
Editor's Concrete Syntax (ECS): a Profile of SGML for Editors

*Topologi Technical Note.
August 13, 2002*

Rick Jelliffe

Editing Concrete Syntax (ECS)

This draft paper formalizes the lexical rules the **Editor's Concrete Syntax** (ECS, pronounced ECS not X) for a family of SGML markup languages which have been in widespread use with colouring text-editors, and combine some of the attractive qualities of SGML and XML. This syntax is suitable for use

- by SGML users to take advantage of colouring text editors,
- by XML users to allow minimized data entry, which can then be normalized to XML, and
- for SGML users, who are transitioning to XML, or who have mixed SGML/XML systems.

1 SGML and XML

SGML's superiority to XML in minimising the number of characters needed for markup is well-known, and has proved popular in HTML. Indeed, XML's goals noted *terseness is of minimal importance*. SGML's terse tagging is important for industrial markup because

- it reduces the number of keystrokes required to tag a document,
- it decreases the conceptual load on the operator, can think in terms of linear tags when convenient rather than nesting ranges,
- it increases the likelihood that a missing tag can be recovered from satisfactorily, without user intervention, and
- it makes better use of valuable screen real-estate.

SGML has a cost for this superiority: the grammar used for parsing a document is determined in part by the SGML abstract syntax, in part by the features enabled and delimiters specified in the SGML declaration, in part by the DTD (such as whether an element has RCDATA or mixed content, and which short-reference map is in scope for a particular context), and even, potentially, by information in the document itself (USEMAP declarations in the instance). This makes simple implementation of SGML text editors quite difficult.

Terse markup's attraction may be obvious to people who use *Wikiwiki*

XML takes a different tack: it introduces well-formedness (WF), so that parsing a document is only determined by the rules of XML, except for the specific issue of whether an attribute value contains multiple tokens or character data. Even though XML WF requires checking that elements match (i.e. some kind of tree or stack machine), parsing it only requires a state machine.

Paradoxically, SGML is frequently edited using either vanilla text editors, or programmer's editors which feature some kind of simple coloring. The colouring is most conveniently implemented using a state machine. That way the effect of changes to text during moment-by-moment editing do not continually require that the stack be maintained. So SGML is frequently edited using stack machine-based tools, which by rights should not be powerful enough, while XML is frequently edited using tree-based tools, which by rights it does not need.

So, in practise, there has been in widespread use a fairly unrecognised markup notation, which sits in between SGML and XML. To give a rough idea of what it is, imagine HTML's lexical rules, *sans* any HTML-specific features (such as that the SCRIPT element does not allow entity references, being declared in an SGML DTD as CDATA.)

This paper formalized this syntax, for use in colouring editors and SGML/XML tools.

2 Productions

Here are basic productions for ECS. (Note: these are ambiguous for simplicity.)

```

file ::= encoding-header? ( tag | reference | data ) *
encoding-header ::= "<?sgml" s+ "encoding=" literal s+ "?"? ">"
tag ::= start-tag | end-tag | doctype | comment | pi | section
start-tag ::= "<" NMSTRT [ ^SEPCHAR && ^DELIM ] ** ( s+ attribute ) * s* "/"? ">"
end-tag ::= "</" NMSTRT [ ^SEPCHAR && ^DELIM ] ** s* ">"
comment ::= "<!--" .* "-->"
pi ::= "<?" .* "?"? ">"
section ::= "<![ " s* "CDATA" s* "[ " .* "]" >"
attribute ::= NMSTRT [ ^SEPCHAR && ^DELIM ] * s* "=" s* literal
literal ::= ("\" .* (reference .*)* "\\") | ("\' .* (reference .*)* '\")
doctype ::= "<!DOCTYPE s+ ( NMSTRT | \"#IMPLIED\" ) .* literal s* (literal s+)? ( \"[\" internals \"] \" )? ">"
internals ::= ( s+ | comment | pi | declaration | section ) *
declarations ::= element_dec | entity_dec | attlist_dec | notation_dec | pref
element_dec ::= "<!ELEMENT" s+ NMSTRT .* ">"
entity_dec ::= "<!ENTITY" s+ NMSTRT .* (literal .*)* ">"
attlist_dec ::= "<!ATTLIST" s+ NMSTRT .* (literal .*)* ">"
notation_dec ::= "<!NOTATION" s+ NMSTRT .* (literal .*)* ">"
reference ::= "&" ( "#" "x"? ) NMSTRT [ ^SEPCHAR && ^DELIM ] * ";"?
pref ::= "%" NMSTRT [ ^SEPCHAR && ^DELIM ] * ";"?

```

Where NMSTRT is a name start character, SEPCHAR is the whitespace, and DELIM is any kind of delimiter. `[^SEPCHAR && ^DELIM]*` means any character that is not whitespace or a delimiter; this is a very simple way to tokenize a document for a coloring editor, and real implementations might use the naming rules to determine the token length.

Note that there are several structures which may be put in an XML document which these productions do not reveal. The purpose of ECS is describe a minimal syntax which editors can use as a base for value-added implementations.

3 ECS in terms of XML

ECS can be described as a series of relaxation of XML 1.0's WF rules:

- there is no requirement that an element starts at the top: content could start, and the same tool can edit the document entity and a subentity;
- start-tags and end-tags do not need to match,
- a start-tag with no end-tag may be acting as a start-tag or an empty-tag,
- references do not need a terminating ";" if they are followed by some non-name character,
- the delimiters "<<" and "&" do not need to be converted to entity references if followed by a name start character,
- a processing instruction may be closed by a ">"
- the following short-forms are allowed: "<>", "</>", "<!>"
- attribute values which are single tokens do not require LIT or LITA delimiters (i.e. " or ')
- slight differences in DTDs are allowed, such as the minimization indicators ("- and "o"), the different keywords for entities and elements, and that system identifiers are not required
- Use `<?sgml version="1.0" encoding="???"?>` instead of `<?xml ...?>`
- name checking is only performed for the first 255 code points; apart from that, anything goes.
- all ISO entity sets are predefined.

Validation

A SlackXML document is **not** *well-formed* XML. The transformation to make it well-formed is SGML DTD dependent. A SlackXML document may be validated using an SGML DTD. If there is no DTD but some other schema, in the absense of other information a parser will treat it as *amply-tagged*, and imply omitted end-tag according to the rules of WebSGML: if the file ends, if a currently open element ends, or if an start-tag for the same element type appears. To allow this, element types used in SlackXML must not be *immediately recursive* (may not contain as a child an element of the same type.)

4 ECS in terms of SGML

XML is described in terms of SGML in <http://www.w3.org/TR/NOTE-sgml-xml-971215>

ECS can be described in terms of SGML, as a superset of RCS (the default Reference Concrete Syntax):

- an element should end in the same entity it begins (this allows easier validation of entities) as in XML,
- the SGML declaration is fixed, with Reference Concrete Syntax delimiters used (e.g. the default delimiter for SGML), large name lengths (as in XML), Unicode (as in XML), OMITTAG and SHORTTAG minimization allowed (as in SGML default, but different to XML), and SHORTREFs allowed (but with the proviso that “matching” shortrefs should end in the same entity as their starting partner),
- only CDATA sections are allowed in the document instance, not using parameter entities,
- RCDATA and CDATA content types are not allowed,
- marked sections in the prolog are ignored by this syntax, except for ones with “CDATA” type,
- declarations are not allowed outside the internal or external subset of the prolog,
- character encoding can be specified using a PI, borrowing from XML,
- empty-tags are allowed but not required

In the terminology of WebSGML (see <http://www.y12.doe.gov/sgml/sc34/document/0029.htm>), an ECS document:

- is an *integrally-stored document instance*,
- should be, if it is to be processed with no DTD, *amply-tagged*.

Validation

An ECS document may be validated using an SGML DTD. If there is no DTD but some other schema, in the absence of other information a parser will treat it as *amply-tagged*, and imply omitted end-tag according to the rules of WebSGML: if the file ends, if a currently open element ends, or if a start-tag for the same element type appears. To allow this, an element type may not be *immediately recursive* (may not contain as a child an element of the same type.)

SGML System Declaration

Here is an SGML *System Declaration* for ECS. (A system declaration describes the features of an SGML system; you compare it with your document’s SGML declaration to see if the system can accept your document.) It is designed to be as compatible as possible with typical SGML declarations in use.

<!SYNTAX -- SGML Declaration for Editor’s Concrete Syntax --

 ISO 8879:1986 (WWW)

CHARSET

 BASESET

 *ISO Registration Number 176//CHARSET
 ISO/IEC 10646-1:1993 UCS-4 with implementation
 level 3//ESC 2/5 2/15 4/6*

 DESCSET

 0 9 UNUSED
 9 2 9
 11 2 UNUSED
 13 1 13
 14 18 UNUSED
 32 95 32
 127 1 UNUSED
 128 32 UNUSED
 160 55136 160

55296 2048 UNUSED -- surrogates --
57344 8190 57344
65534 2 UNUSED -- FFFE and FFFF --
65536 1048576 65536

CAPACITY NONE

SCOPE DOCUMENT

SYNTAX

SHUNCHAR CONTROLS 0 1 2 3 4 5 6 7 8 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 127

BASESET "ISO Registration Number 176//CHARSET
ISO/IEC 10646-1:1993 UCS-4 with implementation
level 3//ESC 2/5 2/15 4/6"

DESCSET
0 1114112 0

FUNCTION

RE 13
RS 10
SPACE 32
TAB SEPCCHAR 9
NEL SEPCCHAR 133

NAMING

LCNMSTRT ""
UCNMSTRT ""
NAMESTRT
58 95 192-214 216-246 248-55295

LCNMCHAR ""
UCNMCHAR ""
NAMECHAR
45-46 183
NAMECASE
GENERAL NO
ENTITY NO

DELIM

GENERAL SGMLREF
HCRO "&#x" -- 38 is the number for ampersand --
SHORTREF NONE

NAMES
SGMLREF

QUANTITY NONE

ENTITIES -- plus all ISO standard entities! --
"amp" 38
"lt" 60
"gt" 62
"quot" 34
"apos" 39

FEATURES

MINIMIZE

DATATAG NO
OMITTAG YES
RANK NO
SHORTTAG
STARTTAG
EMPTY YES
UNCLOSED NO

NETENABL IMMEDNET
ENDTAG
EMPTY YES
UNCLOSED NO
ATTRIB
DEFAULT YES
OMITNAME NO
VALUEY YES
EMPTYNRM YES
IMPLYDEF
ATTLIST YES
DOCTYPE YES
ELEMENT ANYOTHER
ENTITY YES
NOTATION YES
LINK
SIMPLE NO
IMPLICIT NO
EXPLICIT NO
OTHER
CONCUR NO
SUBDOC NO
FORMAL NO
URN NO
KEEPSRE YES
VALIDITY NOASSERT
ENTITIES
REF ANY
INTEGRAL YES
APPINFO NONE
SEEALSO "SlackXML Requirements"
>

SGML Declaration

Here is an notional SGML Declaration for ECS. It is designed to be as compatible as possible with typical SGML declarations in use.

<ISGML -- SGML Declaration for Editor's Concrete Syntax --
"ISO 8879:1986 (WWW)"

CHARSET
BASESET
"ISO Registration Number 176//CHARSET
ISO/IEC 10646-1:1993 UCS-4 with implementation
level 3//ESC 2/5 2/15 4/6"
DESCSET
0 9 UNUSED
9 2 9
11 2 UNUSED
13 1 13
14 18 UNUSED
32 95 32
127 1 UNUSED
128 32 UNUSED
160 55136 160
55296 2048 UNUSED -- surrogates --

57344 8190 57344
65534 2 UNUSED -- FFFE and FFFF --
65536 1048576 65536

CAPACITY NONE

SCOPE DOCUMENT

SYNTAX

SHUNCHAR CONTROLS 0 1 2 3 4 5 6 7 8 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 127

BASESET "ISO Registration Number 176//CHARSET
ISO/IEC 10646-1:1993 UCS-4 with implementation
level 3//ESC 2/5 2/15 4/6"

DESCSET
0 1114112 0

FUNCTION

RE 13
RS 10
SPACE 32
TAB SEPCCHAR 9
NEL SEPCCHAR 133

NAMING

LCNMSTRT ""
UCNMSTRT ""
NAMESTRT
58 95 192-214 216-246 248-55295

LCNMCHAR ""
UCNMCHAR ""
NAMECHAR
45-46 183

NAMECASE
GENERAL NO
ENTITY NO

DELIM

GENERAL SGMLREF
HCRO "&#x" -- 38 is the number for ampersand --
SHORTREF NONE

NAMES
SGMLREF

QUANTITY NONE

ENTITIES -- plus all ISO standard entities! --
"amp" 38
"lt" 60
"gt" 62
"quot" 34
"apos" 39

FEATURES

MINIMIZE

DATATAG NO
OMITTAG YES
RANK NO
SHORTTAG
STARTTAG
EMPTY YES
UNCLOSED NO
NETENABL IMMEDNET

ENDTAG
EMPTY YES
UNCLOSED NO
ATTRIB
DEFAULT YES
OMITNAME NO
VALUEY YES
EMPTYNRM YES
IMPLYDEF
ATTLIST YES
DOCTYPE YES
ELEMENT ANYOTHER
ENTITY YES
NOTATION YES
LINK
SIMPLE NO
IMPLICIT NO
EXPLICIT NO
OTHER
CONCUR NO
SUBDOC NO
FORMAL NO
URN NO
KEEPSRE YES
VALIDITY NOASSERT
ENTITIES
REF ANY
INTEGRAL YES
APPINFO NONE
SEEALSO "ECS"
>
VALIDATE
GENERAL NO
MODEL NO
EXCLUDE NO
CAPACITY NO
NONSGML YES

SDIF PACK NO

A shortform of this could be used:

```
<!SGML ECS PUBLIC "+//IDN topologi.com//SD Editor's Concrete Syntax//EN">
```