DDL Import Attributes

2008

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.

2008

The IMPORT attributes

- A definition IMPORT is specified using the attributes:

  _import.scope - import scope 'Dic','Cat','Grp','Def','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

2008

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)
Def import a single definition saveframe

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

2008

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]

2008

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)

2008

Typical IMPORT examples

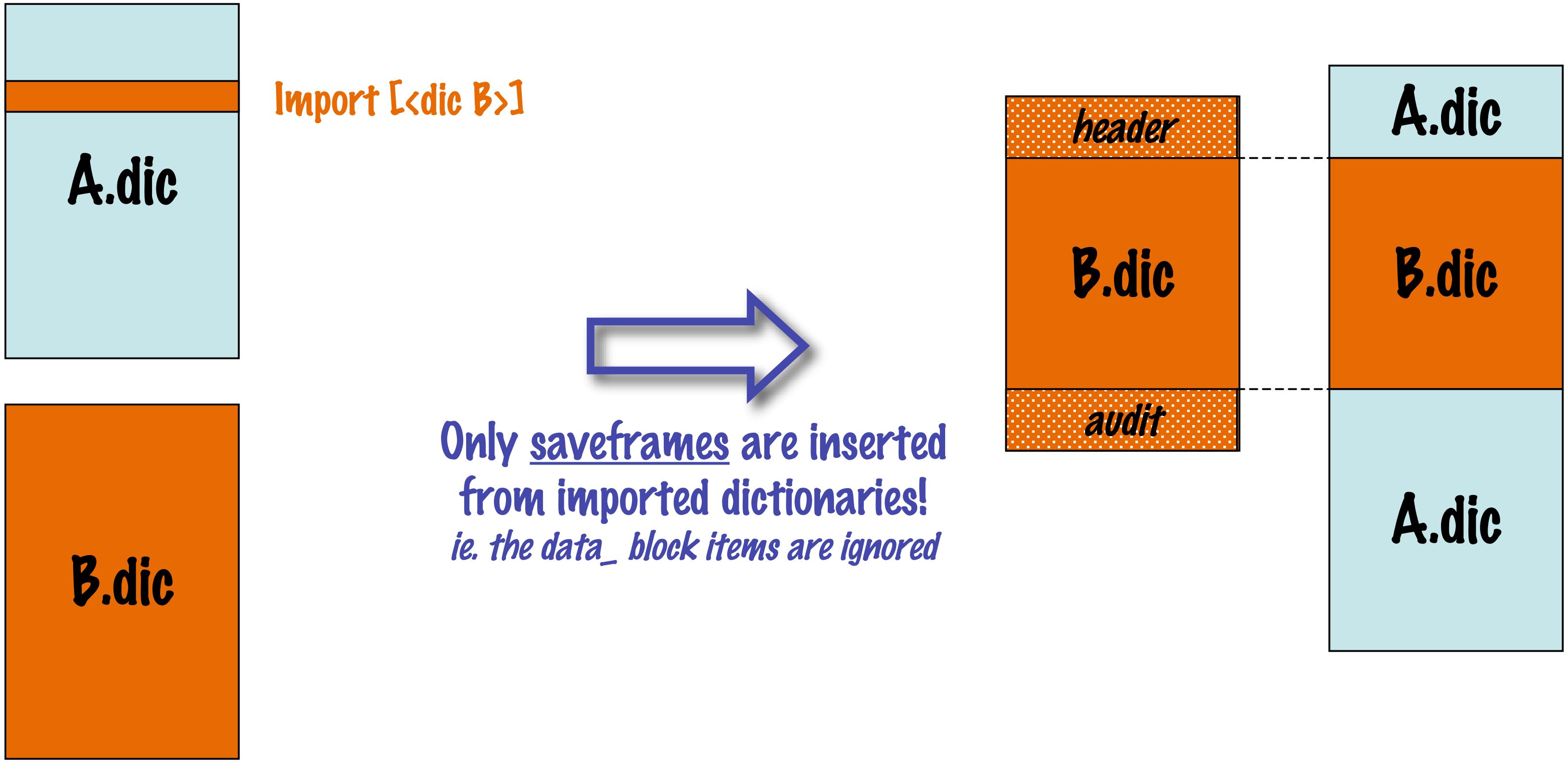
```
loop_
    _import.scope
    _import.block
    _import.file
    _import.if_dupl
    _import.if_miss
      'Dic'  'CORE_CRYSTAL'            'core_crystal.dic'     'Exit'    'Exit'
      'Cat'  'ATOM_SITE'               'core_structure.dic'   'Ignore'  'Exit'
      'Grp'  'CELL'                    'core_crystal.dic'     'Replace' 'Exit'
      'Def'  '_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' ]
```

2008

Example import with scope = 'Dic' #1

Simple example: dictionary A imports dictionary B



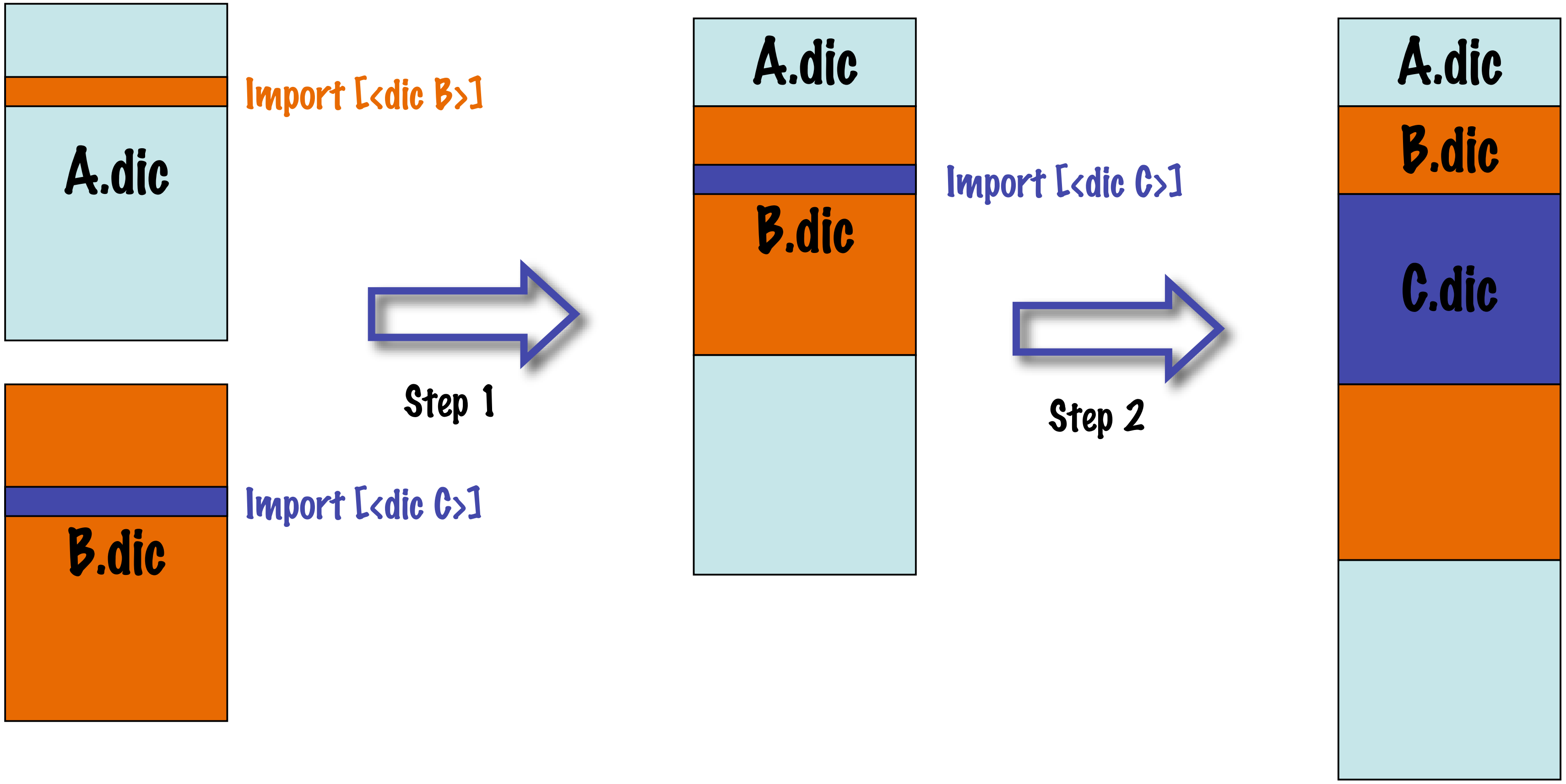
Only saveframes are inserted from imported dictionaries!

ie. the data_ block items are ignored

2008

Example import with scope = ‘Dic’ #2

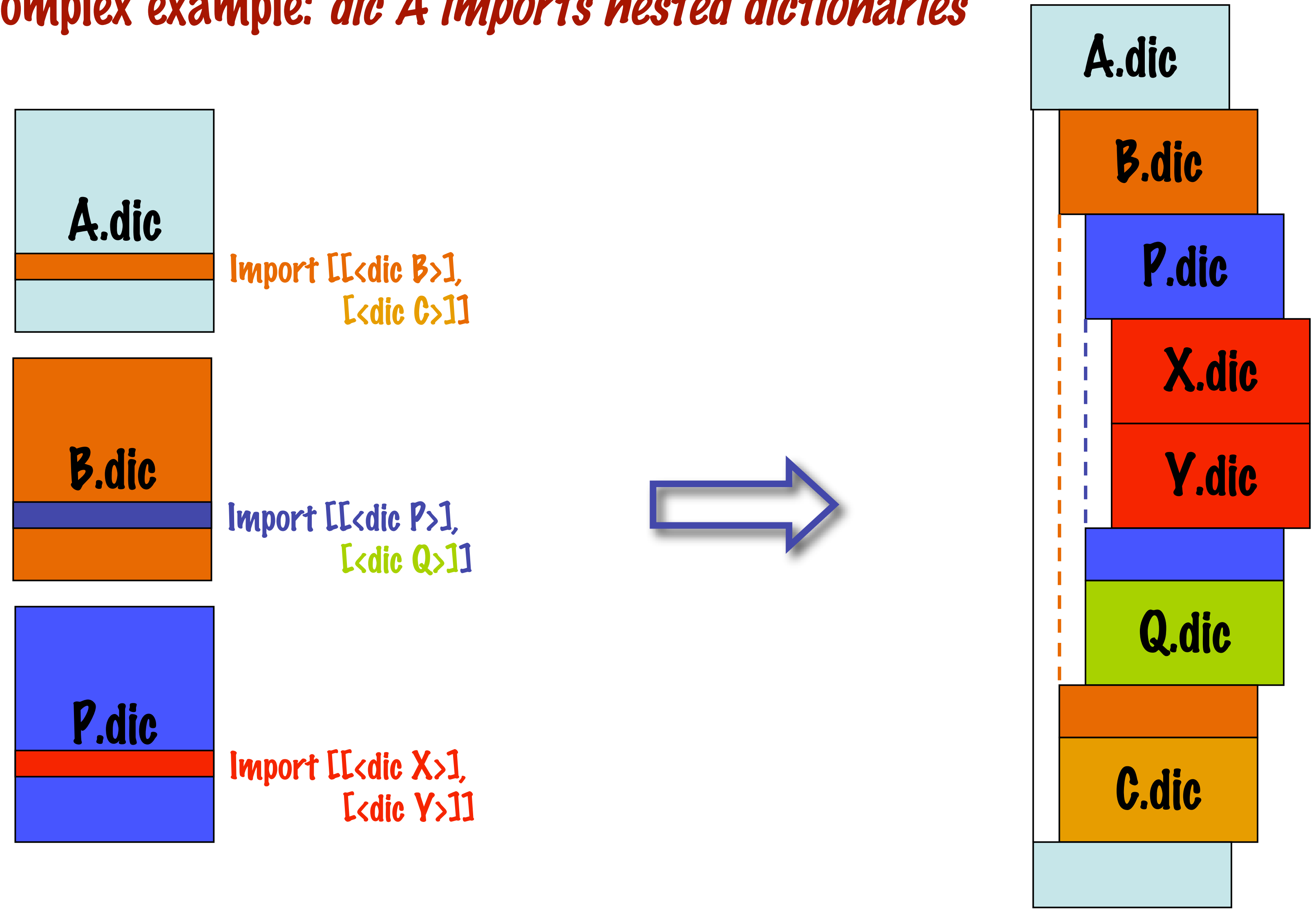
Nested example: *dic A imports dic B and dic B imports dic C*



2008

Example import with scope = 'Dic' #3

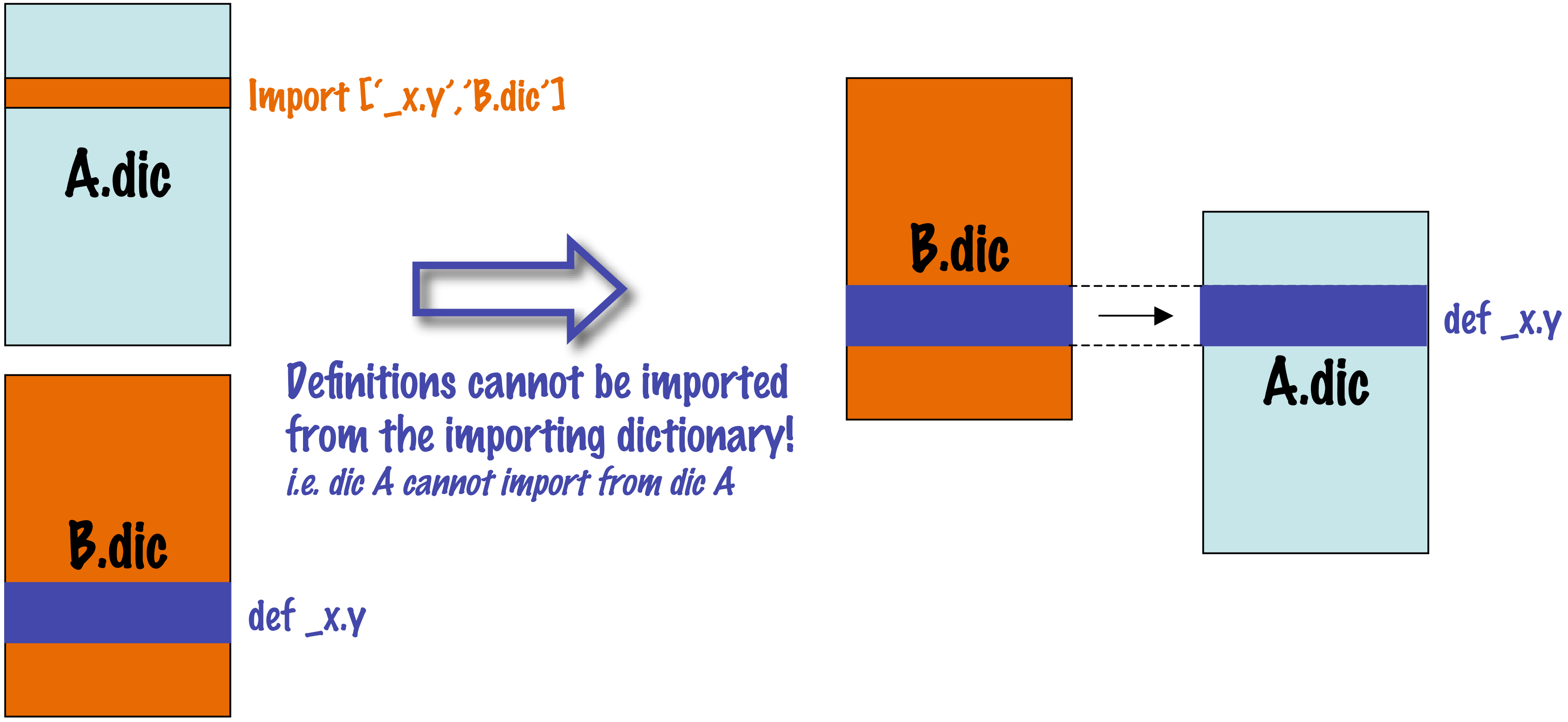
Complex example: *dic A imports nested dictionaries*



2008

Example import with scope = 'Def' #1

Example: *dic A imports a definition from dic B*

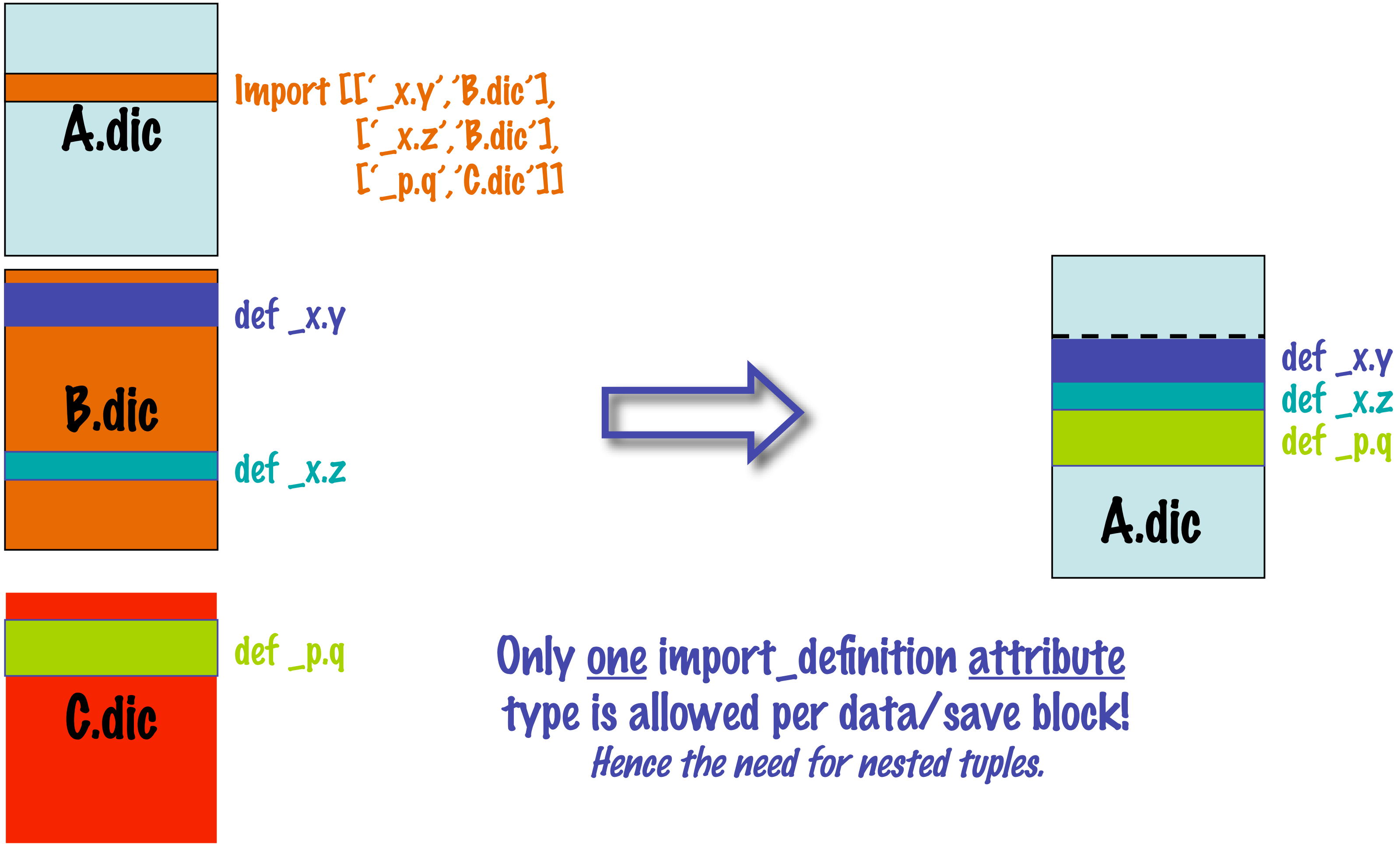


Definitions cannot be imported from the importing dictionary!
i.e. dic A cannot import from dic A

2008

Example import with scope = 'Def' #2

Example: *dictionary A imports multiple definitions*



A.dic

Import [['_x.y','B.dic'],
            ['_x.z','B.dic'],
            ['_p.q','C.dic']]

B.dic

def _x.y

def _x.z

C.dic

def _p.q

def _x.y
def _x.z
def _p.q

A.dic

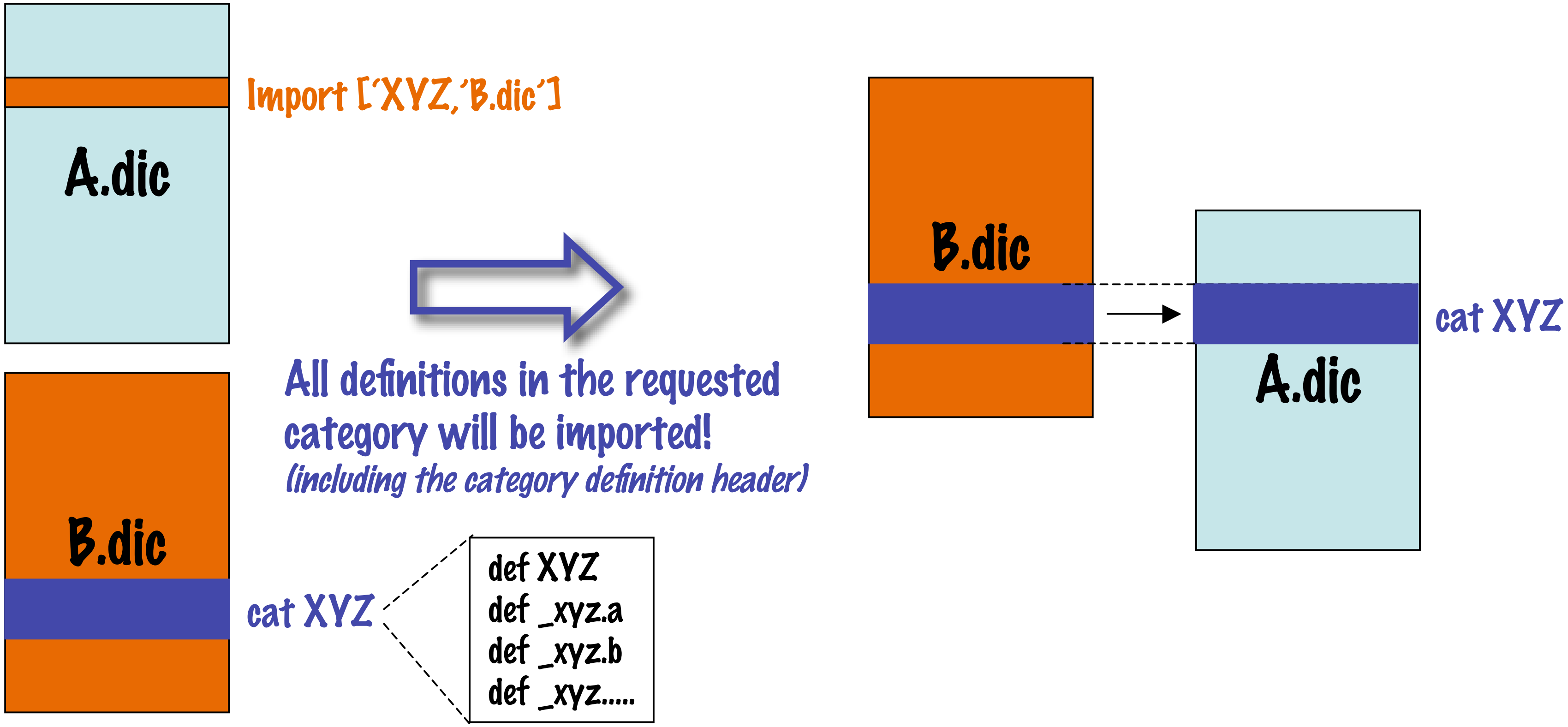
Only one import_definition attribute
type is allowed per data/save block!

Hence the need for nested tuples.

2008

Example import with scope = 'Cat' #1

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



All definitions in the requested category will be imported!

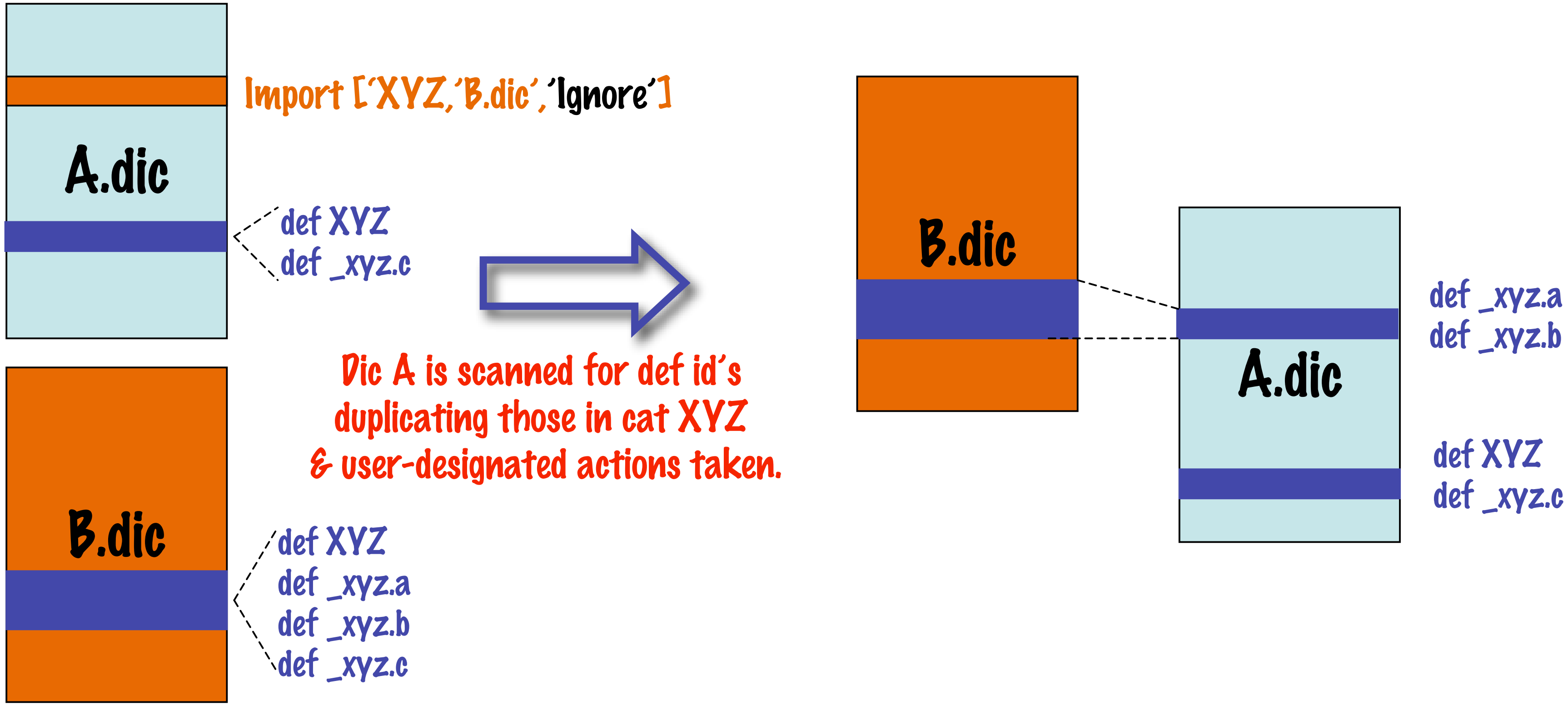
(including the category definition header)

```
def XYZ
def _xyz.a
def _xyz.b
def _xyz.....
```

2008

Example import with scope = ‘Cat’ #2

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

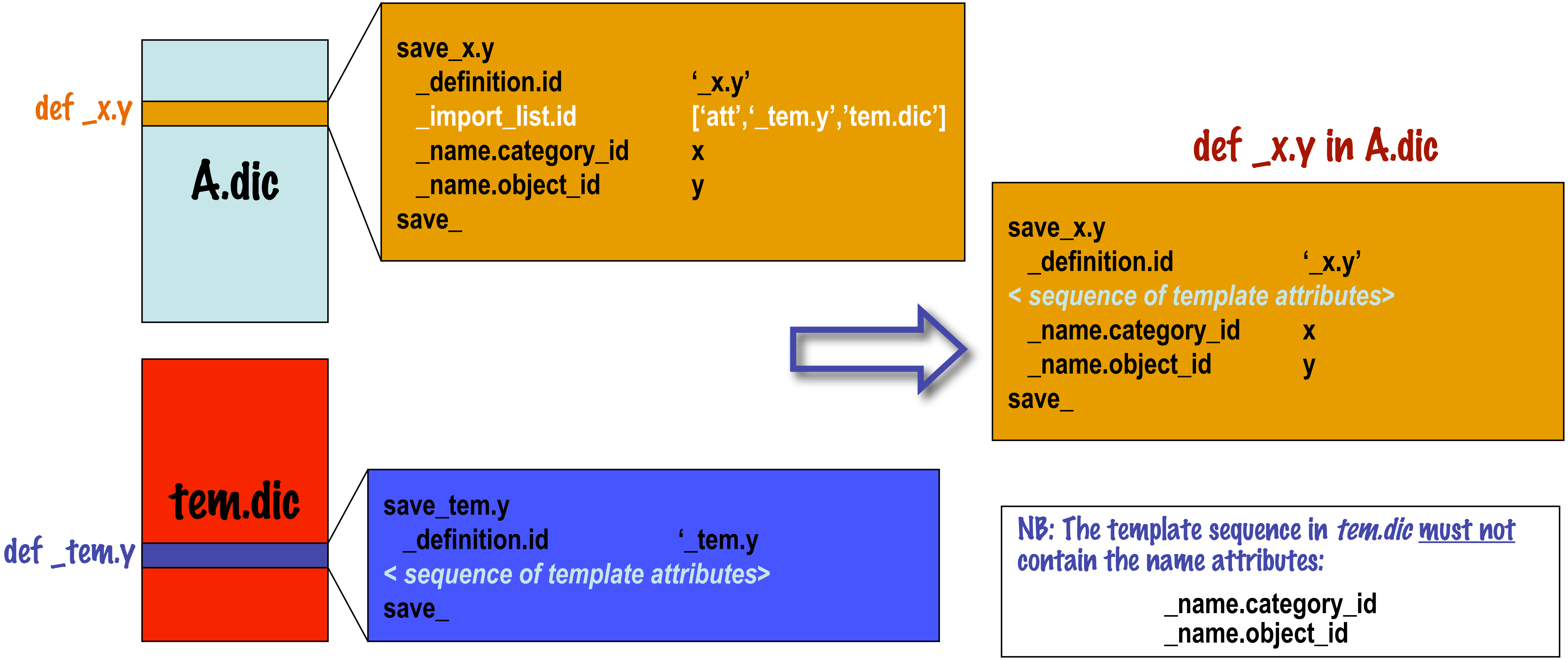


Dic A is scanned for def id’s duplicating those in cat XYZ & user-designated actions taken.

2008

Example import with scope = 'Att'

Example: dictionary A imports definition attributes



```
save_x.y
   _definition.id              '_x.y'
   _import_list.id             ['att','_tem.y','tem.dic']
   _name.category_id           x
   _name.object_id             y
save_
```

```
save_tem.y
   _definition.id              '_tem.y
< sequence of template attributes>
save_
```

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_
```

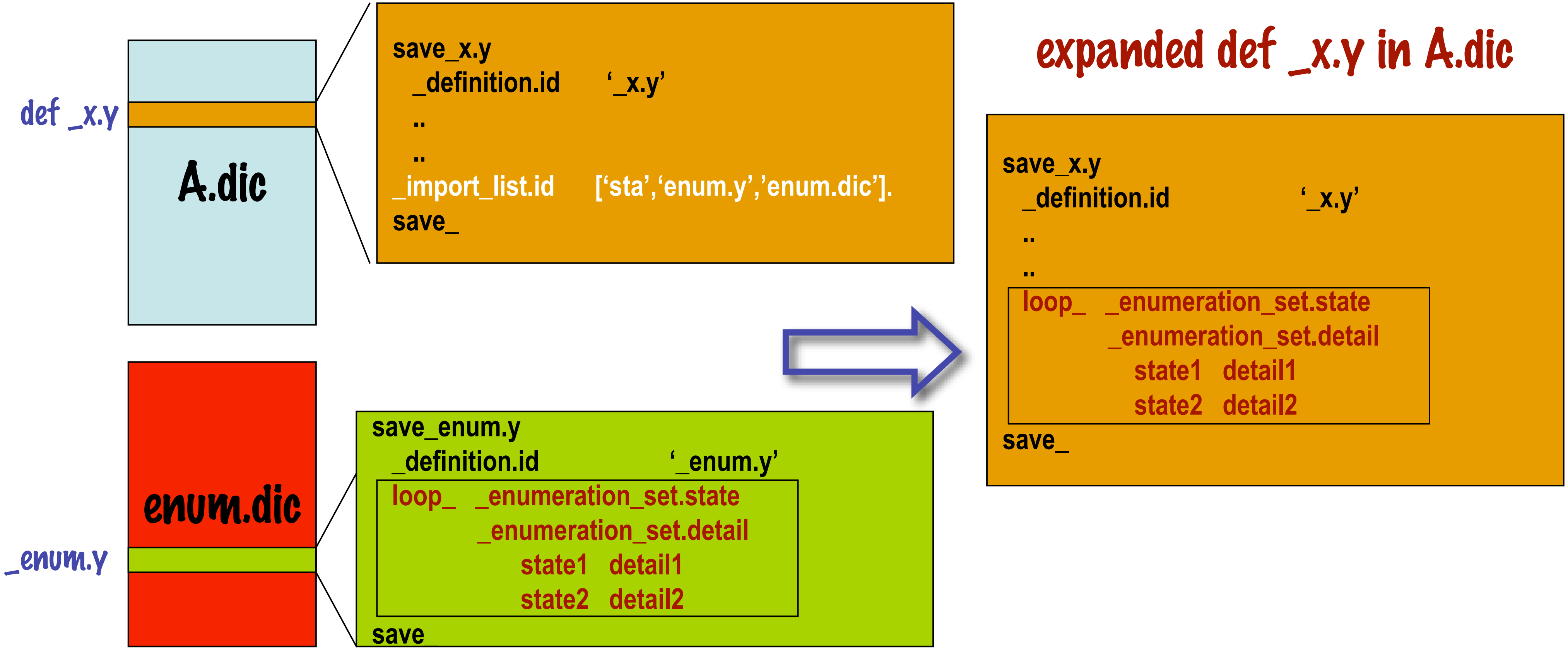
NB: The template sequence in *tem.dic* must not contain the name attributes:

_name.category_id
_name.object_id

2008

Example import with scope = ‘Sta’

Example: dictionary A imports list of enumeration states



def _x.y

A.dic

```
save_x.y
  _definition.id      ‘_x.y’
  ..
  ..
_import_list.id     [‘sta’,‘enum.y’,’enum.dic’].
save_
```

enum.dic

_enum.y

```
save_enum.y
  _definition.id                    ‘_enum.y’
  loop_  _enumeration_set.state
           _enumeration_set.detail
                 state1   detail1
                 state2   detail2
save
```

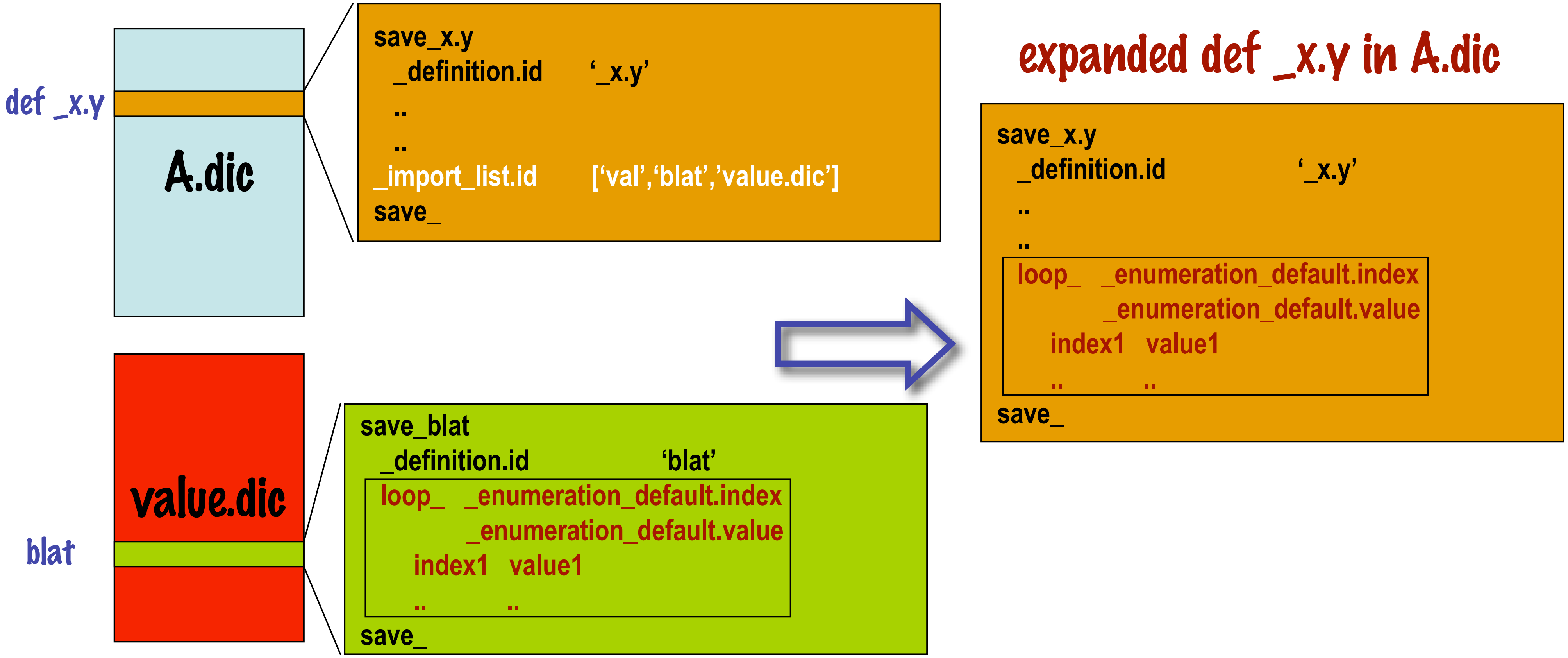
expanded def _x.y in A.dic

```
save_x.y
  _definition.id                    ‘_x.y’
  ..
  ..
  loop_  _enumeration_set.state
           _enumeration_set.detail
                 state1   detail1
                 state2   detail2
save_
```

2008

Example import with scope = ‘Val’

Example: dictionary A imports an default value list



```
save_x.y
  _definition.id       ‘_x.y’
  ..
  ..
_import_list.id        [‘val’,‘blat’,’value.dic’]
save_
```

```
save_blat
  _definition.id                  ‘blat’
  loop_  _enumeration_default.index
         _enumeration_default.value
     index1   value1
     ..          ..
save_
```

expanded def _x.y in A.dic

```
save_x.y
  _definition.id                  ‘_x.y’
  ..
  ..
  loop_  _enumeration_default.index
         _enumeration_default.value
     index1   value1
     ..          ..
save_
```

2008

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

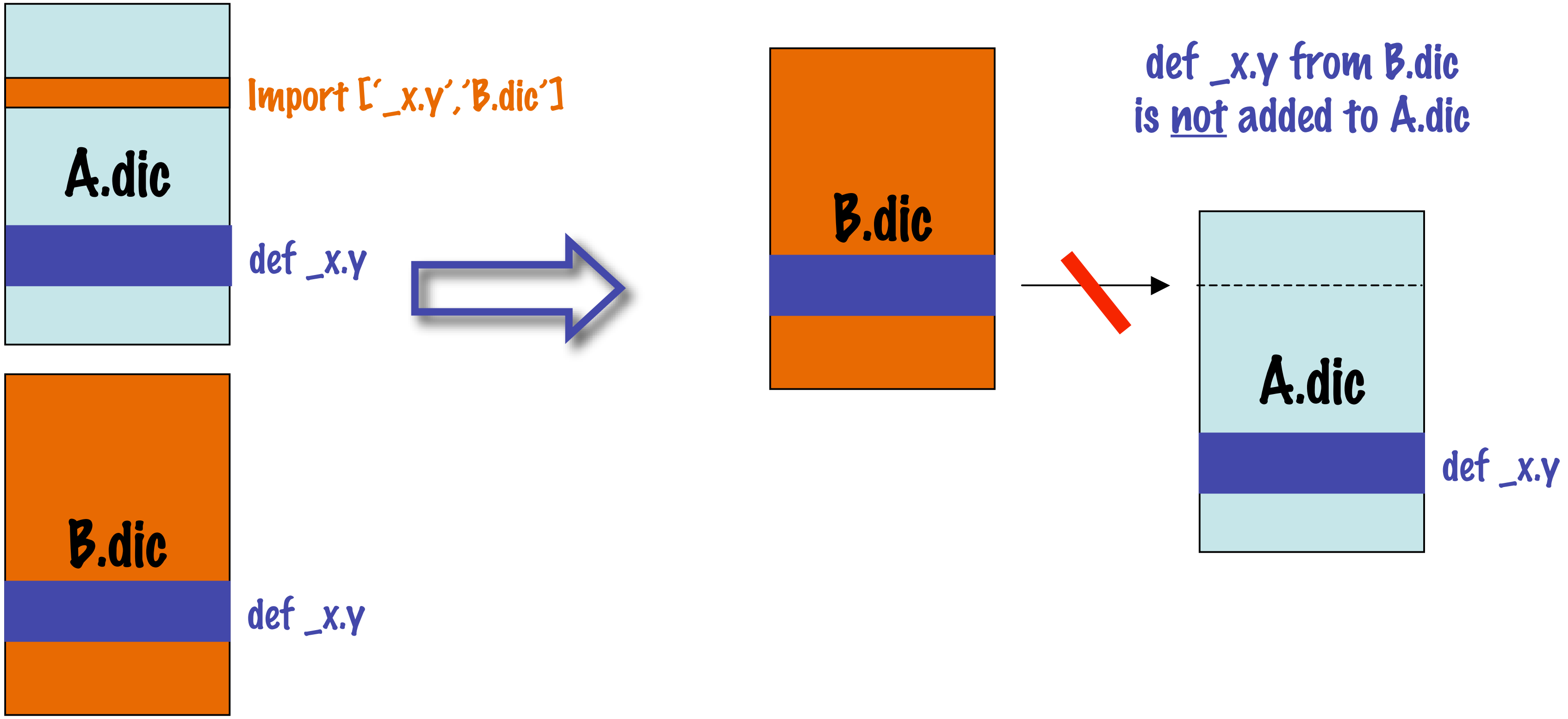
- Actions if a definition is missing :

  Ignore - ignore import request
  Exit - exit with fatal error

2008

Importation Protocols: Duplicate #1

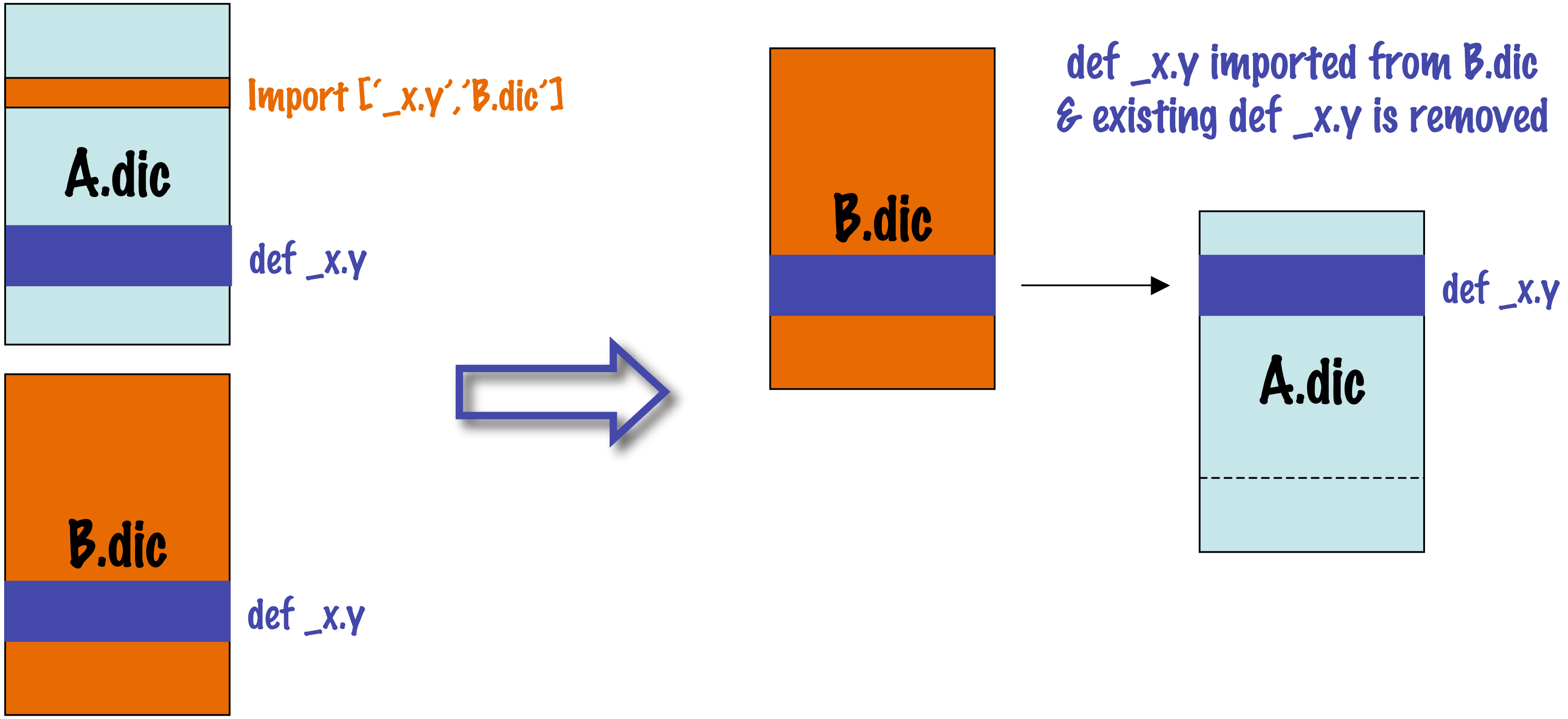
Example: *If the duplicate action is set to "Ignore"*



2008

Importation Protocols: Duplicate #2

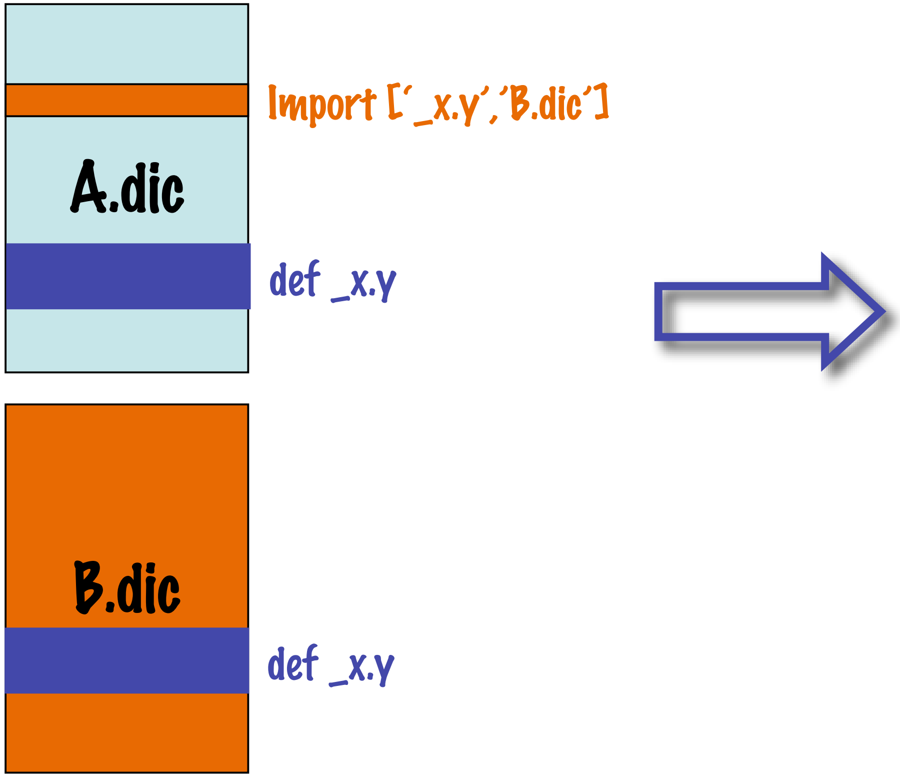
Example: *If the duplicate action is set to "Replace"*



2008

Importation Protocols: Duplicate #3

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

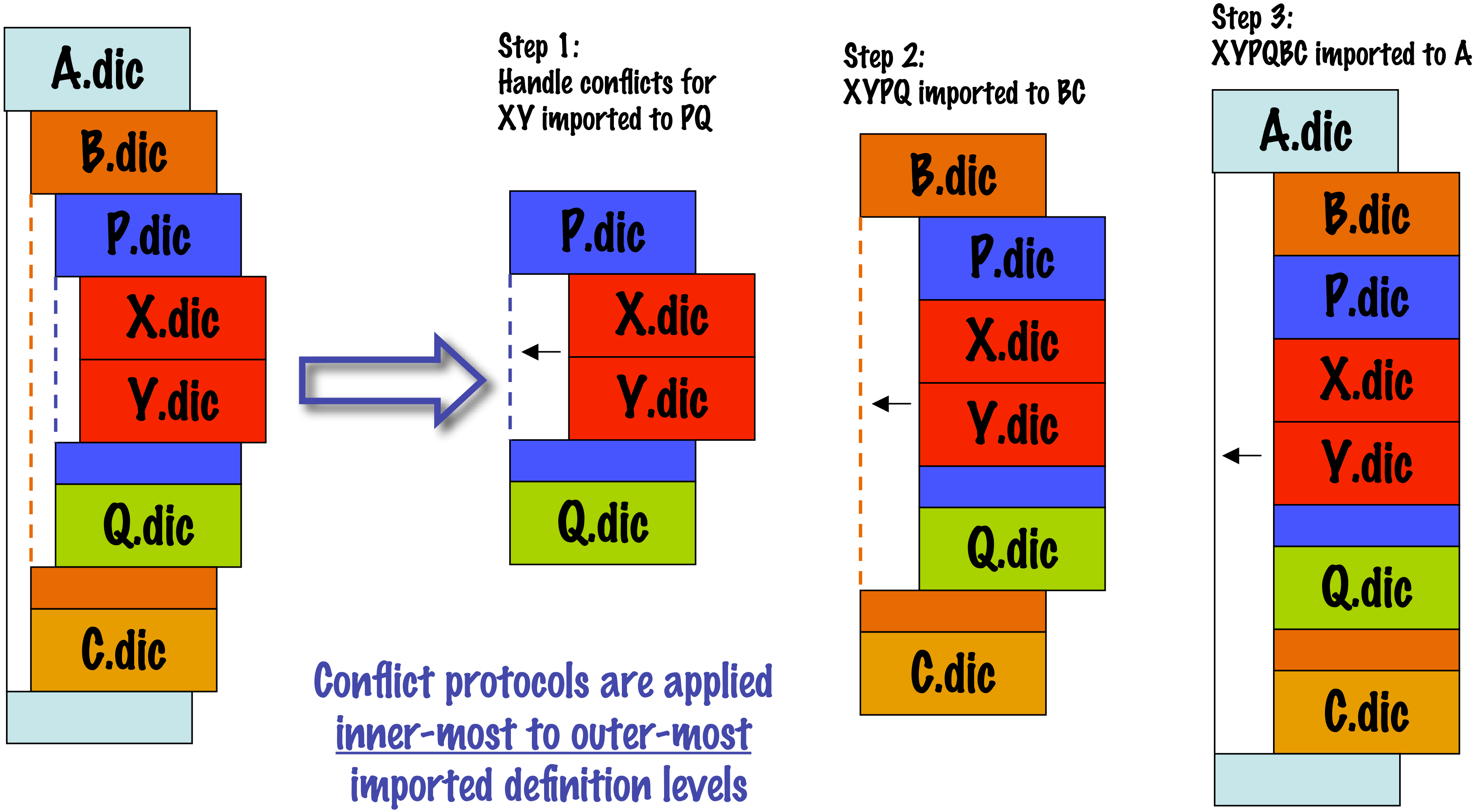


Duplicate causes a fatal error and the entire importation process is stopped.

2008

Importation Protocols: Conflicts #4

Complex example: *Handling conflicts in <u>nested</u> 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
<u>inner-most to outer-most</u>
imported definition levels

2008