DDL Import Attributes

Role of IMPORTATION in Dictionaries

Importation of definitions between dictionary files:

- Allows specific discipline (domain) dictionaries to be built without the duplication of commonly-used definitions.
- Definitions can be easily shared across domains; the definitions of commonly-used data are not duplicated.
- Avoids definition redundancy within dictionaries by minimising the definition content of closely related items.
- Allows commonly-used definition data (such as default-value & enumeration state lists) to be placed in a single file.

The IMPORT attributes

- A definition IMPORT is specified using the attributes:

  _import.scope - import scope 'dic','cat','grp','itm','att','sta','val'

  _import.block - name of imported definition block

  _import.file - file containing imported block

  _import.if_dupl - action taken if definition duplicated

  _import.if_miss - action taken if definition missing

IMPORT scopes

The _import.scope attribute has the allowed states:

dic	import all definitions in the dictionary file
cat	import all definitions in a single category
grp	import all definitions in a group category (with children)
itm	import a single item definition
att	import template attributes to within a definition
sta	import enumeration state-list to within a definition
val	import enumeration default value-list to within a definition

Application of IMPORT Attributes

Importation attributes may be applied either in a

- » loop_ or a
- » list-string

>> The *loop_* list is expressed as:

```
loop_
    _import.scope
    _import.block
    _import.file
    _import.if_dupl
    _import.if_miss
```

>> The *list-string* attribute _import_list.id is expressed as:

[*.scope, *.block, *.file, *.if_dupl, *.if_miss]

IMPORT Conflict Protocols

- _import.if_dupl controls the action if duplicate definitions arise
- _import.if_miss controls the action if requested definition is missing
- _import.if_dupl and _import.if_miss attributes are *optional*
- _import.if_dupl action codes for treating duplicate definitions:
  - Ignore - ignore the import request
  - Replace - replace *existing* definition block with *requested* import definition block
  - Exit - exit with fatal error (default action)
- _import.if_miss action codes for treating missing definitions:
  - Ignore - ignore the import request
  - Exit - exit with fatal error (default action)

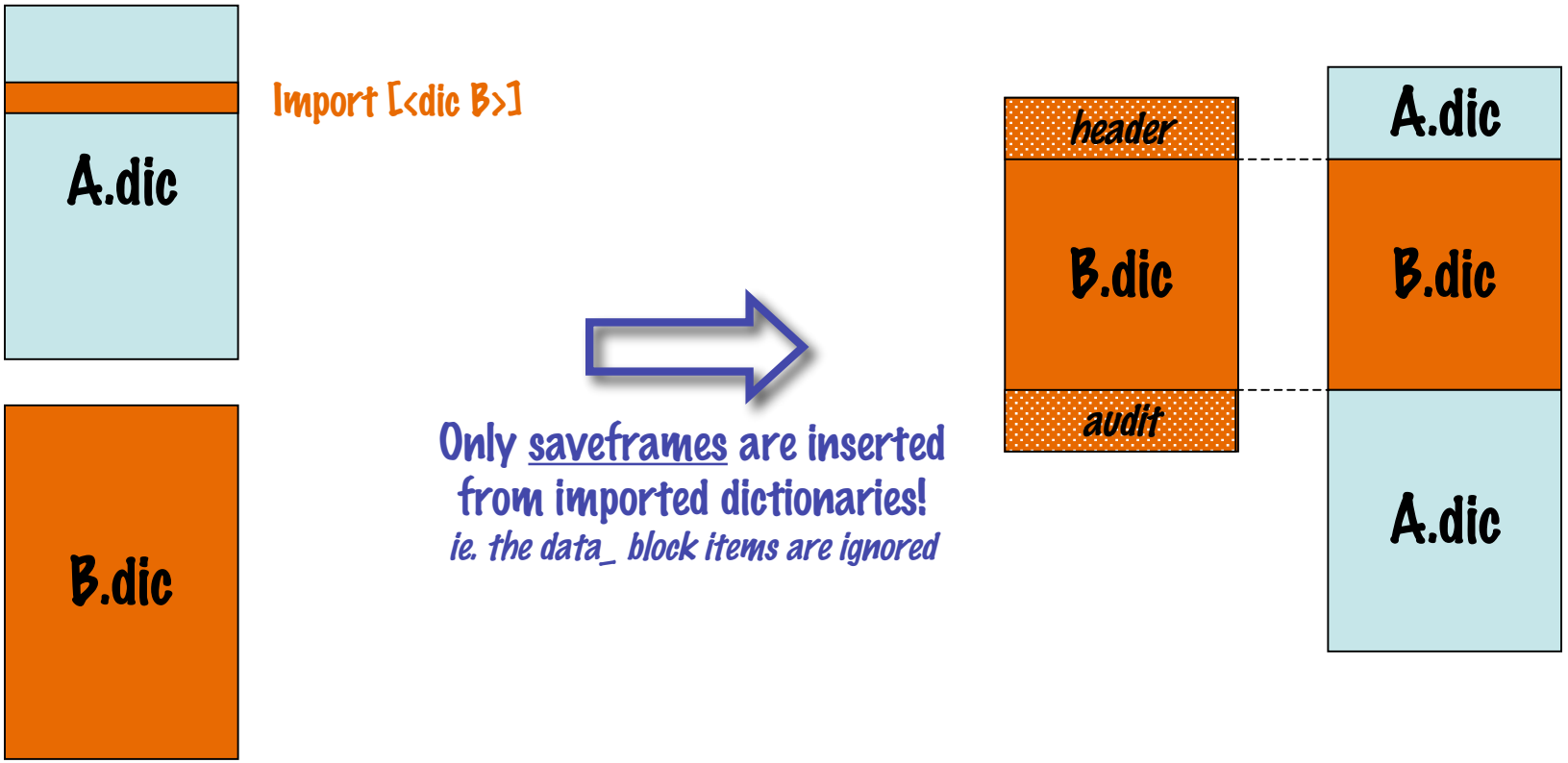
Typical IMPORT examples

```
loop_
    _import.scope
    _import.block
    _import.file
    _import.if_dupl
    _import.if_miss
      'dic'  'CORE_CRYSTAL'          'core_crystal.dic'      'exit'     'exit'
      'cat'  'ATOM_SITE'             'core_structure.dic'    'ignore'   'exit'
      'grp'  'CELL'                  'core_crystal.dic'      'replace'  'exit'
      'itm'  '_atom_site.multiplicity' 'core_structure.dic'  'replace'  'exit'

_import_list.id   ['att', 'Miller_index', 'com_att.dic', 'exit', 'exit' ]
_import_list.id   ['sta', 'colour_hue', 'com_val.dic', 'exit', 'ignore' ]
_import_list.id   ['val', 'Cromer_Mann_a1', 'com_val.dic', 'exit', 'exit' ]
```

Example import with scope = 'dic' #1

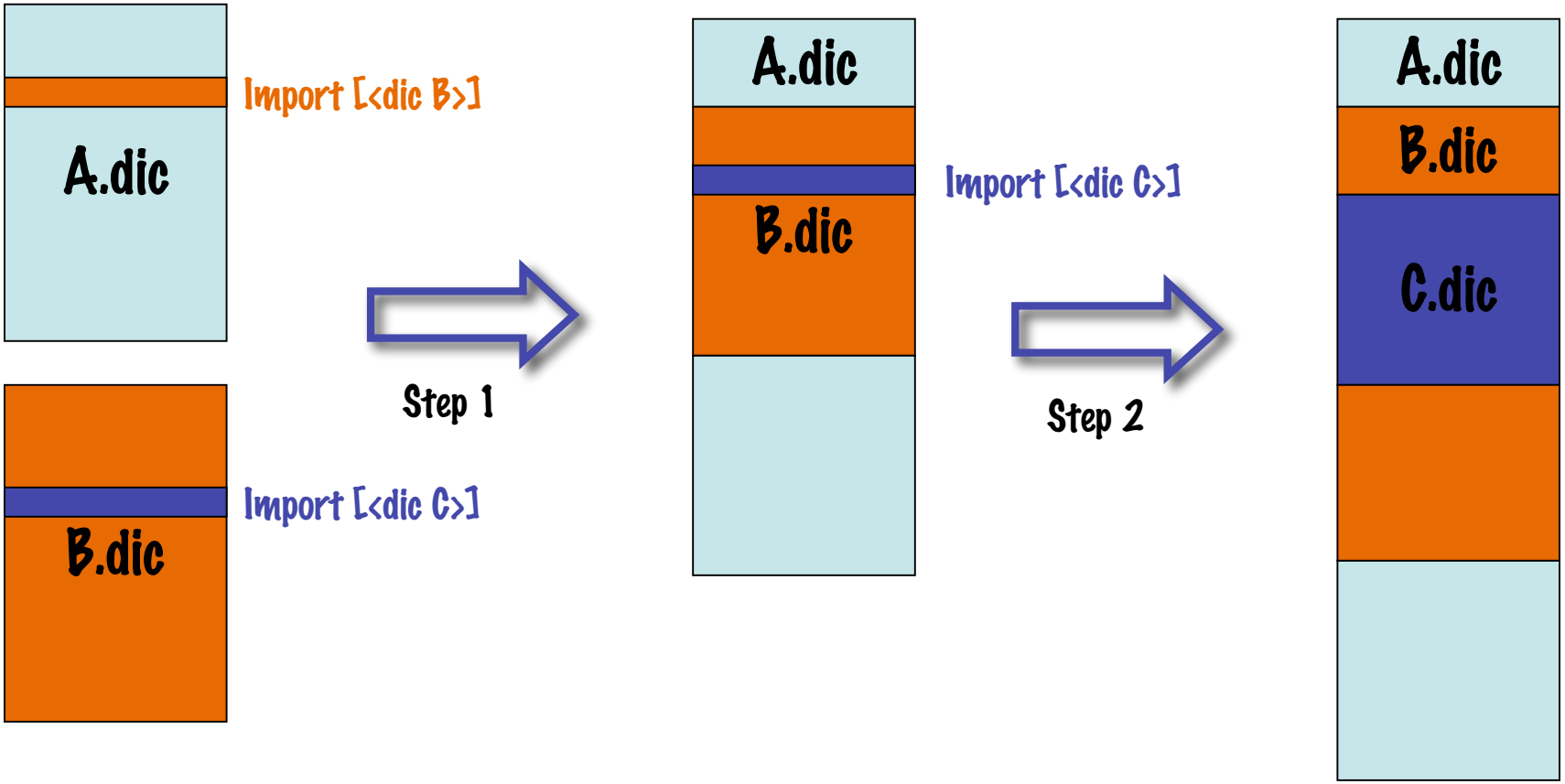
Simple example: dictionary A imports dictionary B



2007

Example import with scope = 'dic' #2

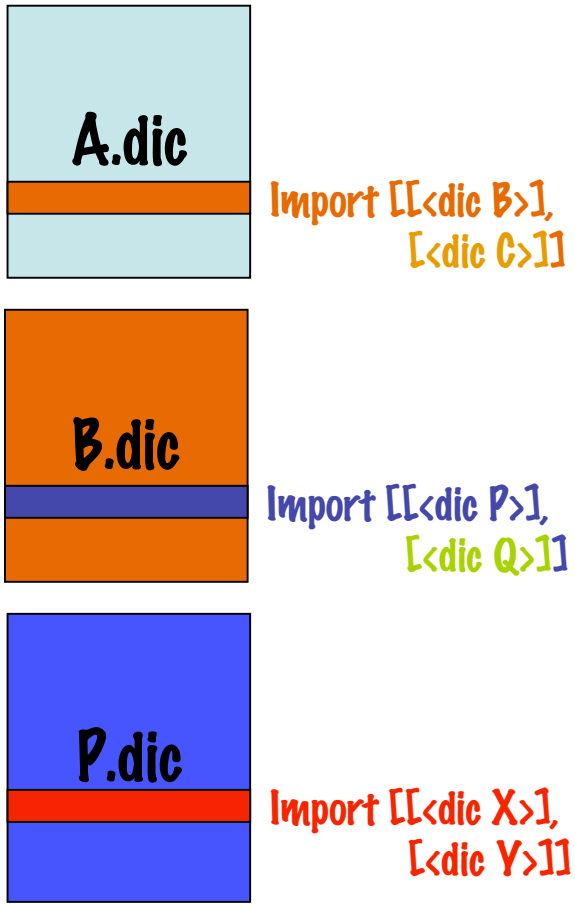
Nested example: *dic A imports dic B and dic B imports dic C*

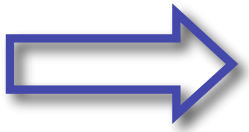


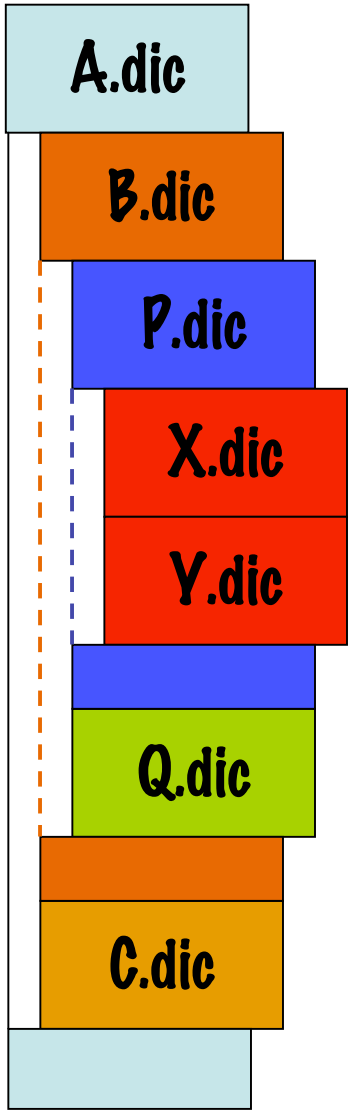
2007

Example import with scope = 'dic' #3

Complex example: *dic A imports nested dictionaries*



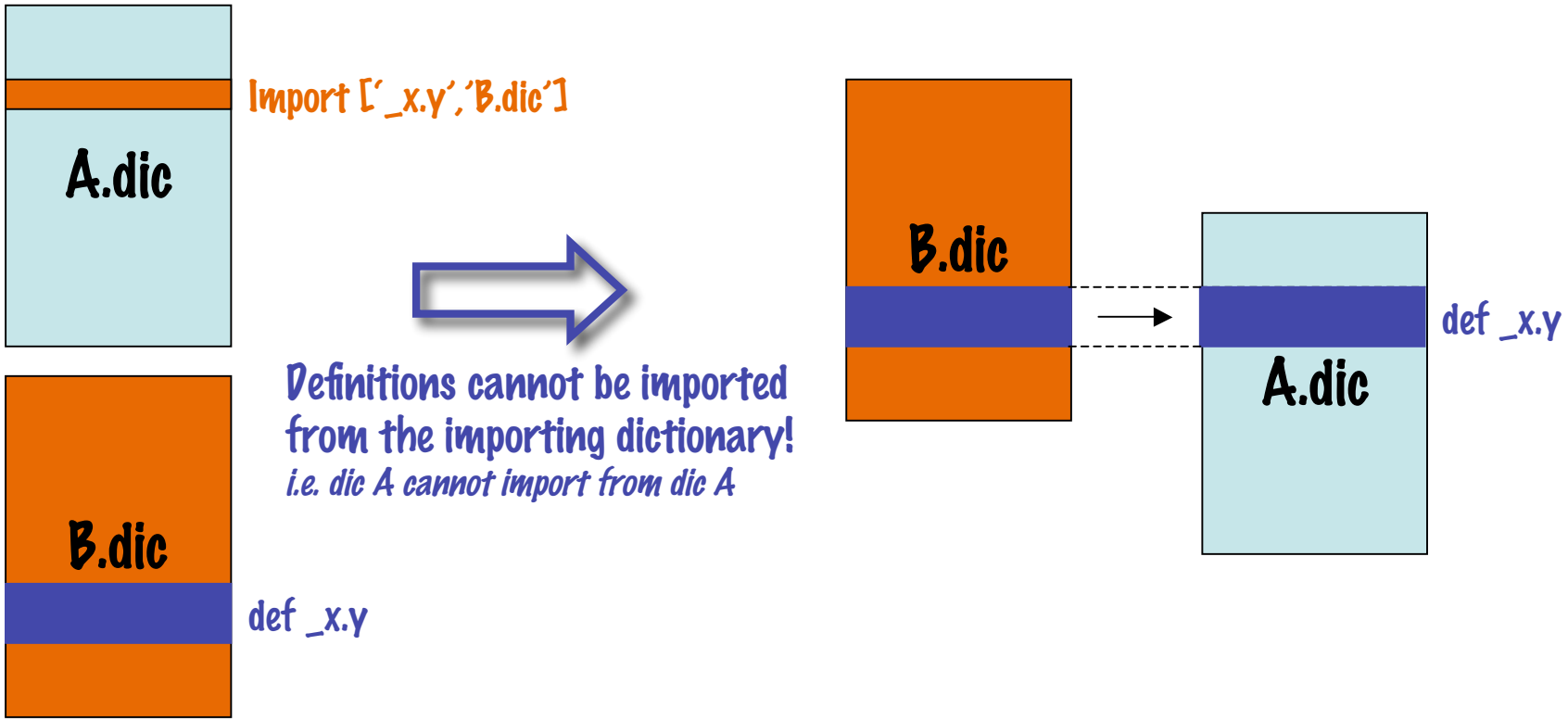




2007

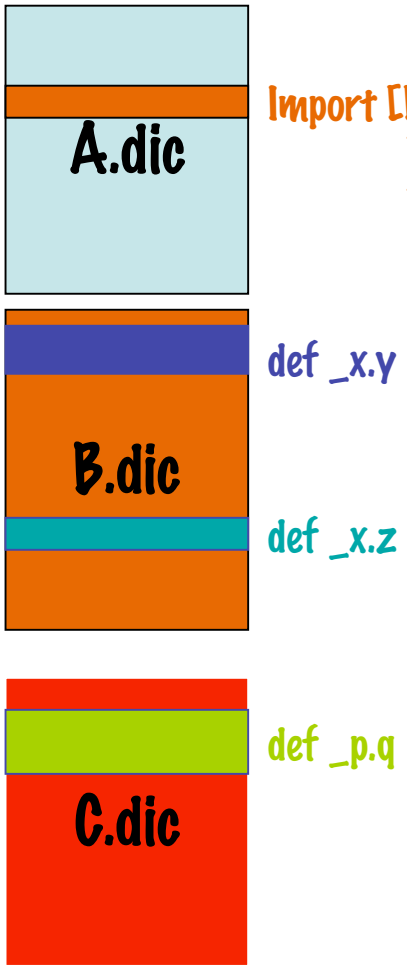
Example import with scope = 'itm' #1

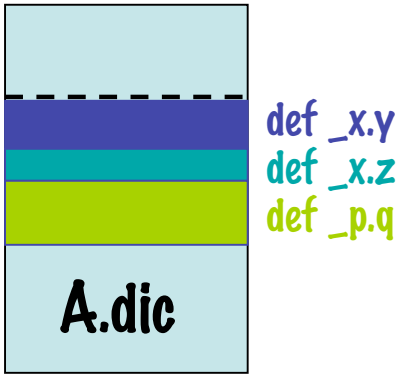
Example: *dic A imports an item definition from dic B*



Example import with scope = 'itm' #2

Example: *dictionary A imports multiple item definitions*



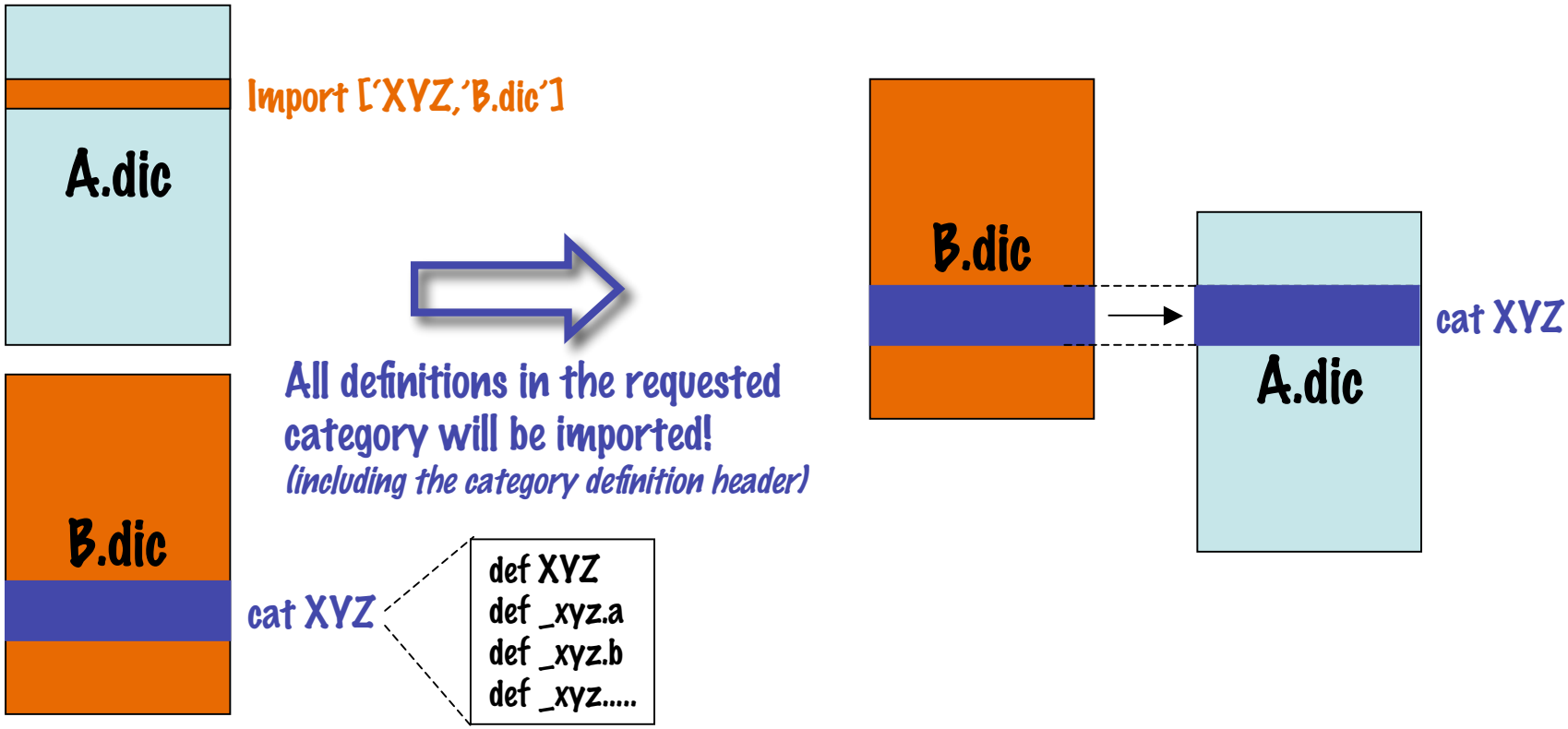


Only <u>one</u> import_definition <u>attribute</u> type is allowed per data/save block!

Hence the need for nested tuples.

Example import with scope = 'cat' #1

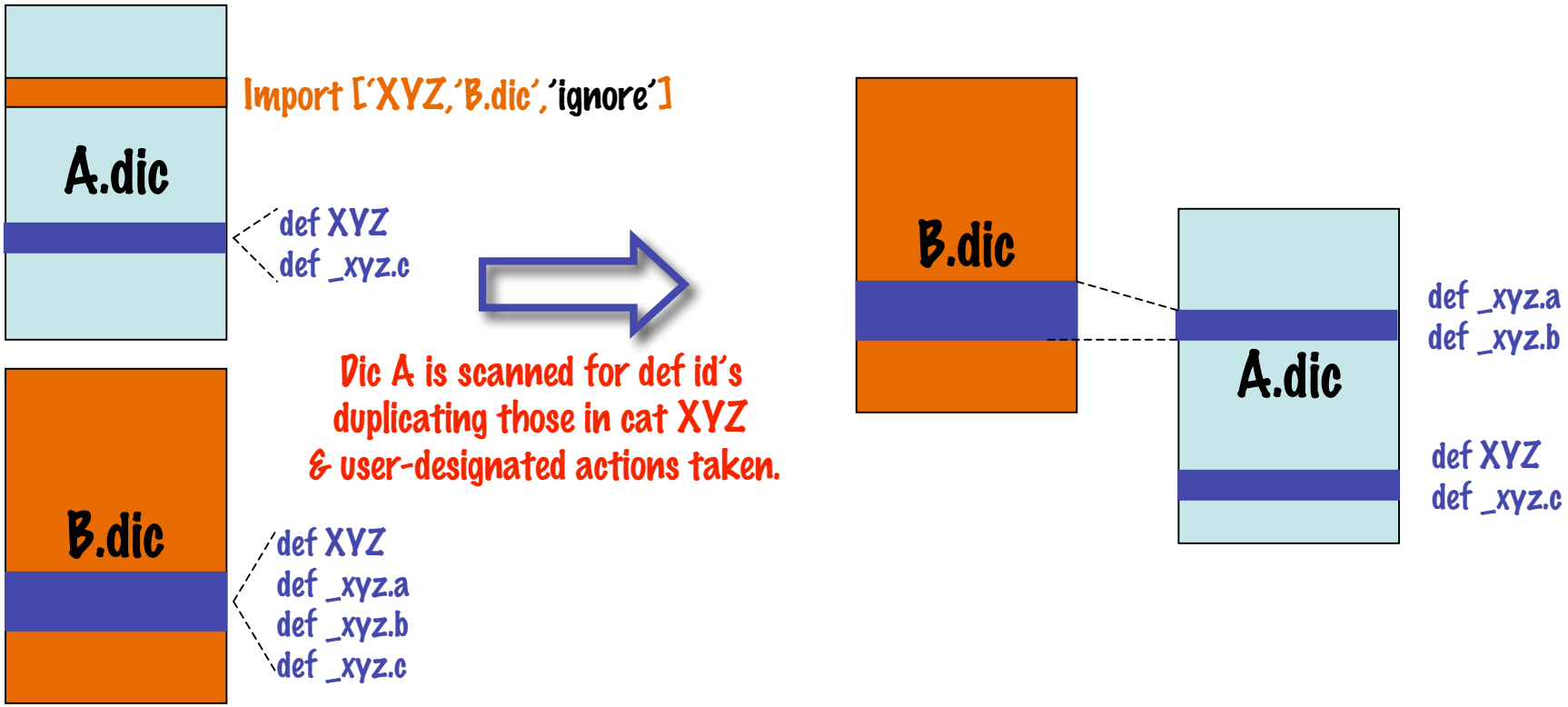
Example: *dic A imports unique category from dic B*



2007

Example import with scope = 'cat' #2

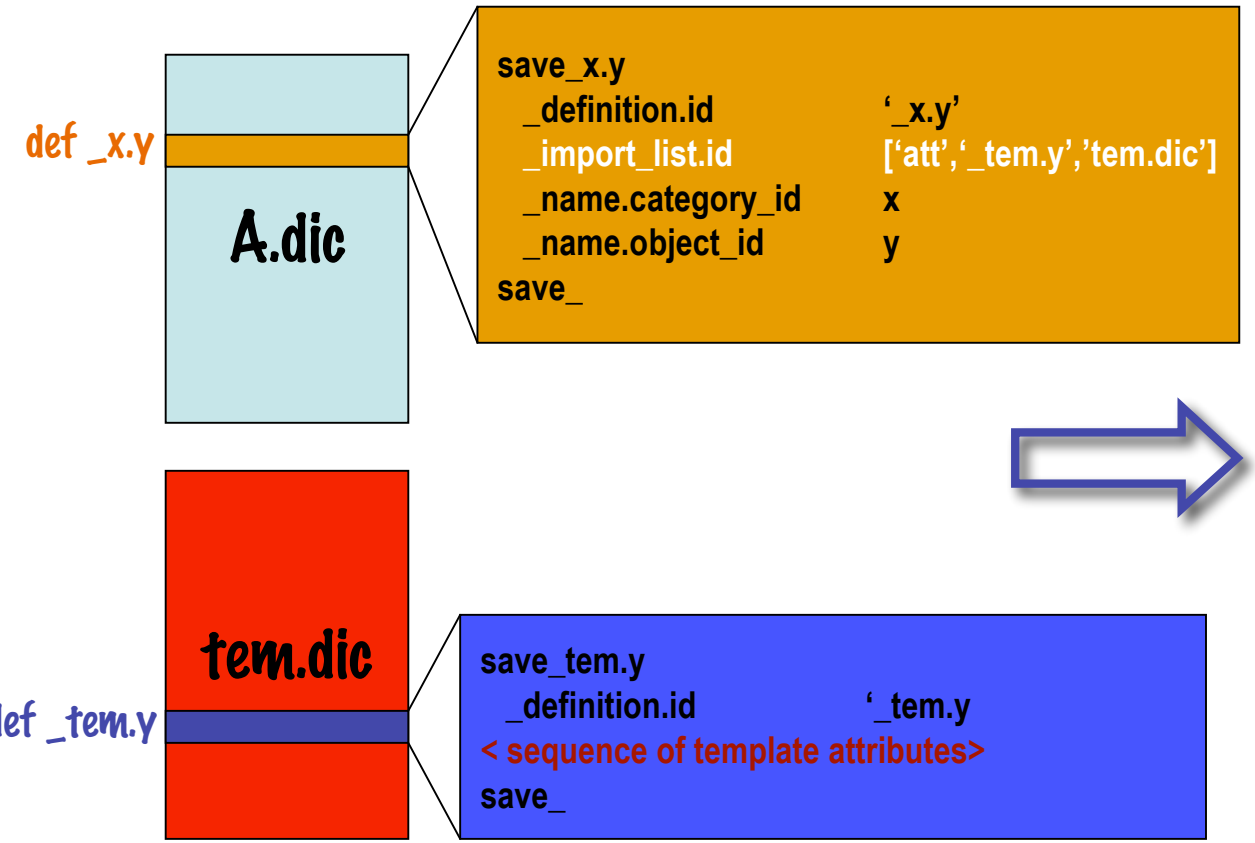
Example: *dic A imports a category with non-unique items from dic B*



2007

Example import with scope = 'att'

Example: dictionary A imports definition attributes



def _x.y in A.dic

```
save_x.y
  _definition.id              '_x.y'
< sequence of template attributes>
  _name.category_id           x
  _name.object_id             y
save_
```

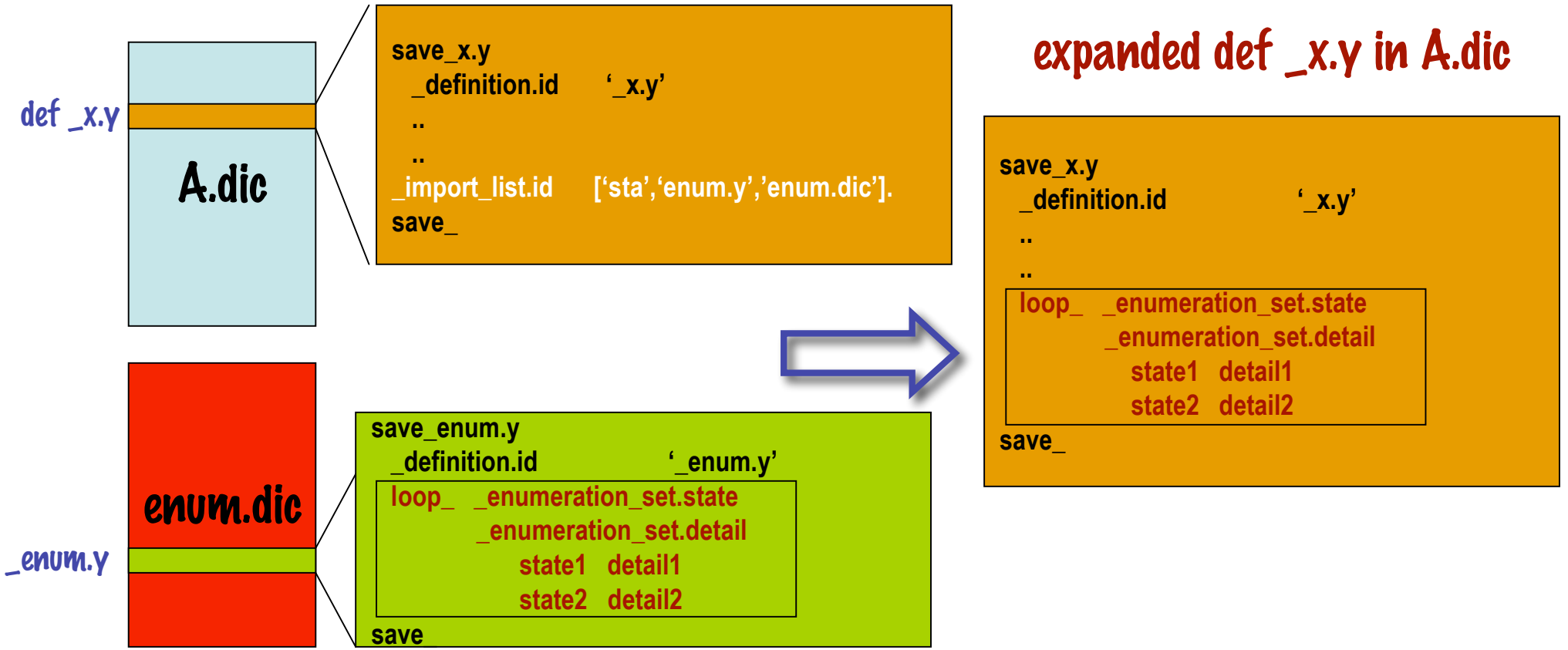
NB: The template sequence in *tem.dic* must not contain the name attributes:

_name.category_id
_name.object_id

2007

Example import with scope = 'sta'

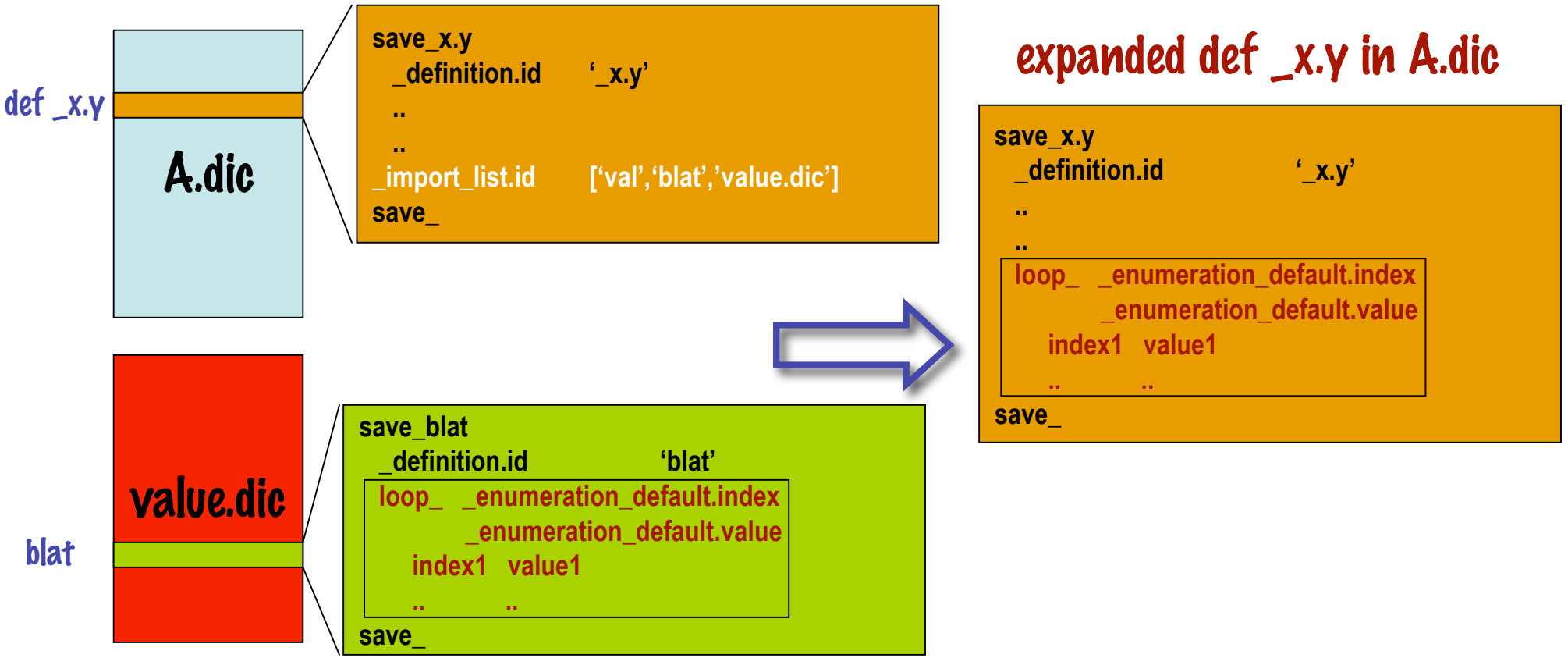
Example: dictionary A imports list of enumeration states



2007

Example import with scope = 'val'

Example: dictionary A imports an default value list



2007

Importation Protocols
User Designated Actions

>>>>> Critical to coalescing distributed dictionaries is a rigorous protocol for resolving duplicate or missing definitions.

- **Actions if a duplicate definition is encountered :**

  Ignore - ignore import request
  Replace - replace *existing* definition with *imported* definition block
  Exit - exit with fatal error

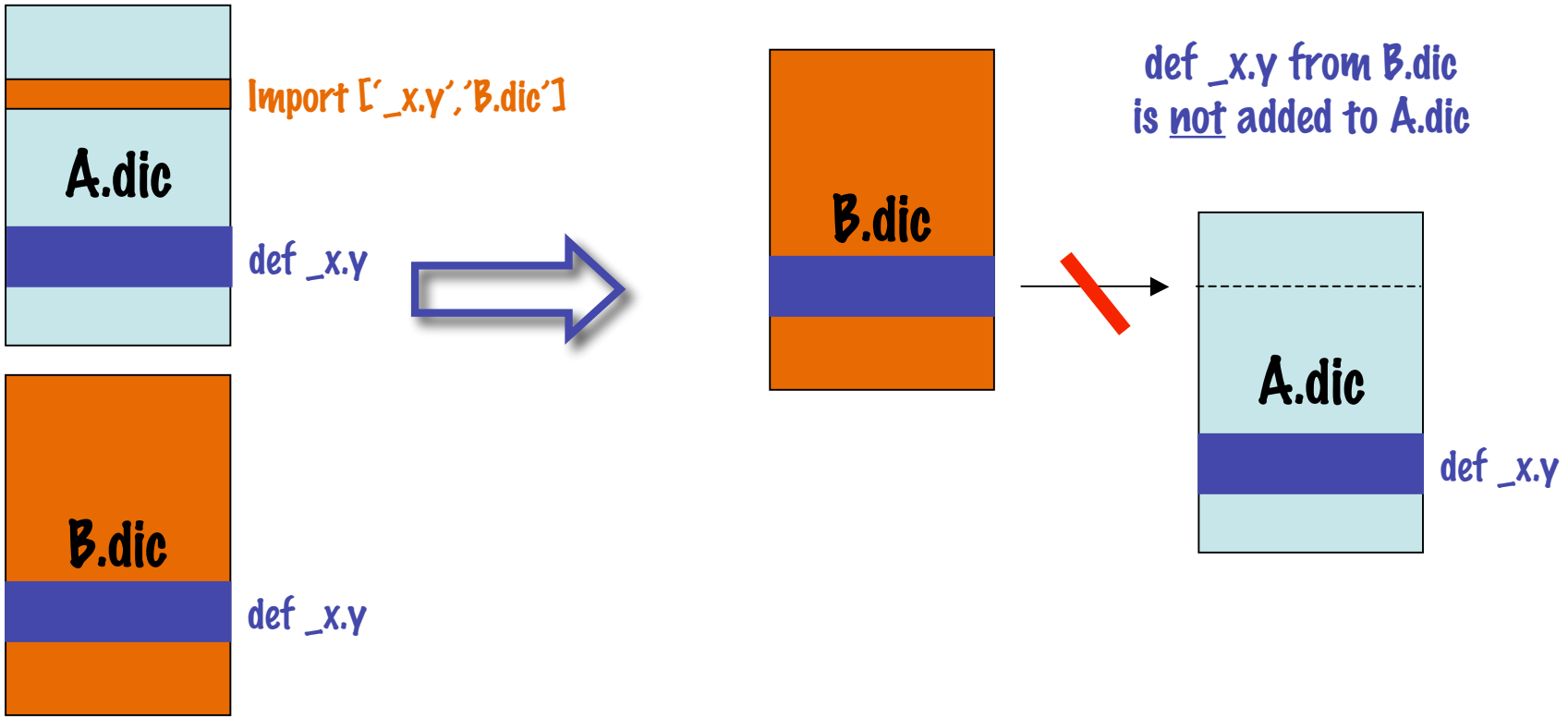
- **Actions if a definition is missing :**

  Ignore - ignore import request
  Exit - exit with fatal error

2007

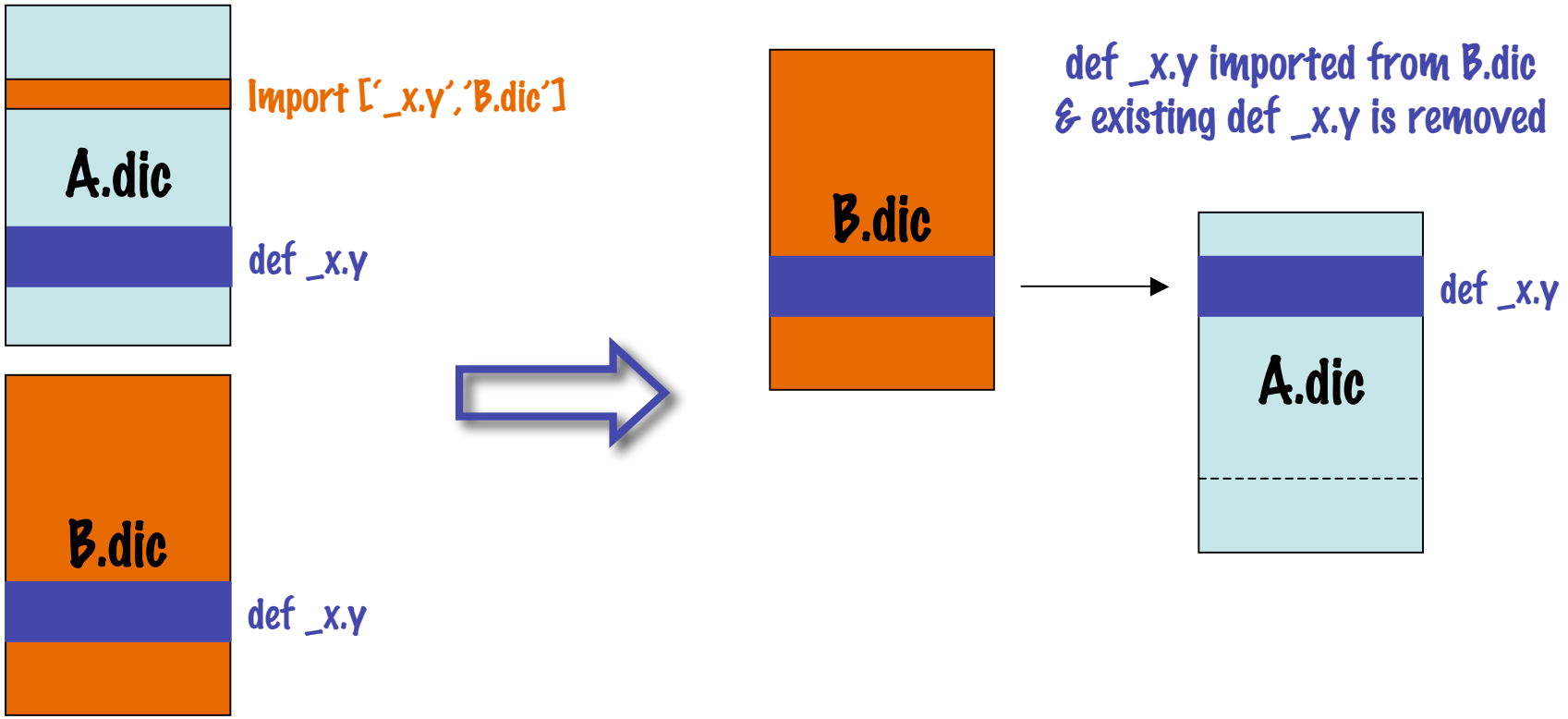
Importation Protocols: Duplicate #1

Example: *If the duplicate action is set to "ignore"*



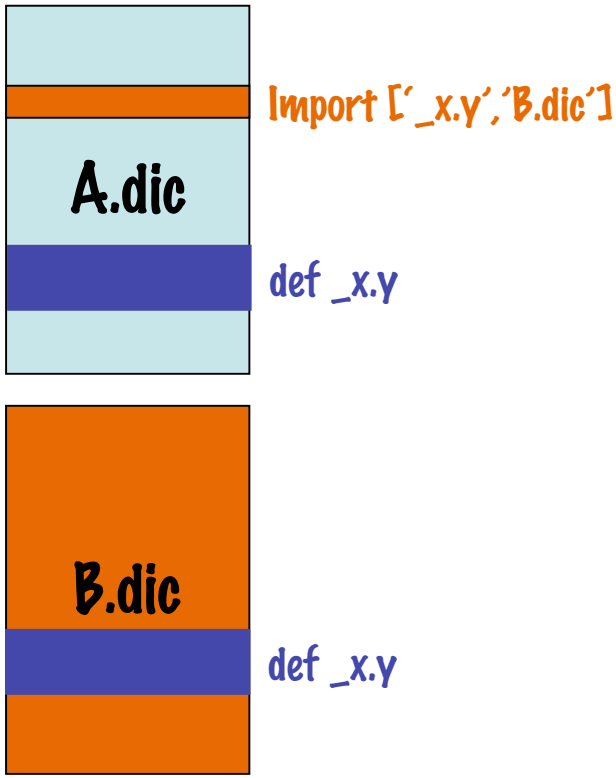
Importation Protocols: Duplicate #2

Example: *If the duplicate action is set to "replace"*



Importation Protocols: Duplicate #3

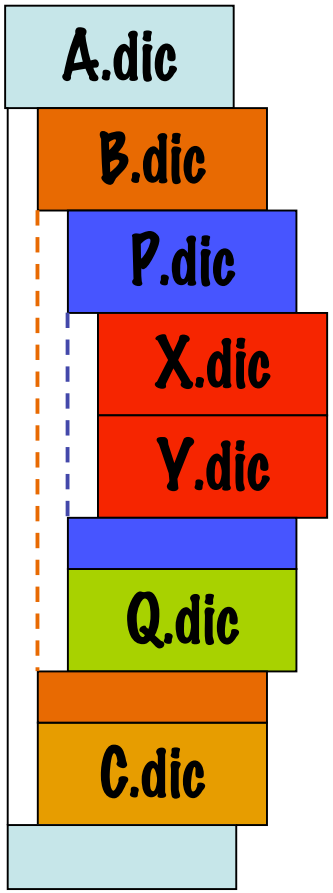
Example: *If the duplicate code is set at "exit"*



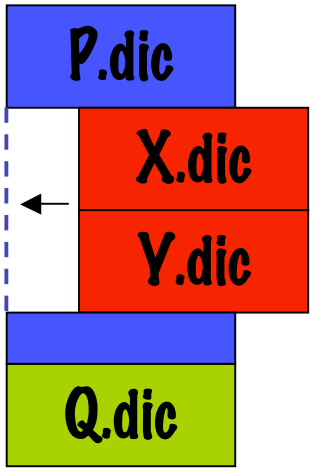
Duplicate causes a fatal error and the entire importation process is stopped.

Importation Protocols: Conflicts #4

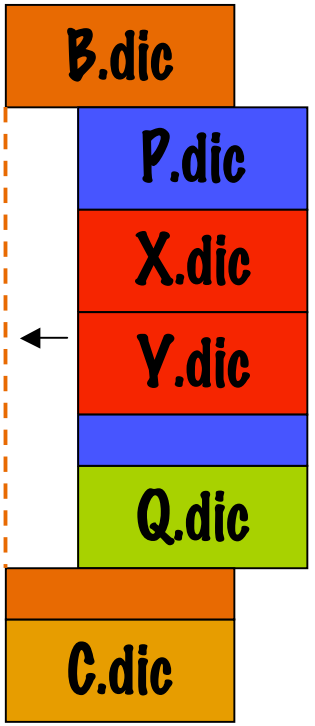
Complex example: *Handling conflicts in nested importations*



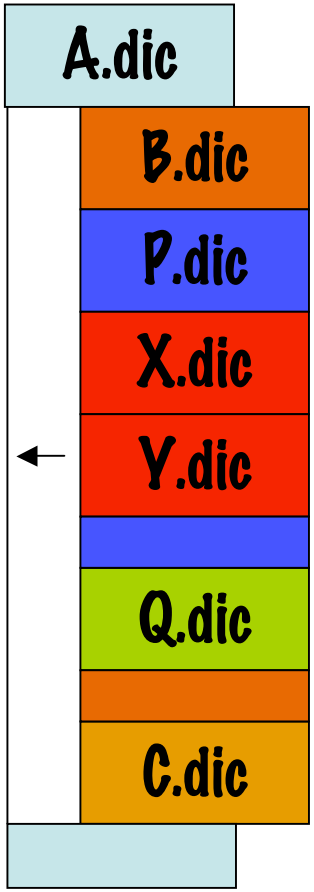
Step 1:
Handle conflicts for
XY imported to PQ



Step 2:
XYPQ imported to BC



Step 3:
XYPQBC imported to A



Conflict protocols are applied
inner-most to outer-most
imported definition levels

2007