"mime" or "binhex", the former being the default value.

We conclude with two more remarks on DTDs. First, a DTD can be interpreted as an Extended Backus-Naur Form (EBNF). For example, the declaration

```
<!ELEMENT email (head,body)>
```

is equivalent to the rule

```
email ::= head body
```

which means that an email consists of a head followed by a body. And second, recursive definitions are possible in DTDs. For example,

```
<!ELEMENT bintree ((bintree root bintree)|emptytree)>
```

defines binary trees: a binary tree is the empty tree, or consists of a left subtree, a root, and a right subtree.

### A.2.2 XML Schema

XML Schema offers a significantly richer language for defining the structure of XML documents. One of its characteristics is that its syntax is based on XML itself. This design decision provides a significant improvement in readability, but more important, it also allows significant reuse of technology. It is no longer necessary to write separate parsers, editors, pretty printers, and so on to obtain a separate syntax, as was required for DTDs; any XML will do. An even more important improvement is the possibility of reusing and refining schemas. XML Schema allows one to define new types by extending or restricting already existing ones. In combination with an XML-based syntax, this feature allows one to build schemas from other schemas, thus reducing the workload. Finally, XML Schema provides a sophisticated set of data types that can be used in XML documents (DTDs were limited to strings only).

An XML schema is an element with an opening tag like