SWI-Prolog SGML/XML parser

Jan Wielemaker
SWI,
University of Amsterdam
The Netherlands
E-mail: jan@swi.psy.uva.nl

May 11, 2000

Abstract

1 Introduction

Markup languages have recently regained popularity for two reasons. One is document exchange, which is largely based on HTML, an instance of SGML and the other is for data-exchange between programs, which is often based on XML, which can be considered simpli ed and rationalised version of SGML.

James Clark's SP parser is a exible SGML and XML parser. Unfortunately it has some drawbacks. It is very big, not very fast, cannot work under event-driven input and is generally

```
[],
[ element(head,
          [],
          [ element(title,
                    [],
                    [ 'Demo'
                    ])
          ]),
  element(body,
          [],
          [ '\n\n',
```

```
load_html_file(File, Term) :-
        dtd(html, DTD),
        load_structure(File, Term,
                       [ dtd(DTD),
```

Attributes declaring namespaces (`xmlns:`*`ns`*`=`*`url`* are reported as if `xmlns` is not a de ned resource.

In- many cases-0.9nat mes-0aes -ces

```
load_dtd(DTD, DtdFile) :-
        open_dtd(DTD, [], DtdOut),
        open(DtdFile, read, DtdIn),
        copy_stream_data(DtdIn, DtdOut),
        close(DtdIn),
        close(DtdOut).
```

open_dtd(+*DTD*, +*Options*, -*OutStream*)
: Open either a DTD as an output stream. The option-list is currently empty. See `load_dtd/2` for an example.

dtd(+*DocType*, -*DTD*)
: Find the DTD representing the indicated *doctype*. This predicate uses a cache of DTD

```
elements_in_xml_document(File, Elements) :-
        load_structure(File, _,
                       [ dialect(xml),
                         dtd(DTD)
                       ]),
        dtd_property(DTD, elements(Elements)),
        free_dtd(DTD).
```

3.6 Parsing Primitives

new_sgml_parser(*-Parser, +Options*)

a clean solution, especially on si9 T and medium-sized documents. It however is unsuitable for parsing really big documents. Such documents can only be handled with the call-back output interface realised by the `call`(*Event, Action*) option of `sgml_parse/2`. Event-driven

sgml_register_catalog