Introduction

This report present the data structures used to attach informations to some word of text, mainly to use in the hypertext Emacs interface.

```
import call from "ri.newgen"

import entity from "ri.newgen"

import loop from "ri.newgen"

import reference from "ri.newgen"
```

I need to store pointer object:

```
External void_star
```

1 Attachments

Are only a list of `attachment`s:

```
attachments = attachment*
```

2 Attachment

Here is what can be attached to a word:

```
attachment = attachee x begin:int x end:int
```

`begin` is the position of the attachment begin in the output file and `end` the position of its end.

The various objects that can be attached:

```
attachee = statement_line_number:int + persistent reference + persistent
call + persistent declaration:entity + type:string + persistent loop
+ persistent module_head:entity + complementary_sections:unit + complexities:unit
+ continuation_conditions:unit + cumulated_effects:unit + out_regions:unit
+ preconditions:unit + privatized_regions:unit + proper_effects:unit
+ proper_regions:unit + regions:unit + static_control:unit + transformers:unit
+ decoration:unit + comment:unit
```

Persistence is need because we do not want the RI to be broken when the attachments are freed.

3 The mapping used to attach various internal informations

Each word can have a list of attachments:

```
word_to_attachments = word_pointer:void_star->attachments
```

`word_pointer` is an `int` instead of a `string` since the address must be used instead of the string content. So it needs a cast...