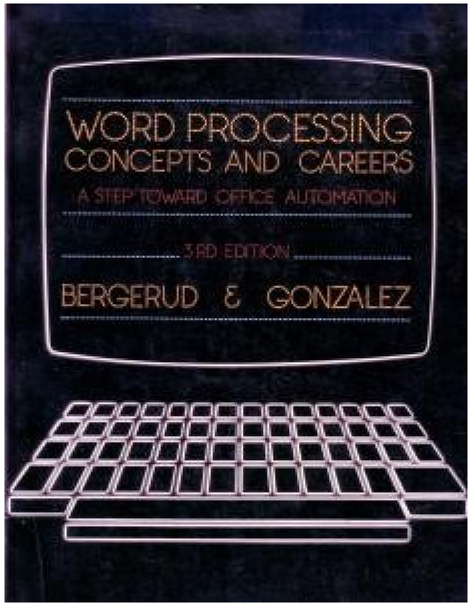
Generative Typesetting

John Haltiwanger

Libre Graphics Meeting 2011

Generative typesetting is a name for a relatively old-fashioned process that has been reimplemented using modern tools.

Utilizes _visually semantic_ markup and a translation layer without any preconceptions.



WORD PROCESSING
CONCEPTS AND CAREERS
A STEP TOWARD OFFICE AUTOMATION
3RD EDITION
BERGERUD & GONZALEZ

And then there was TeX...

1978

Electronic typesetting systems and cold-press publishing machines have largely replaced the older methods of printing, but they had not necessarily succeeded in producing the same level of quality.

Meanwhile, semantics...

1980

Brian Reid develops a system called Scribe which concerns itself with representing the semantic structure of a document. This allows the computer to understand when a title is a title, a section is a section, etc.

```
@Heading(Genesis)

@Begin(Quotation)

   And then there were quotes, of
indeterminate  style...

@End(Quotation)
```

LaTeX = TeX * Scribe

Early 1980s

Leslie Lamport develops LaTeX at SRI International. Over the course of the decade it becomes a de facto standard for journal publication in mathematic and scientific fields.

```
\title{Semantics what?!}
\begin{document}
   Set in style...
   Right?
\end{document}
```

The Reign of the <>

From Scribe we see a steady evolution of semantic markup into SGML then to HTML and finally XML.

But as the systems get more general and computable, they get less and less easy for a human to parse.

Enter Markdown

2000s

John Gruber and Aaron Swartz develop Markdown, one of the first pre-formats. Enabled by the glory of Perl Regular Expressions.

Based on the conventions that arose over decades of e-mail, Markdown implements a _visually semantic_ markup.

Followed by many other options.

Flexible Options

A new approach is necessary.

Why?

Because typographic workflows are infinitely variable. This is one reason why generative approaches like LaTeX fade away when the output is not an article with standardized, preset style.

Mind the Subtext

A mutable translation layer.

Bolt your own front-ends and backends (*styles* and *effects*).

Input files not only gain access to multiple output formats, their agency now includes an infinite variety of styles.

Subtext Architecture

Styles and Effects configurations are Lua tables (or can be mapped to tables from a more generic syntax, perhaps looking something like CSS).

Context (ConTeXt) as the primary PDF backend.

But your effects can be anything.

Preliminary Syntax

```
style{“header”,
      start = “#”,
      stop = “#”,
      inline = “no”}
effect{“header”,
       format = “context”,
       setup = “\setuphead[subject][style={\switchtobodyfont[16pt,sc]”,
       start = '\subject{',
       stop = '}' }
effect{“header”,
       format = “html”,
       setup = “<style type='text/css'>h1.smallcaps{ font-variant: small-
caps; }</style>”,
       start = “<h1 class='smallcaps'>”,
       stop = “</h1>” }
```

Advantages

Customizability means future-proofing and workflow-consolidation.

Separation of *style* and *effect* means that documents can be re-set without changing their structure.

No programming (outside of Context, HTML, XML, etc).

Further

Email : john.haltiwanger@gmail.com

Website : http://drippingdigital.com/blog

Twitter : @jceasless

Identi.ca : @ab5tract

IRC : ab5tract@freenode