAQAAMAAJ&pg=PA16`**)

10. As a rule, begin each paragraph with a topic sentence;
end it in conformity with the beginning.

```
<?xml version="1.0" encoding="UTF-8"?>
<s>For this reason,
   <np>the
      <adj>most generally useful</adj>
      kind <gen>of paragraph</gen></np>,
   <intj aff="emph" int="spec">particularly
      <pos>in <choice>exposition and argument,</choice></pos>
      </intj>
   <praed>
      <v>is</v>
      <np>that
         <pos>in which
            <list>
               <clause>
                  <np>the topic sentence</np>
                  <praed>
                     <v>comes</v>
                     <pos>
                        <choice>at or near</choice>
                        the beginning</pos>
                     </praed>;</clause>
               <clause>
                  <np>the succeeding sentences</np>
                  <praed>
                     <choice>
                        <v>explain</v>
                        or
                        <v>establish</v>
                        or
                        <v>develop</v>
                        </choice>
                     <np>the statement made
                        <pos>in the
                           <np>topic sentence</np>
                           </pos>
                        </np>
                     </praed>;</clause>
               and
               <clause>
                  <np>the final sentence</np>
                  <choice>either
                     <praed>
                        <v>emphasizes</v>
                        <np>the thought
                           <gen>of
                              <np>the topic sentence</np>
                           </gen>
                        </np>
                     </praed>
                     or
                     <praed>
                        <v>states</v>
                        <np>some important consequence</np>
                     </praed>
                  </choice>
               </clause>
            </list>
         </pos>
      </np>
   </praed>
.</s>
```

Only two choices?

```
<?xml version="1.0" encoding="UTF-8"?>
<s>For this reason,
   <np>the
      <adj>most generally useful</adj>
      kind <gen>of paragraph</gen></np>,
   <intj aff="emph" int="spec">particularly
      <pos>in <choice>exposition and argument,</choice></pos>
      </intj>
   <praed>
      <v>is</v>
      <np>that
         <pos>in which
            <list>
               <clause>
                  <np>the topic sentence</np>
                  <praed>
                     <v>comes</v>
                     <pos>
                        <choice>at or near</choice>
                        the beginning</pos>
                     </praed>;</clause>
               <clause>
                  <np>the succeeding sentences</np>
                  <praed>
                     <choice>
                        <v>explain</v>
                        or
                        <v>establish</v>
                        or
                        <v>develop</v>
                        </choice>
                     <np>the statement made
                        <pos>in the
                           <np>topic sentence</np>
                           </pos>
                        </np>
                     </praed>;</clause>
               and
               <clause>
                  <np>the final sentence</np>
                  <choice>either
                     <praed>
                        <v>emphasizes</v>
                        <np>the thought
                           <gen>of
                              <np>the topic sentence</np>
                           </gen>
                        </np>
                     </praed>
                     or
                     <praed>
                        <v>states</v>
                        <np>some important consequence</np>
                     </praed>
                  </choice>
               </clause>
            </list>
         </pos>
      </np>
   </praed>
.</s>
```

utilities-online.info (Domenico Briganti) provides a “straight up” cast
(XML to JSON)
shown here with slight adjustments
http://www.utilities-online.info/xmltojson/#.W1d6jmdMKIM

```
{
  "s": {
     "#text": [
        "For this reason,\n",
        ",\n ",
        "\n."
     ],
     "np": {
        "#text": [
           "the\n ",
           "\n kind "
        ],
        "adj": "most generally useful",
        "gen": "of paragraph"
     },
     "intj": {
        "-aff": "emph",
        "-int": "spec",
        "#text": "particularly\n ",
        "pos": {
           "#text": "in ",
           "choice": "exposition and argument,"
        }
     },
     "praed": {
        "v": "is",
        "clause": {
           "#text": "that\n ",
           "pos": {
              "#text": "in which\n ",
              "list": {
                 "clause": [
                    {
                       "np": "the topic sentence",
                       "praed": {
                          "v": "comes",
                          "pos": {
                             "choice": "at or near",
                             "#text": "\n the beginning"
                          }
                       },
                       "#text": ";"
                    },
                    {
                       "np": "the succeeding sentences",
                       "praed": {
                          "choice": {
                             "v": [
                                "explain",
                                "establish",
                                "develop"
                             ],
                             "#text": [
                                "\n or\n ",
                                "\n or\n "
                             ]
```

What a mess

What can we say about all this?

Cut to *dissolve into detail* ...

Not just about tradeoffs in tech choices

If language really reduced to substitutions and transformations, we would have no problem

Prospective and retrospective (descriptive/emulative) systems would already align

We could proceed by observing the regularity, then describing it

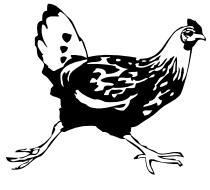
However, life is not (always, much) like that and language is not always language

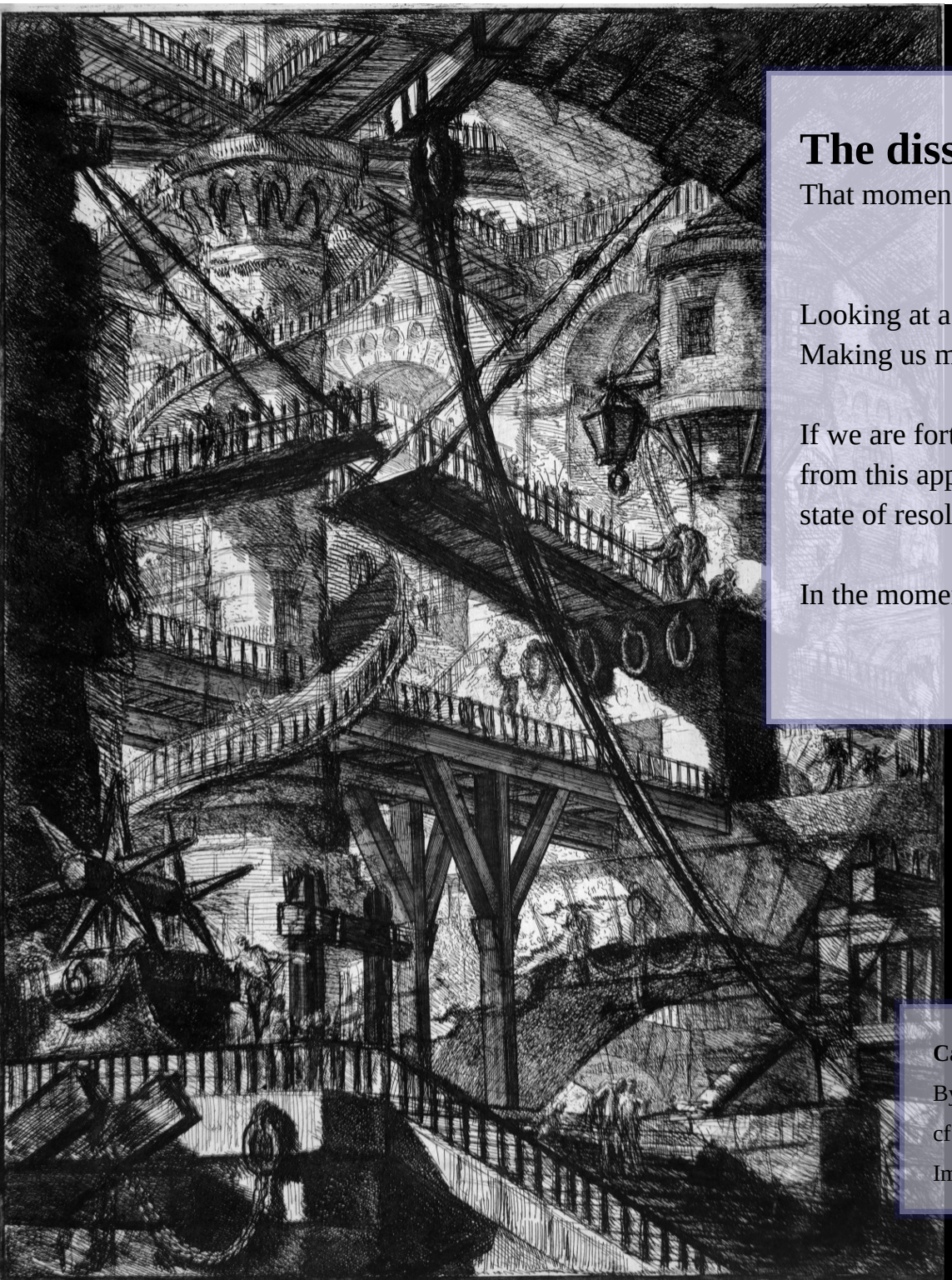
> (If dinosaurs could always fly, there would be no mystery how birds ever learned)

Instead, we have the *semantic surge* of human language and culture

> — poetry, administration, finance, law, science, journalism, technology —

Somehow, we manage to make enough sense out of the mishmash is everywhere!





The dissolve into detail

That moment of discovery

... things are more complicated ...

Looking at a new system
Making us more or less *anxious* or *apprehensive*

If we are fortunate, we are able to proceed quickly
from this apprehension, to another (more stable)
state of resolution.

In the moment we must ask:

Is a new tech only a series of confusing dead ends?
Or is it an opportunity?

Carceri [Prisons] **folder 7**

By Giovanni Battista Piranesi, 1745

cf Wikimedia Commons pgFR58dxJPdbfw (Public Domain)

Image from a print in the Dresden Art Center

Robot Armies

Clashing by night (and day too)

We like Markdown because it's so tasty!
We like JSON because it's easy to digest!
We like YAML because there's no mess!
We like XML because it's nutritious!

None of these do justice to what we know

The syntax matters less

What matters more

- The aptness of the vocabulary
- The fitness of the model
- (Almost impossible to provide top down)

Only then —

- The stability of the stack
- Other externalities

Where and when is technical fit the driving force in adoption?

Advancement happens in fits and starts: it's “lumpy”

Somehow, we manage to make enough sense out of the mishmash is everywhere!

Nor does it seem transformational linguistics is entirely wrong — the truth is somewhere *between*

... XML / HTML / Markdown / JSON / YAML ...

I am actually okay with all of it!

Latent Question

In modeling, what are we able to live without?

When is it okay not to know something, to defer...?

How are we acquiring this information, anyway, and what is it for?

> Shouldn't we worry about that first?

If the fool would persist in his folly he would become wise.

William Blake, *Proverbs of Hell* (1793)

Semantic Stairway 2018

Depicts tradeoff between *power* for *future flexibility* (lifecycle)

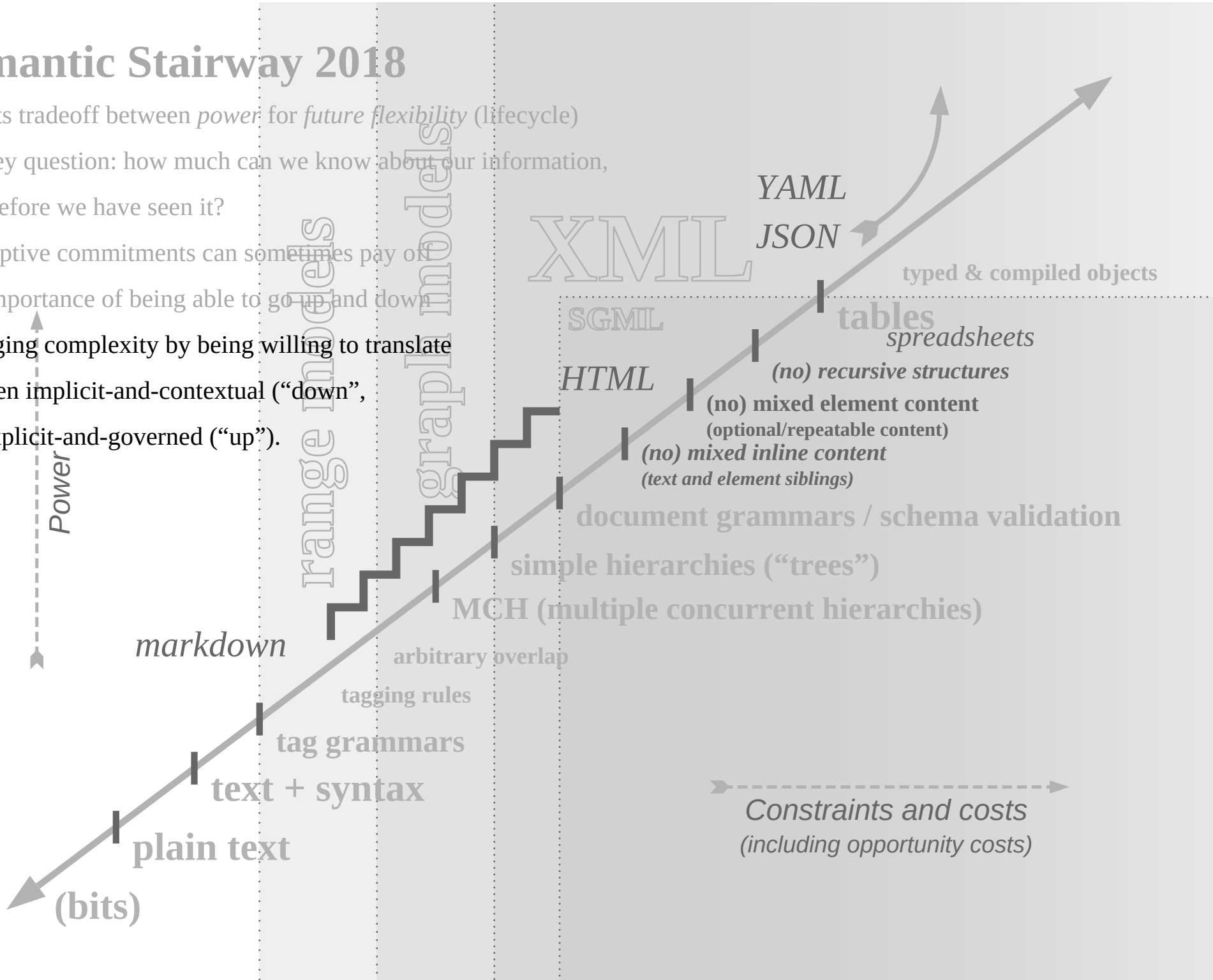
The key question: how much can we know about our information,

before we have seen it?

Preemptive commitments can sometimes pay off

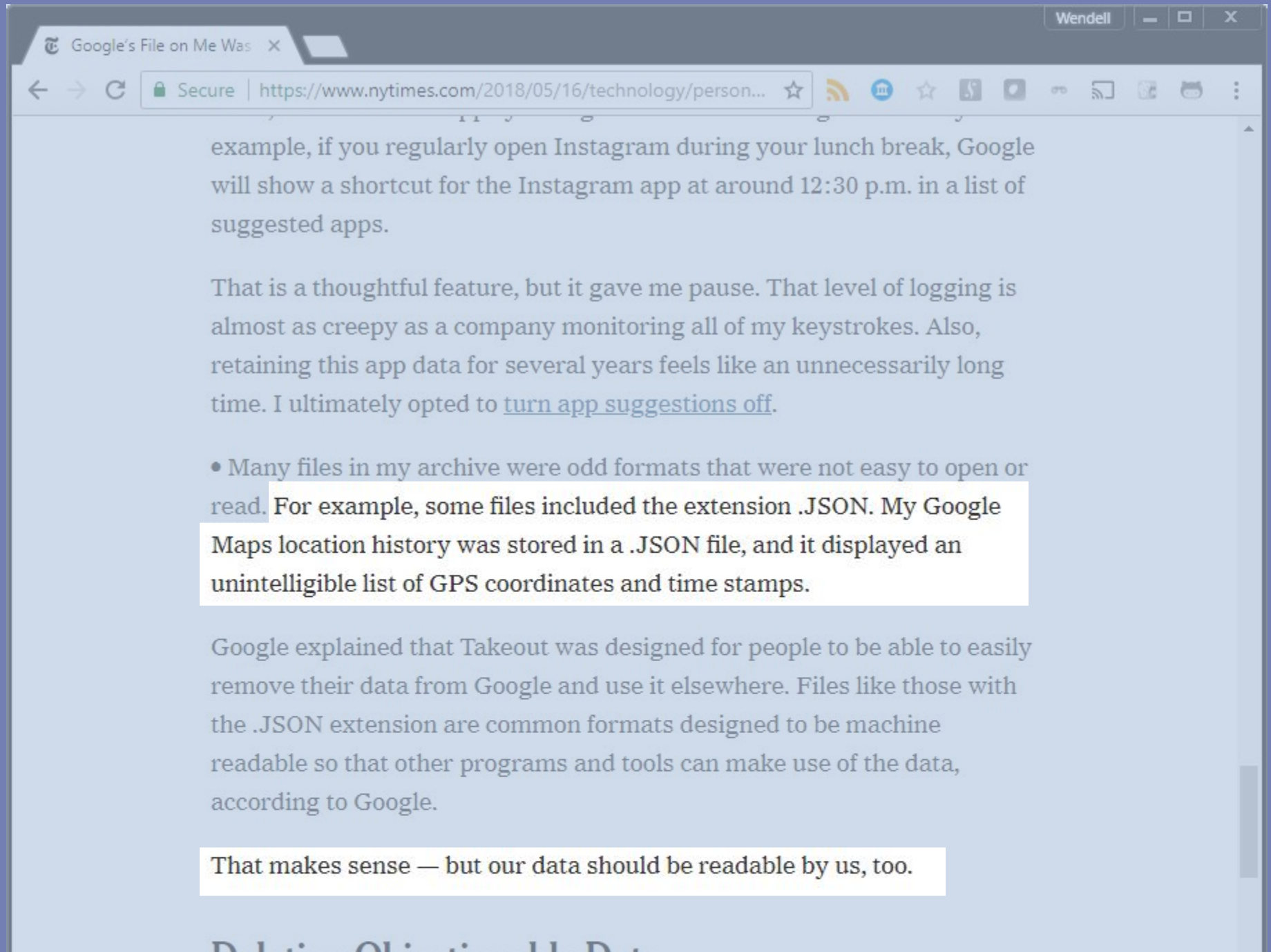
The importance of being able to go up and down

Managing complexity by being willing to translate between implicit-and-contextual (“down”, and explicit-and-governed (“up”).



Wanting our code to be legible

Screenshot from the online edition of *The New York Times*, May 16 2018



JSONishness

- More legible (yes, well)
- Easier to parse (also debatable, dissolving into detail)
- Lighter weight (for certain kinds of data, perhaps)
- ... *Advantages we get only when we can make some concessions up front* ...
- Monster JSON is just as ugly as Monster XML.

JSON is good once your data has been collected and organized

Once it has *taken shape*, virtuous cycles can start

Meanwhile, what's the matter with order

Order matters ... especially when “*it can be in any order*” (order “*doesn't matter*”): B. Tommie Usdin, Balisage

Translation: wherever “*it can be in any order*” ... order is unconstrained in the model (“*it doesn't matter*”) ... it may carry information in the instance (*it matters*)

```
<p>It was getting hot. Strangely, the kettle whistled.</p>
<p>I woke up with a start. What a dream.</p>
```

Encoding affordances

As a markup technology XML supports `(a | b)*` content models

Not excluding `(b | i | #PCDATA)*`

Structured data systems and object-oriented systems usually fail at this

... the “*soupiness*” of “*documentary data*” ...

The solution could be to focus on complementary strengths

Not “*or*” but “*and*”?

Expecting XML and JSON to survive and sometimes rub up against each other.

- This may mean tolerating a degree of parallel evolution.
- It also means being able to convert data where necessary

The good news - XDM/XSLT/XQuery have proven fully capable

New XDM **map** and **array** objects offer clean structural analog to JSON

Also (of course!) we have an XML tag notation for JSON syntax

New specifications (XPath 3.1) provide functional support in standard libraries

- XML developers can produce JSON as an XML->XML conversion, commodity serializer does the rest

Mappings from any XML to an optimal or even adequate JSON may not be trivial ...

... offer a mapping, however, and XSLT/XQuery can do the work

(In other words the non-trivial part is the specification not the implementation.)

We will develop a set of methods and techniques, possibly libraries

Oh: also, we must be ready to consume JSON as well

(Which is also not so hard if the data is any good)

In other words, we must offer XML technology as an enabling technology

… While this is (also) about markup technologies not just XML

Fortunately, this also a well understood problem space!

LaFontaine, Lee, Robie, Rennau, Holman, Cagle, MarkLogic, BaseX just to name a few.

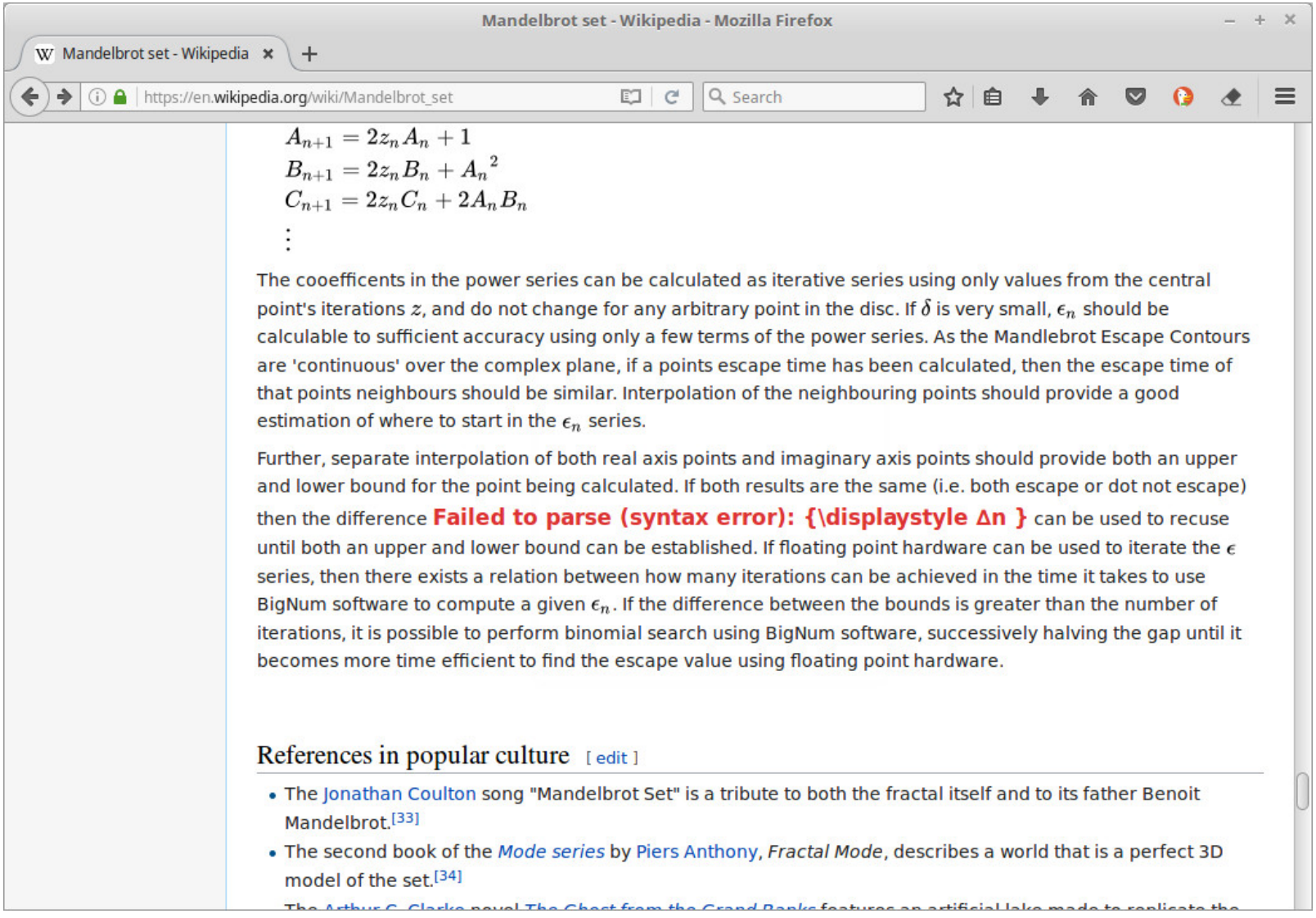
XML	JSON
Element/attribute (tree) structure is more or less abstracted from data, enabling leverage/ layering.	Presumably, data offers tight bindings to runtime object structures.
How high are you on the learning curve?	Comes for free if you're tackling Javascript (who isn't?)
Super-flexible at multiple levels of scale.	At its best when embedded in/with other processes/specs
Can be produced in a number of different ways (even from uncontrolled sources), even by hand.	Normally produced only by machined methods. But these include forms interfaces. So there is a niche.

Use XML as a back end and for interchange, while using JSON for application bindings (for applications that want it)?

XML	JSON
Elements and attributes.	Objects with properties of various data types.
Can capture almost anything gracefully including *"soupy"* documents. *"Graceful is as graceful does"*: adequacy is determined in the application.	Graceful as long as the object design itself is graceful.
XPath offers powerful addressing even over unknown, disparate and generalized data sets. When structure is not known ahead of time, XPath can interrogate and cope.	Addressing is supposed to be free (in Javascript or other host) but costs of analysis of unknown data can be high. This means that ease of addressing requires a fairly simple, rational and clear structure, known ahead of time and not liable to abuse or *"creativity"*.
Grouping is supported in a multitude of ways in XML, which is indeed a way of making groups of like and unlike data objects. Because grouping in XML is so easy and fluid, XML is tolerant of mixed-scale environments where the *"unit of interest"* is defined by context. (E.g.: paragraph, section, article, issue, journal, repository)	There is no concept of *"group"*, only object property hierarchy (since properties can be objects). Some structural grouping can be provided via arrays; or implicit semantic grouping can be done via normalization/flattening with property value assignment. In ordinary architectures, however, JSON objects are designed to work as discrete entities, perhaps grouped or aggregated in an application.
As just noted, XML accommodates various levels of scale in both data size and complexity.	Data sets are typically small in the instance; where not, someone else does the lifting. Since like XML, JSON is composable, in theory it can address scaling requirements as well as XML. In practice, JSON developers are not as tolerant of large and deep datasets.
External constraint sets (schemas) can be strict or loose, facilitating design, project scoping, incremental development, open development, and interchange.	External schemas? (2018 update: Oh, cool!)
Definitions of tagging and naming can be wired down, or shared out to the application - specifications can be and are *layered* in capable systems	All names have to be wired down or at least that has been the assumption in the past. But if we have schemas, how far behind can (functional) transformations be, or other metalanguages?

XML is tried and true in documentary workflows especially "high touch content-driven"; in structured data exchange XML's record is less compelling.	JSON has never really been tried with documentary data in any sustained way TMK. However, documentary data applications commonly have many structured data applications to go along with them.
Hand edited frequently, plus also not.	Don't ever want to edit JSON by hand, I wouldn't think.
[Entry costs of XML: comments redacted] Sensitive dependence on initial conditions!	If you are already fronting a web-based application, JSON exposure of any well-controlled data set is easy and free: entry costs are low low low when your language has native support for JSON. If you are not already fronting an application or your platform of choice has no special support for JSON, you have more range of platform choice (and other ideas become thinkable) and frequently even better serialization formats..
Whether it is easy to make your XML into JSON, depends entirely on whether and how the data *"fits"*. It is frequently feasible (even easy) to aquire JSON using "pulls" from XML; but it can be difficult to map entire XML datasets into JSON *"equivalents"*.	Generally it's pretty easy to produce nice XML from JSON.

Conclusion: if you know you are going to have to produce JSON, promise indexes and reports before you promise full text -- which is difficult to express in JSON. Even if you have already captured it cleanly using XML.



Screenshot from Wikipedia page on the Mandelbrot Set, July 2019

Markdownishness

Why (we think) we like markdown

```
## Why (we think) we like markdown

 - Nicely avoids all these problems by simply handing them to HTML
   - Or whatever abstract syntax is in back
 - Hence, markdown offers no solution / leverage over modeling per se
 - But a powerful tool in our toolkit

Funny thing: most markdown (in wide use) has no grammar to speak of

 - Its validity against any formal model, that is, is *mediated*
   - via the HTML (or other) markup to which it maps
   - and its grammar/s and rules (effectively supervening any local rules)
 - Meanwhile, its expressiveness is constrained by how cleanly the syntax maps

The practical consequence of this is that markdown hits a "complexity wall" in the kinds of information it can represent,
 especially when it comes to internal organization. In the real world, chunks of markdown may be organized among
 themselves, but they have only loose (commonly implicit) internal structure if any at all; indeed it can scarcely handle
 more than "p soup", i.e. html `p` elements with chunks of other stuff thrown in.

Since most of this kind of stuff is validated (only) "in the application", it makes no practical difference whether a
 formal grammar is respected; indeed it could be an issue when formal grammars (or indeed any specs) constrain against
 desired features.

### Thought experiment: how about cheating and not bothering to parse it anyway?

Instead, just cast it over to your favorite XML and try parsing that

This shows that what matters is the exception handling in any case

**What should happen when the parse succeeds?**

**What should happen when the parse fails?**
```

These problems are *fractal* insofar as we can zoom out and see the same question arise, at another functional/strategic level.

What is true of the parse, is also true of the workflow.

What we need

Adoption is a self-fulfilling prophecy.

At the same time, context is everything.

In an environment of many syntaxes and models,

What do we need from our tools?

- Flexibility
- Bridges

Easier up and down the stairway

This means tools, parsers, utilities

Tolerance for *“foreign formats”*

More innovations to bridge the gaps

“Dynamic markdown”?

Metaschemas?

A Generic Spreadsheet Language?

Dynamic markdown

No particular reason a markdown syntax must be hard-wired?

(This strategy is an alternative to rendering it to a target such as HTML and then transforming it.)

Could an abstract specification of a syntax-to-tag mapper, do as well as a grammar?

(Whether such a mapper would actually be good enough for a sufficient proportion of inputs, is the question.)

Map markdown as declared in the spec, to element structures declared in the spec.

```
## My note

.note Here's my note in my own personal markdown. It even has a (link)[http://example.com].

### Philosophical subsection

No markup, no semantics.

.special When we have processing ... we have semantics. Things look different when transformations are easy.

.quote Those who have art and science, have religion. Those who have no art and no science: they can have religion.
 [Goethe]

.special What's the difference between "markup", and not-markup?

(Glib answer: it's always markup. Actual but also unhelpful answer: it's relative to the situation.)
```

Steven Pemberton's “*invisible XML*” as applied to markdown ...

Metaschemas

All mature tagging languages end up with metaschemas: name your favorite tag set and there is likely to be some technology behind its schema maintenance - if not a metaschema by that name, then something functionally equivalent

Their benefits, for schema maintenance and generalization, are well understood

How about metaschemas specifically built to bridge the gaps between formats?

Architectural Forms showed a way to do this with markup vocabularies

A metaschema can produce a family of schemas with transformations (mapping) baked in

We can go even further if (for example) our metaschema enforces constraint sets for both XML and JSON at the same time.

Generic Spreadsheet Toolkit

Millions of people will never write *“code”*

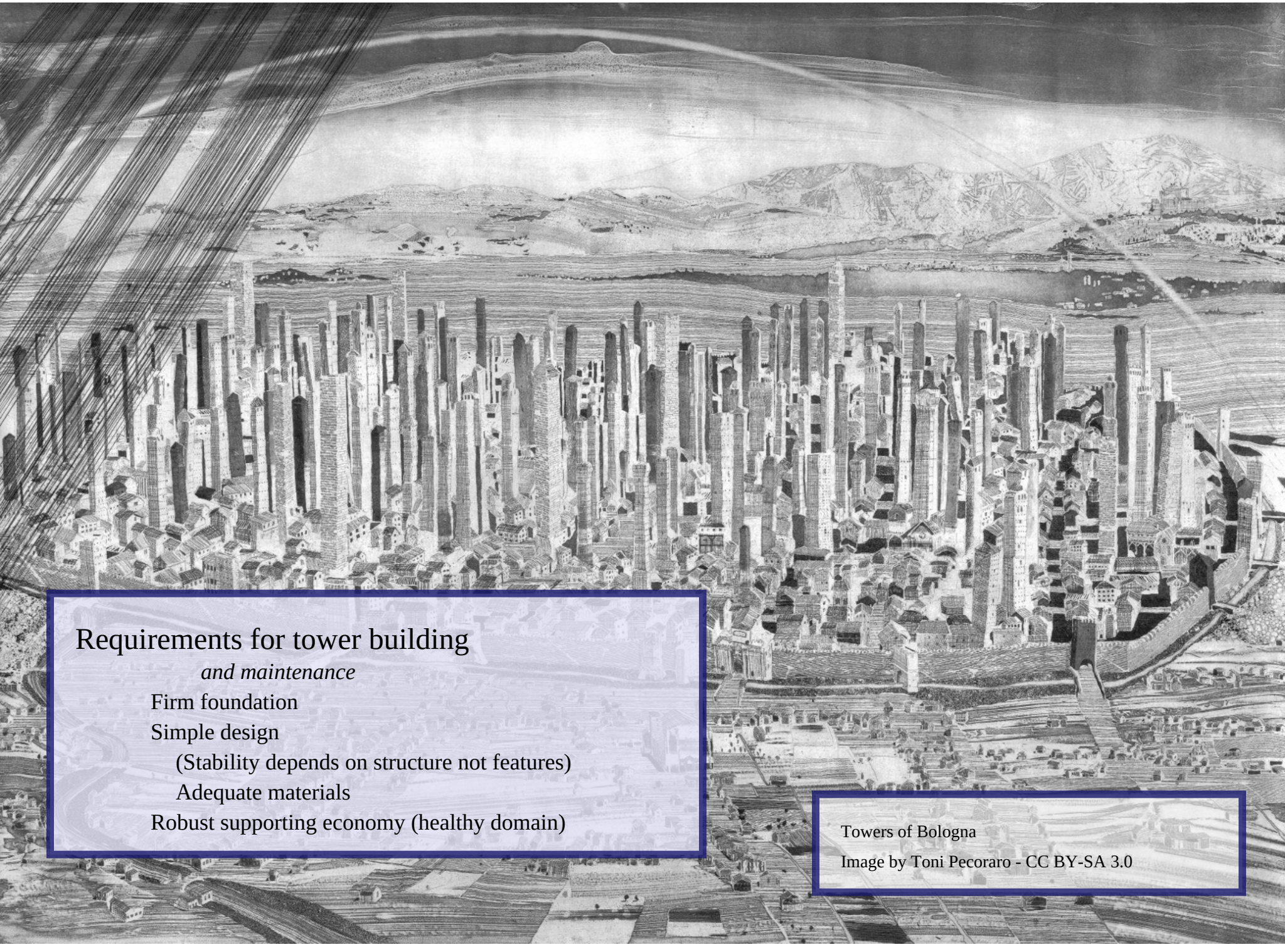
But *“program”* routinely in the form of spreadsheets

How about a generic spreadsheet language? (We could call it GSML)

It would be one step away from an Abstract User Interface such as XForms

Could serve as a back end in / for (spreadsheet-ish) applications

And an interchange *“pivot”* format to richer semantics



Requirements for tower building

and maintenance

Firm foundation

Simple design

(Stability depends on structure not features)

Adequate materials

Robust supporting economy (healthy domain)

Towers of Bologna

Bologna 2013

(population 380000)

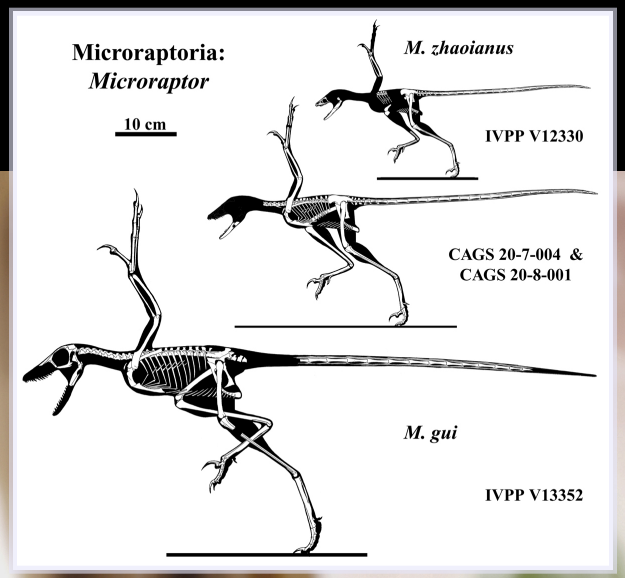
Photo by the author

Thoughts?

The towers have been taken down — but tower builders become cathedral builders

What does the long view look like: markup tech from an evolutionary perspective?

Which came first, the genotype or the phenotype?



Microraptor Skeletons, by Qi Long (Wikimedia Commons)

Four-winged thieves (microraptors)

By Durbed, CC BY-SA 3.0 Wikimedia Commons