Раздел. Правила разработки sc-моделей баз знаний согласно методике, основанной на редактировании исходных текстов
= [*

Правила набора исходных текстов в файлы .scs и .gwf.
Файлы .scsi и их подключение

Основная информация о фрагменте *базы знаний* хранится в .scs- и .gwf-файлах, а .scsi-файлы являются вспомогательными.

Файлы .scs содержат исходный текст фрагмента *базы знаний* на языке *SCs*, а .gwf-файлы - на языке *SCg*.

.scsi-файлы служат для того, чтобы задать содержимое некоторой *структуры*, описываемой в рамках основного .scs-файла.

Для подключения вспомогательных файлов .scsi в основной файл .scs следует использовать следующий синтаксис:

```
системный_идентификатор_структуры = [*^"file://путь_к_файлу.scsi"*];;
```

Например, необходимо сформировать исходный файл для следующего фрагмента *базы знаний*:

Раздел. Предметная область примеров
∈ *атомарный раздел*
=
[*
Предметная область примеров
∈ *предметная область*
*]

Тогда содержимое .scs-файла будет следующим:

```
section_subjdomain_examples
=> nrel_main_idtf:
    [Раздел. Предметная область примеров]
    (* <- lang_ru;; *);
    [Section. Subject domain of examples]
    (* <- lang_en;; *);
<- atomic_section;;
section_subjdomain_examples = [*^"file://section_subjdomain_examples.scsi"*];;
```

А содержимое файла .scsi будет иметь вид:

```
subject_domain_of_examples
=> nrel_main_idtf:
    [Предметная область примеров]
    (* <- lang_ru;; *);
    [Subject domain of examples]
    (* <- lang_en;; *);
<- subject_domain;;
```

При наборе исходных текстов необходимо использовать только системные идентификаторы!

Если при сборке *базы знаний* и последующем просмотре в системе набранного фрагмента какие-либо *sc-элементы* идентифицируются *системными идентификаторами*, а не основными для текущего языка диалога, необходимо тщательно проверить исходный текст набранного фрагмента *базы знаний* на наличие опечаток или неверно использованных *системных идентификаторов*.

Если разработчик фрагмента уверен, что в исходном тексте всё верно, однако *sc-элемент* по-прежнему идентифицирован некорректно, значит, в текущем состоянии *базы знаний* отсутствует описание данного *sc-элемента*, и необходимо описать хотя бы минимальные сведения о нём (указать тип *sc-элемента* и *основные идентификаторы*), при необходимости предварительно согласовав это с тем, кто поставил задачу.

При наборе исходных текстов в .scs-файлах необходимо указывать тип *sc-узлов*, соответствующих вновь вводимым *понятиям*:

- sc_node_not_relation для *классов*
- sc_node_norole_relation для *неролевых отношений*
- sc_node_role_relation для *ролевых отношений*

Указывать тип необходимо только для sc-узлов, обозначающих понятия, которые раньше не присутствовали в системе!

Если какое-либо *понятие* участвует в отношении *включение**, *разбиение** и др., но непосредственно не описывается в создаваемом фрагменте *базы знаний*, то указывать структурный тип для него повторно не требуется.

Исключением также являются названия *разделов*, *предметных областей* и др. *понятия*, являющиеся экземплярами уже описанных ранее *классов*.

Для *sc-элементов*, обозначающих экземпляры каких-либо *классов*, может дополнительно указываться их структурный тип:
- sc_node_not_binary_tuple для *небинарных связей*
- sc_node_struct для *структур*
- sc_node_abstract для *абстрактных сущностей*
- sc_node_material для *материальных сущностей*

Например:

Понятие	Запись в исходном тексте на языке SCs
знание	knowledge <- sc_node_not_relation; => nrel_main_idtf: [знание] (* <- lang_ru;; *);;
метазнание*	nrel_metaknowledge <- sc_node_norole_relation; => nrel_main_idtf: [метазнание*] (* <- lang_ru;; *);;
ключевой sc-элемент’	rrel_key_sc_element

	<- sc_node_role_relation; => nrel_main_idtf: [ключевой sc-элемент’] (* <- lang_ru;; *);;
Предметная область знаний	subject_domain_of_knowledges => nrel_main_idtf: [Предметная область знаний] (* <- lang_ru;; *);;
Раздел. Предметная область знаний	section_subjdomain_knowledges => nrel_main_idtf: [Раздел. Предметная область знаний] (* <- lang_ru;; *);;

Подключение внешних файлов

Если набираемый фрагмент базы знаний представляет собой только естественно-языковой текст, то такой фрагмент оформляется только в виде специальным образом размеченного .html-файла. Затем этот .html-файл подключается в соответствующий формируемому фрагменту базы знаний файл .scs.

Все естественно-языковые вставки при описании *sc-элементов* (пояснения, определения, примечания и др.) также оформляются в виде специальным образом размеченных .html-файлов.

Нередко описание какого-либо фрагмента *базы знаний* сопровождается графической иллюстрацией, в этом случае возникает необходимость подключить к исходному тексту изображение.

Файлы .html и файлы, содержащие изображения, к исходному .scs-файлу подключаются одинаково.

Для формирования знака файла в *sc-памяти* используется следующий синтаксис:

“file://путь_к_файлу”

Например, необходимо сформировать следующий фрагмент *базы знаний*:

пример понятия
∈ *ключевой sc-элемент’*:
o
∈ *пояснение*
=> *$трансляция sc-текста*:*
[***пример понятия*** - это ...]
∈ *Русский язык*

∈ *ключевой sc-элемент’*:
o
∈ *описание типичного экземпляра*
=> *$трансляция sc-текста*:*
[

]

На языке SCs в исходном тексте этот фрагмент имеет следующий вид:

```
example_concept
<- sc_node_not_relation;
=> nrel_main_idtf:
      [пример понятия]
      (* <- lang_ru;; *);
<- rrel_key_sc_element:
      ...
      (*
            <- explanation;;
            <= nrel_sc_text_translation:
            ...
            (*
                  -> rrel_example:
                  "file://content_html/explanation_for_example_concept.html"
                  (* <- lang_ru;; *);;
            *);;
      *);
<- rrel_key_sc_element:
      ...
      (*
            <- description_of_a_typical_instance;;
            <= nrel_sc_text_translation:
            ...
            (*
                  -> rrel_example:
                    "file://content_img/description_for_example_concept.png";;
            *);;
      *);;
```

Принципы разметки .html-файлов

.html-файлы размечаются обычными тегами, характерными для языка HTML, однако для того, чтобы иметь возможность использовать какие-либо упоминаемые в тексте элементы *базы знаний* в качестве аргументов для каких-либо команд, их необходимо выделять специальными парными тегами:

```
<sc_element sys_idtf  = "системный_идентификатор">
</sc_element>
```

Кроме этого, .html-файл не должен содержать тегов верхнего уровня типа <html> или <body>, а начинаться сразу с тегов <p>, и т. п.

Согласно общим правилам оформления естественно-языковых текстов в *sc-памяти, идентификаторы* таких *sc-элементов* выделяются в естественно-языковом фрагменте курсивом (жирным и нежирным).

Например, необходимо разметить следующий естественно-языковой фрагмент:

[*пример понятия* является искусственно созданным *понятием* для обеспечения понимания принципов разметки .html-файлов].

Соответствующий .html-файл будет иметь следующий вид:

 <sc_element sys_idtf = "example_concept">пример понятия</sc_element> является искусственно созданным <sc_element sys_idtf = "concept">понятием</sc_element> для обеспечения понимания принципов разметки .html-файлов.

Если при сборке *базы знаний* и последующем просмотре в системе набранного фрагмента какие-либо выделенные таким образом *понятия* в .html-вставках отмечены красным цветом, необходимо тщательно проверить размеченный .html-файл на наличие опечаток и неверно использованных *идентификаторов.*

Правила формирования системных идентификаторов. Правила именования файлов

Системные идентификаторы, как правило, формируются путём аккуратного прямого перевода русскоязычного *идентификатора* на английский язык.

Если *идентификатор* получается слишком длинным, допускается сокращение слов, пропуск предлогов, некоторых несмыслообразующих слов.

При формировании *системных идентификаторов sc-элементов* желательно учитывать их иерархию, например, понятия *раздел* и *раздел-документация* будут иметь *системные идентификаторы* section и section_documentation, хотя прямой перевод на английский второго *идентификатора* предполагает иной порядок слов (documentation section).

Имена файлов, как правило, формируются согласно тому, что в нём описано, однако имя файла желательно делать как можно короче, но его название должно отражать содержимое, например section_project_ims.scs

Правила форматирования .scs-файлов

Каждое новое описываемое понятие при наборе .scs-файла следует писать с новой строки. Таким образом, структура .scs-файла становится похожей на группу упорядоченных sc.n-статей.

Рекомендуется использовать сокращённую форму описания, то есть SCs-код более высоких расширений.

Рассмотрим вышеуказанную рекомендацию на примере описания следующего *понятия*:

знание
⊃ *раздел*

⊃ *предметная область*
⊃ *онтология*

Полная форма имеет вид:

```
knowledge => nrel_strict_inclusion: section;;
knowledge => nrel_strict_inclusion: subject_domain;;
knowledge => nrel_strict_inclusion: ontology;;
```

Сокращённая форма имеет вид:

```
knowledge
=> nrel_strict_inclusion:
      section;
=> nrel_strict_inclusion:
      subject_domain;
=> nrel_strict_inclusion:
      ontology;;
```

или

```
knowledge
=> nrel_strict_inclusion:
      section;
      subject_domain;
      ontology;;
```

Таким образом, сокращённая форма имеет преимущество над полной в том, что нет необходимости несколько раз указывать *системный идентификатор* описываемого *понятия*.

Следует обратить внимание на расстановку символов «;» и «;;» при использовании сокращённой формы записи.

Вложенность sc.s-предложений должна быть показана с помощью горизонтальной табуляции и расположения открывающей и закрывающей скобок на одном уровне.
*]