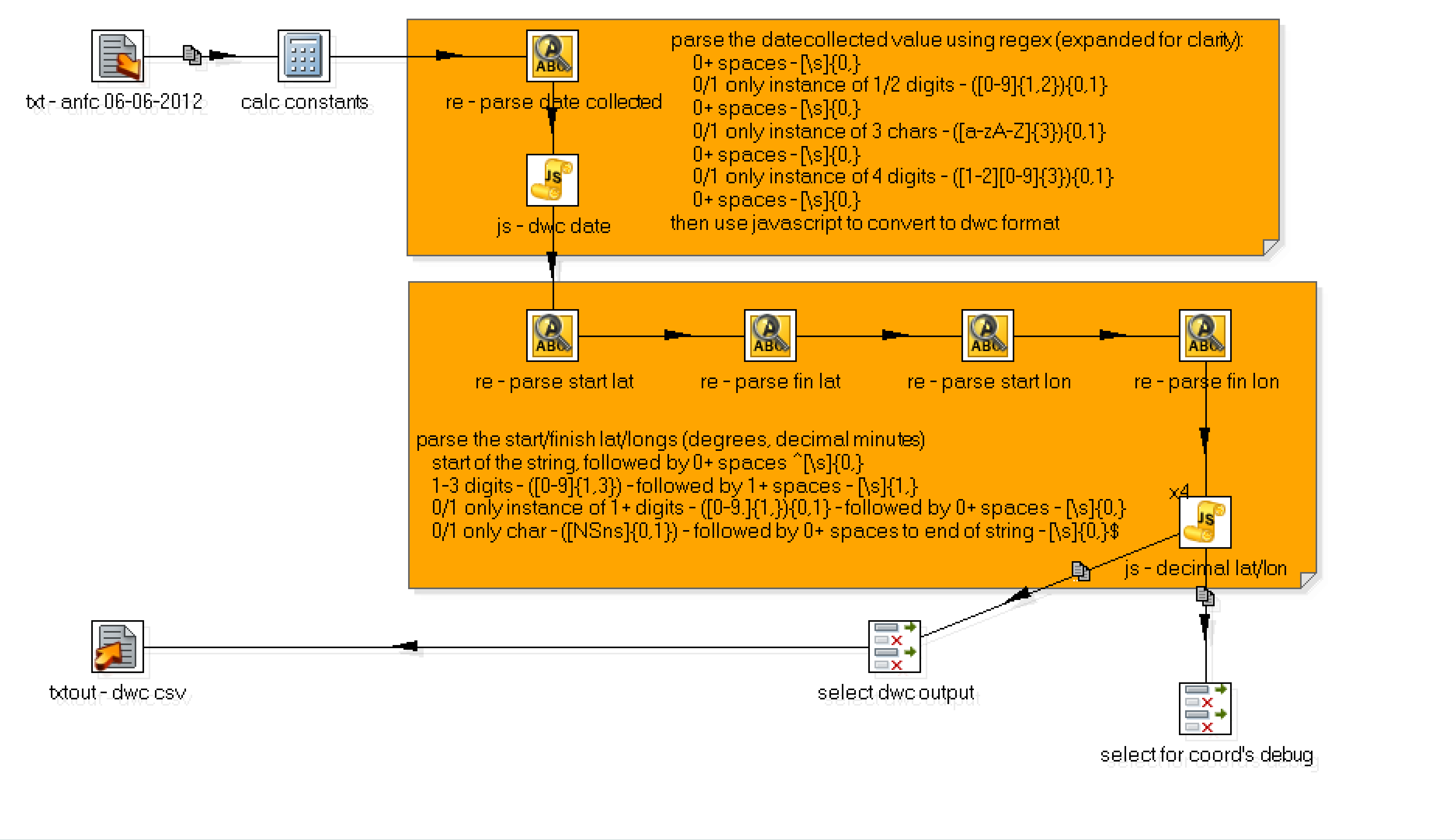
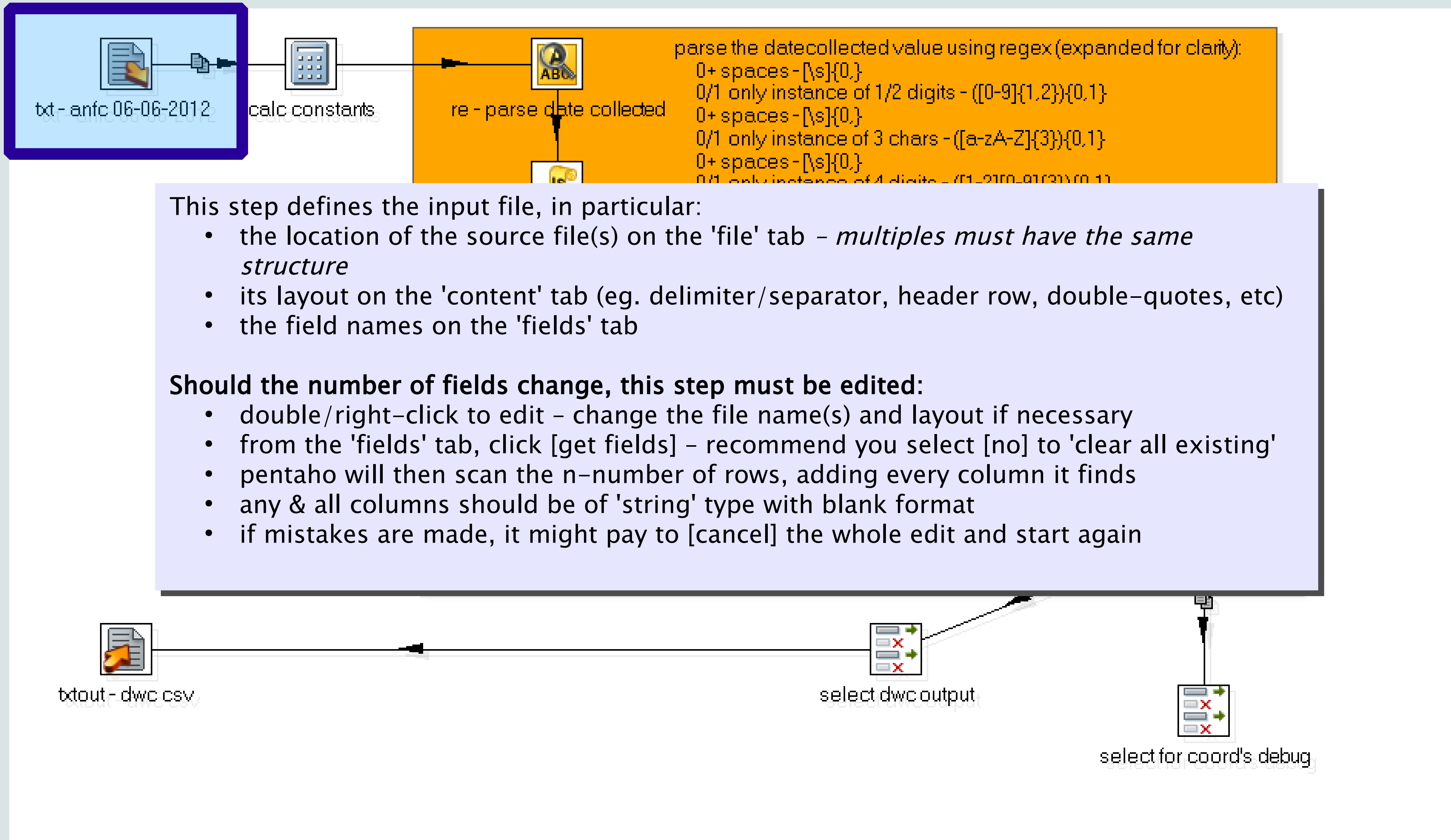


parse the datecollected value using regex (expanded for clarity):
  0+ spaces - [\s]{0,}
  0/1 only instance of 1/2 digits - ([0-9]{1,2}){0,1}
  0+ spaces - [\s]{0,}
  0/1 only instance of 3 chars - ([a-zA-Z]{3}){0,1}
  0+ spaces - [\s]{0,}
  0/1 only instance of 4 digits - ([1-2][0-9]{3}){0,1}
  0+ spaces - [\s]{0,}
then use javascript to convert to dwc format

parse the start/finish lat/longs (degrees, decimal minutes)
  start of the string, followed by 0+ spaces ^[\s]{0,}
  1-3 digits - ([0-9]{1,3}) - followed by 1+ spaces - [\s]{1,}
  0/1 only instance of 1+ digits - ([0-9.]{1,}){0,1} - followed by 0+ spaces - [\s]{0,}
  0/1 only char - ([NSns]{0,1}) - followed by 0+ spaces to end of string - [\s]{0,}$



parse the datecollected value using regex (expanded for clarity):

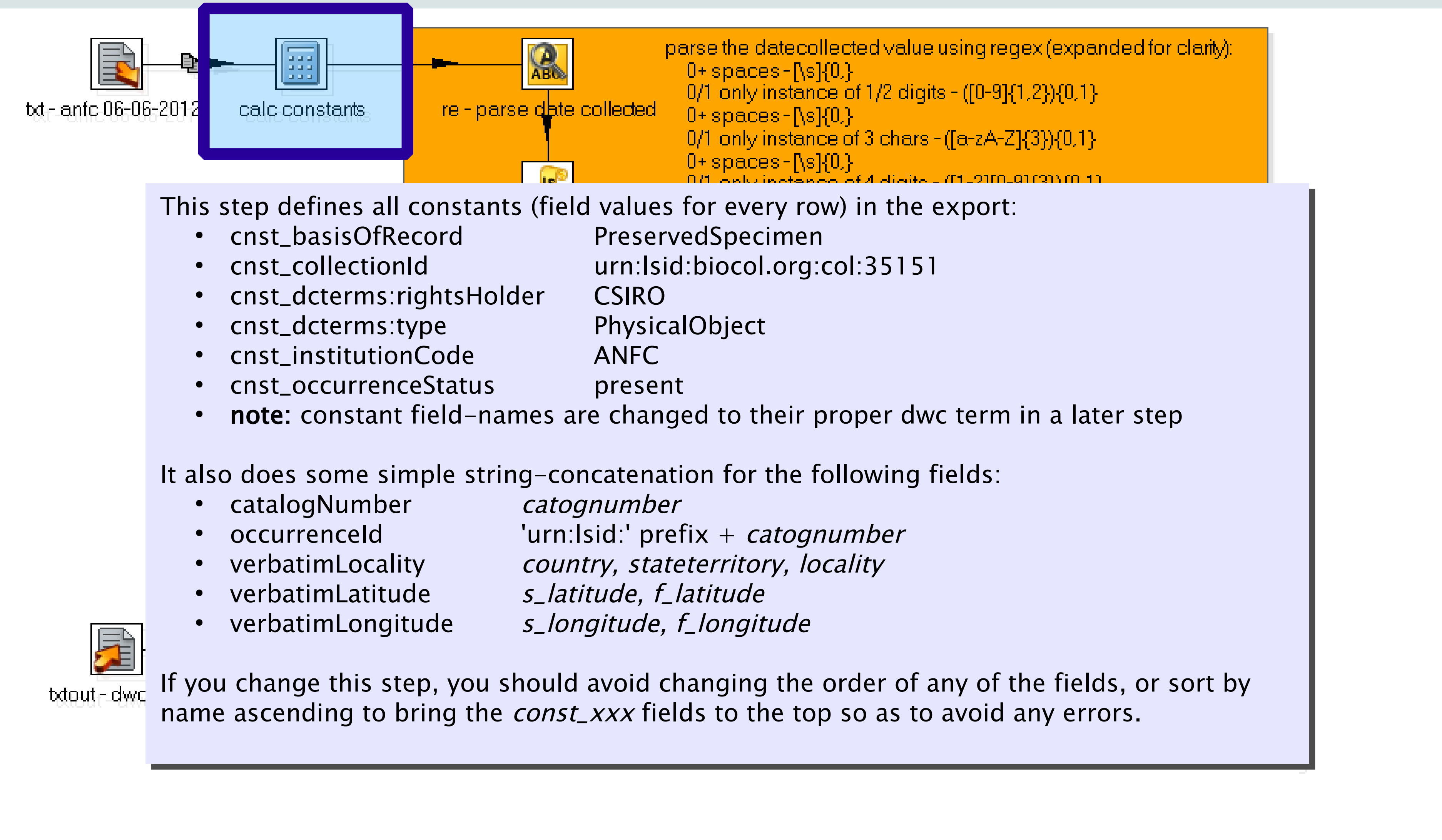
```
    0+ spaces - [\s]{0,}
    0/1 only instance of 1/2 digits - ([0-9]{1,2}){0,1}
    0+ spaces - [\s]{0,}
    0/1 only instance of 3 chars - ([a-zA-Z]{3}){0,1}
    0+ spaces - [\s]{0,}
    0/1 only instance of 4 digits - ([1-2][0-9]{3}){0,1}
```

This step defines the input file, in particular:

- the location of the source file(s) on the 'file' tab – *multiples must have the same structure*
- its layout on the 'content' tab (eg. delimiter/separator, header row, double-quotes, etc)
- the field names on the 'fields' tab

Should the number of fields change, this step must be edited:

- double/right-click to edit – change the file name(s) and layout if necessary
- from the 'fields' tab, click [get fields] – recommend you select [no] to 'clear all existing'
- pentaho will then scan the n-number of rows, adding every column it finds
- any & all columns should be of 'string' type with blank format
- if mistakes are made, it might pay to [cancel] the whole edit and start again



parse the datecollected value using regex (expanded for clarity):
0+ spaces - [\s]{0,}
0/1 only instance of 1/2 digits - ([0-9]{1,2}){0,1}
0+ spaces - [\s]{0,}
0/1 only instance of 3 chars - ([a-zA-Z]{3}){0,1}
0+ spaces - [\s]{0,}
0/1 only instance of 4 digits - ([1-2][0-9]{3}){0,1}

This step defines all constants (field values for every row) in the export:

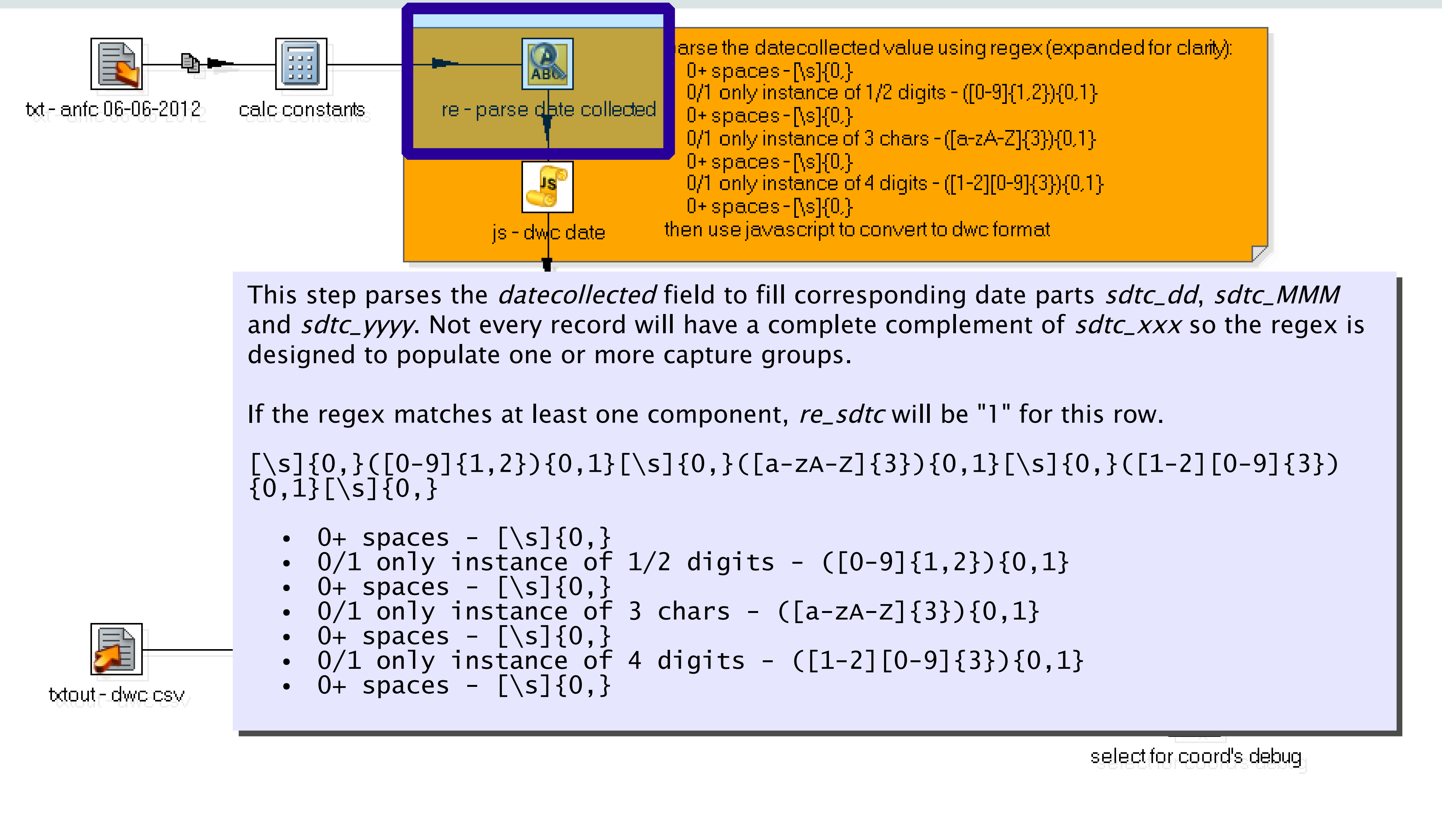
cnst_basisOfRecord	PreservedSpecimen
cnst_collectionId	urn:lsid:biocol.org:col:35151
cnst_dcterms:rightsHolder	CSIRO
cnst_dcterms:type	PhysicalObject
cnst_institutionCode	ANFC
cnst_occurrenceStatus	present

- **note:** constant field-names are changed to their proper dwc term in a later step

It also does some simple string-concatenation for the following fields:

catalogNumber	*catognumber*
occurrenceId	'urn:lsid:' prefix + *catognumber*
verbatimLocality	*country, stateterritory, locality*
verbatimLatitude	*s_latitude, f_latitude*
verbatimLongitude	*s_longitude, f_longitude*

If you change this step, you should avoid changing the order of any of the fields, or sort by name ascending to bring the *const_xxx* fields to the top so as to avoid any errors.



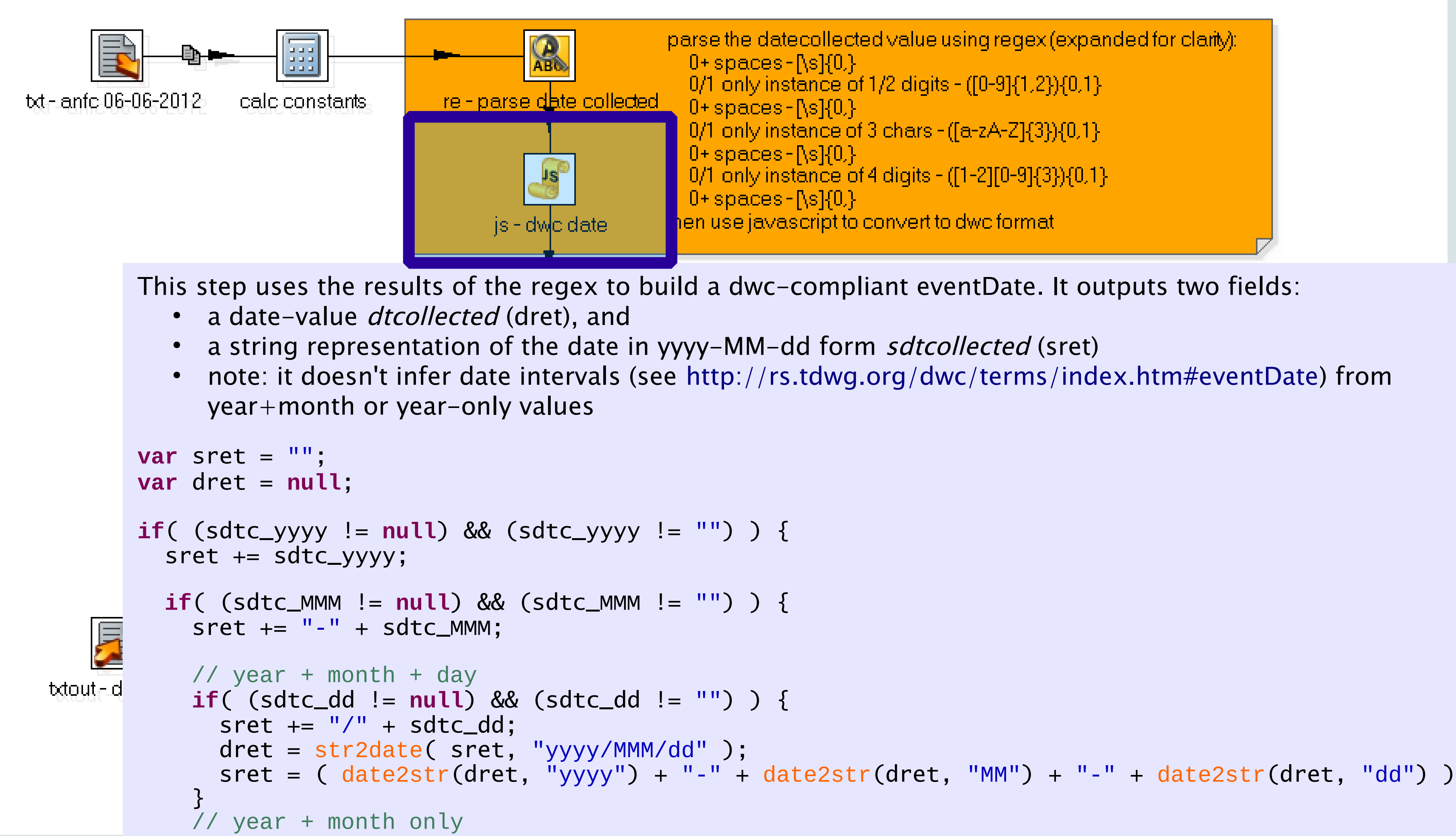
arse the datecollected value using regex (expanded for clarity):
0+ spaces - [\s]{0,}
0/1 only instance of 1/2 digits - ([0-9]{1,2}){0,1}
0+ spaces - [\s]{0,}
0/1 only instance of 3 chars - ([a-zA-Z]{3}){0,1}
0+ spaces - [\s]{0,}
0/1 only instance of 4 digits - ([1-2][0-9]{3}){0,1}
0+ spaces - [\s]{0,}
then use javascript to convert to dwc format

This step parses the *datecollected* field to fill corresponding date parts *sdtc_dd*, *sdtc_MMM* and *sdtc_yyyy*. Not every record will have a complete complement of *sdtc_xxx* so the regex is designed to populate one or more capture groups.

If the regex matches at least one component, *re_sdtc* will be "1" for this row.

```
[\s]{0,}([0-9]{1,2}){0,1}[\s]{0,}([a-zA-Z]{3}){0,1}[\s]{0,}([1-2][0-9]{3})
{0,1}[\s]{0,}
```

- `0+ spaces - [\s]{0,}`
- `0/1 only instance of 1/2 digits - ([0-9]{1,2}){0,1}`
- `0+ spaces - [\s]{0,}`
- `0/1 only instance of 3 chars - ([a-zA-Z]{3}){0,1}`
- `0+ spaces - [\s]{0,}`
- `0/1 only instance of 4 digits - ([1-2][0-9]{3}){0,1}`
- `0+ spaces - [\s]{0,}`



parse the datecollected value using regex (expanded for clarity):
  0+ spaces - [\s]{0,}
  0/1 only instance of 1/2 digits - ([0-9]{1,2}){0,1}
  0+ spaces - [\s]{0,}
  0/1 only instance of 3 chars - ([a-zA-Z]{3}){0,1}
  0+ spaces - [\s]{0,}
  0/1 only instance of 4 digits - ([1-2][0-9]{3}){0,1}
  0+ spaces - [\s]{0,}
hen use javascript to convert to dwc format

This step uses the results of the regex to build a dwc-compliant eventDate. It outputs two fields:

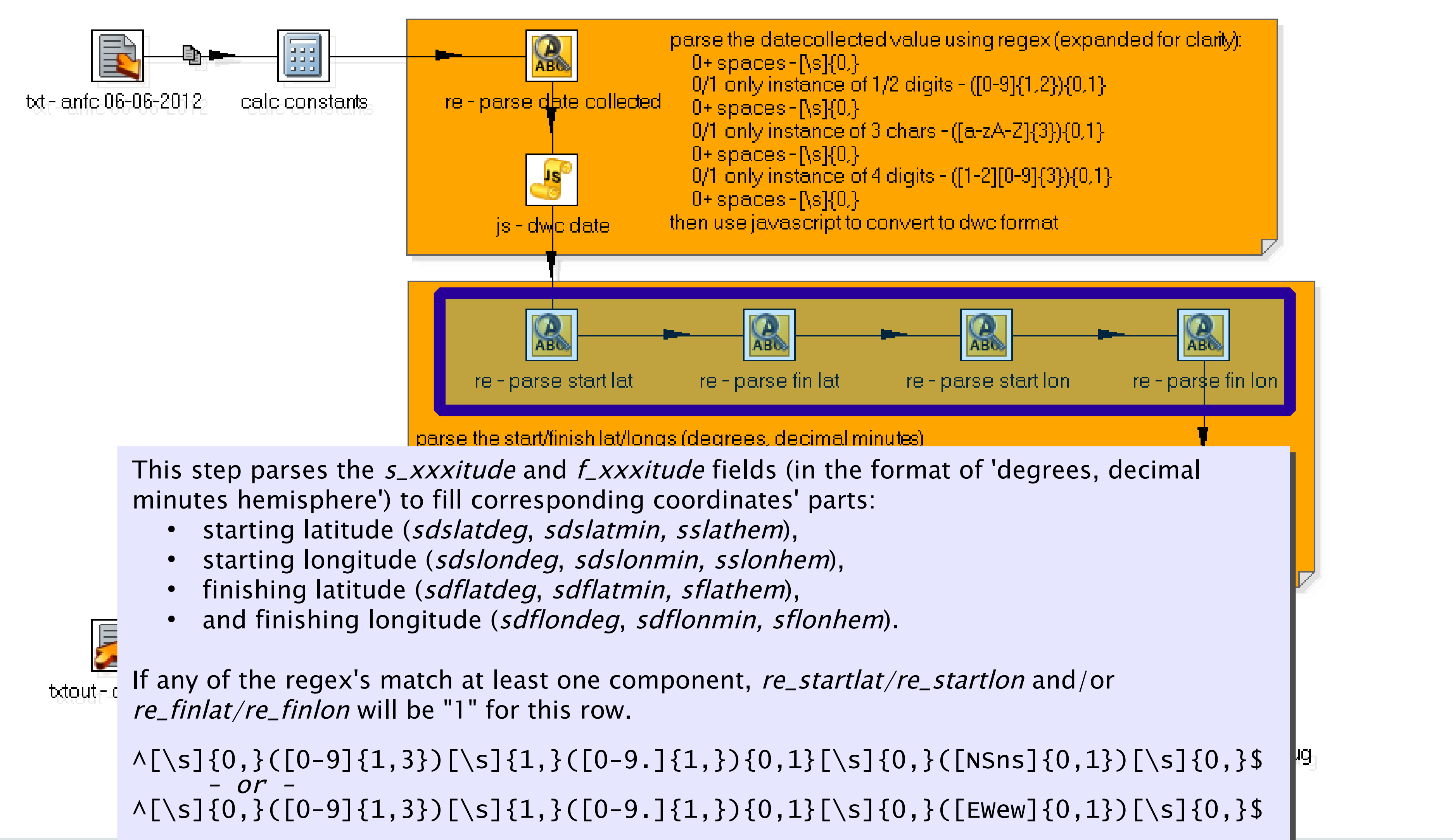
- a date-value *dtcollected* (dret), and
- a string representation of the date in yyyy-MM-dd form *sdtcollected* (sret)
- note: it doesn't infer date intervals (see http://rs.tdwg.org/dwc/terms/index.htm#eventDate) from year+month or year-only values

```
var sret = "";
var dret = null;

if( (sdtc_yyyy != null) && (sdtc_yyyy != "") ) {
  sret += sdtc_yyyy;

  if( (sdtc_MMM != null) && (sdtc_MMM != "") ) {
    sret += "-" + sdtc_MMM;

    // year + month + day
    if( (sdtc_dd != null) && (sdtc_dd != "") ) {
      sret += "/" + sdtc_dd;
      dret = str2date( sret, "yyyy/MMM/dd" );
      sret = ( date2str(dret, "yyyy") + "-" + date2str(dret, "MM") + "-" + date2str(dret, "dd") )
    }
    // year + month only
    else {
      dret = str2date( sret, "yyyy/MMM" );
      sret = ( date2str(dret, "yyyy") + "-" + date2str(dret, "MM") )
    }
  }
  // year only
  else {
    dret = str2date( sret, "yyyy" );
  }
}
```



parse the datecollected value using regex (expanded for clarity):

```
    0+ spaces - [\s]{0,}
    0/1 only instance of 1/2 digits - ([0-9]{1,2}){0,1}
    0+ spaces - [\s]{0,}
    0/1 only instance of 3 chars - ([a-zA-Z]{3}){0,1}
    0+ spaces - [\s]{0,}
    0/1 only instance of 4 digits - ([1-2][0-9]{3}){0,1}
    0+ spaces - [\s]{0,}
```

then use javascript to convert to dwc format

parse the start/finish lat/longs (degrees, decimal minutes)

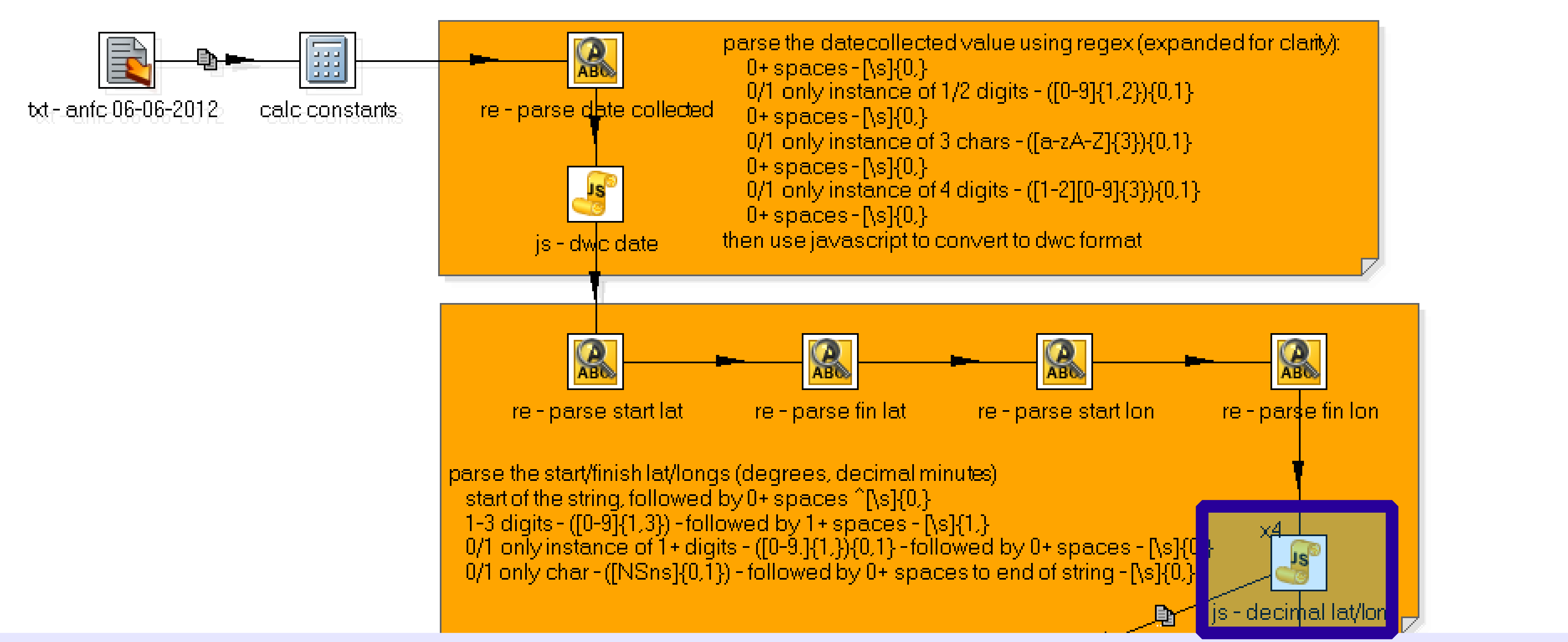
This step parses the *s_xxxitude* and *f_xxxitude* fields (in the format of 'degrees, decimal minutes hemisphere') to fill corresponding coordinates' parts:

- starting latitude (*sdslatdeg*, *sdslatmin, sslathem*),
- starting longitude (*sdslondeg*, *sdslonmin, sslonhem*),
- finishing latitude (*sdflatdeg*, *sdflatmin, sflathem*),
- and finishing longitude (*sdflondeg*, *sdflonmin, sflonhem*).

If any of the regex's match at least one component, *re_startlat/re_startlon* and/or *re_finlat/re_finlon* will be "1" for this row.

```
^[\s]{0,}([0-9]{1,3})[\s]{1,}([0-9.]{1,}){0,1}[\s]{0,}([NSns]{0,1})[\s]{0,}$
     – or –
^[\s]{0,}([0-9]{1,3})[\s]{1,}([0-9.]{1,}){0,1}[\s]{0,}([EWew]{0,1})[\s]{0,}$
```

- start of the string, followed by 0+ spaces ^[\s]{0,}
- 1-3 digits – ([0-9]{1,3}) – followed by 1+ spaces – [\s]{1,}
- 0/1 only instance of 1+ digits – ([0-9.]{1,}){0,1} – followed by 0+ spaces – [\s]{0,}
- 0/1 only char – ([NSns]{0,1}) – followed by 0+ spaces to end of string – [\s]{0,}$



```
parse the datecollected value using regex (expanded for clarity):
    0+ spaces - [\s]{0,}
    0/1 only instance of 1/2 digits - ([0-9]{1,2}){0,1}
    0+ spaces - [\s]{0,}
    0/1 only instance of 3 chars - ([a-zA-Z]{3}){0,1}
    0+ spaces - [\s]{0,}
    0/1 only instance of 4 digits - ([1-2][0-9]{3}){0,1}
    0+ spaces - [\s]{0,}
then use javascript to convert to dwc format
```

```
parse the start/finish lat/longs (degrees, decimal minutes)
   start of the string, followed by 0+ spaces ^[\s]{0,}
   1-3 digits - ([0-9]{1,3}) - followed by 1+ spaces - [\s]{1,}
   0/1 only instance of 1+ digits - ([0-9.]{1,}){0,1} - followed by 0+ spaces - [\s]{0,}
   0/1 only char - ([NSns]{0,1}) - followed by 0+ spaces to end of string - [\s]{0,}
```

This step uses the results from the regex's to convert parts of the *s_xxxitude* and *f_xxxitude* fields (in the format of 'degrees, decimal minutes hemisphere') to decimal degrees:

- string-format decimal latitude (*sdlat*) and decimal longitude (*sdlon*),
- finest precision of the coordinates, as a number between 0 and 1 (*sdprec*),
- finishing decimal lat-/longitude (*sdflondeg*, *sdflonmin, sflonhem*)

Not every record will have a complete complement of *sds.../sdf...* however, the regex will expect at least the corresponding *...deg* components for a lat/lon pair otherwise no output for that pair will be generated.

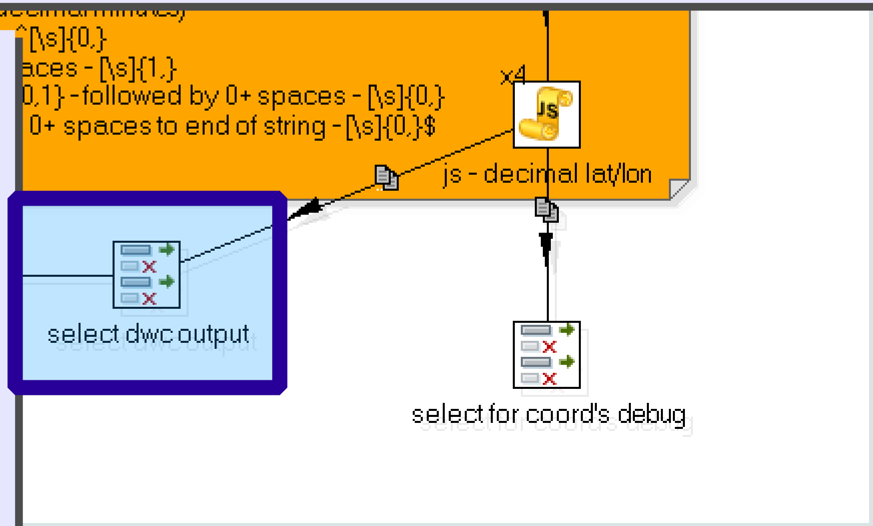
In the event where start and finish coordinates are matched, *sfootprintwkt* will also be generated, following the pattern: *LINESTRING(long/x lat/y, long/x lat/y)*.

source-code is available here:
http://code.google.com/p/ala-datamob/source/browse/trunk/artefacts/sourcecode/spatial/jConvertDD.js
http://code.google.com/p/ala-datamob/source/browse/trunk/artefacts/sourcecode/spatial/jParseDMS.js

This step filters out any data not bound for the export, and renames any added or derived fields to their appropriate dwc term. If a field is not listed in this step, it will not appear visible to the next step (which handles the output).

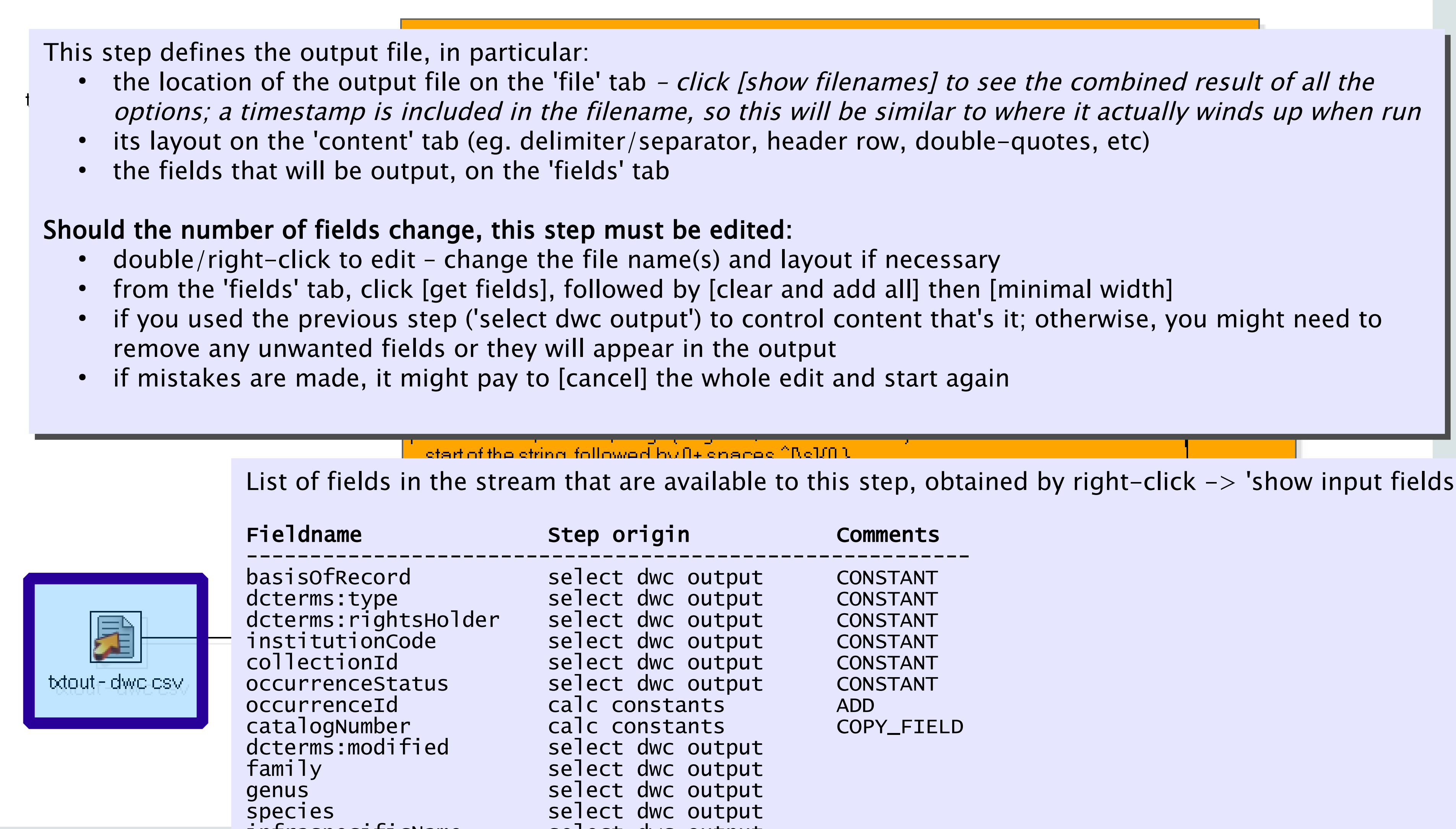
Should the number of fields change, this step must be edited:

- double/right-click to edit and move to the 'select & alter' tab (default)
- click [get fields to select]
- remember the current last field index, then click [add new]; *pentaho will now add every field that is available in the stream, but not currently listed*
- left-click the first new row, press and hold shift, then left-click the last new row and release shift; *you will now have all the new rows selected*
- next deselect any new fields you wish to *keep* by holding ctrl while left-click the desired field row
- finally, press the delete key to remove all rows not being kept; *you should now be left with the original rows, plus the new rows that you intend adding*
- if mistakes are made, it might pay to [cancel] the whole edit and start again



Abridged listing of fields included currently – *if 'renamed to' is blank, field is included without name being changed.*

Stream field	Renamed to
cnst_basisOfRecord	basisOfRecord
cnst_dcterms:type	dcterms:type
cnst_dcterms:rightsHolder	dcterms:rightsHolder
cnst_institutionCode	institutionCode
cnst_collectionId	collectionId
cnst_occurrenceStatus	occurrenceStatus
occurrenceId	
catalogNumber	
dateentered	dcterms:modified
Family	family
Genus	genus
Species	species
Subspecies	infraspecificName
scientificname	scientificName
verbatimLocality	
minimumdepth	
maximumdepth	
datecollected	verbatimEventDate
sdtcollected	eventDate
sdlat	decimalLatitude
sdlon	decimalLongitude
sdprec	coordinatePrecision
sfootprintwkt	footprintWKT
verbatimLatitude	
verbatimLongitude	



start of the string, followed by 0+ spaces ^[\s]{0,}

This step defines the output file, in particular:

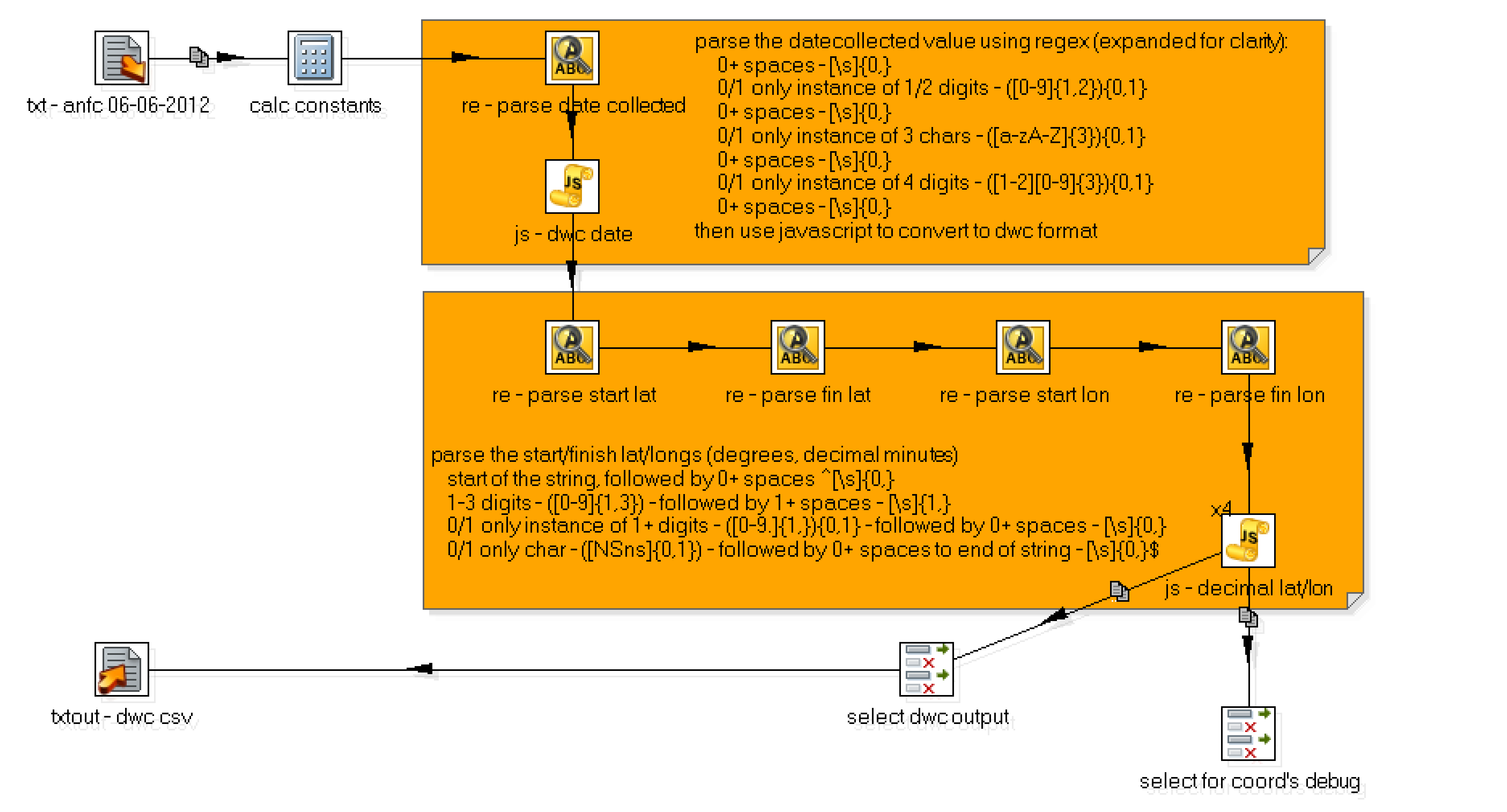
- the location of the output file on the 'file' tab – *click [show filenames] to see the combined result of all the options; a timestamp is included in the filename, so this will be similar to where it actually winds up when run*
- its layout on the 'content' tab (eg. delimiter/separator, header row, double-quotes, etc)
- the fields that will be output, on the 'fields' tab

Should the number of fields change, this step must be edited:

- double/right-click to edit – change the file name(s) and layout if necessary
- from the 'fields' tab, click [get fields], followed by [clear and add all] then [minimal width]
- if you used the previous step ('select dwc output') to control content that's it; otherwise, you might need to remove any unwanted fields or they will appear in the output
- if mistakes are made, it might pay to [cancel] the whole edit and start again

List of fields in the stream that are available to this step, obtained by right-click -> 'show input fields':

```
Fieldname                Step origin             Comments
----------------------------------------------------------
basisOfRecord            select dwc output       CONSTANT
dcterms:type             select dwc output       CONSTANT
dcterms:rightsHolder     select dwc output       CONSTANT
institutionCode          select dwc output       CONSTANT
collectionId             select dwc output       CONSTANT
occurrenceStatus         select dwc output       CONSTANT
occurrenceId             calc constants          ADD
catalogNumber            calc constants          COPY_FIELD
dcterms:modified         select dwc output
family                   select dwc output
genus                    select dwc output
species                  select dwc output
infraspecificName        select dwc output
scientificName           select dwc output
verbatimLocality         calc constants          ADD3
minimumdepth             txt - anfc 06-06-2012
maximumdepth             txt - anfc 06-06-2012
verbatimEventDate        select dwc output
eventDate                select dwc output
decimalLatitude          select dwc output
decimalLongitude         select dwc output
coordinatePrecision      select dwc output
footprintWKT             select dwc output
verbatimLatitude         calc constants          ADD
verbatimLongitude        calc constants          ADD
```



parse the datecollected value using regex (expanded for clarity):
    0+ spaces - [\s]{0,}
    0/1 only instance of 1/2 digits - ([0-9]{1,2}){0,1}
    0+ spaces - [\s]{0,}
    0/1 only instance of 3 chars - ([a-zA-Z]{3}){0,1}
    0+ spaces - [\s]{0,}
    0/1 only instance of 4 digits - ([1-2][0-9]{3}){0,1}
    0+ spaces - [\s]{0,}
then use javascript to convert to dwc format

parse the start/finish lat/longs (degrees, decimal minutes)
  start of the string, followed by 0+ spaces ^[\s]{0,}
  1-3 digits - ([0-9]{1,3}) - followed by 1+ spaces - [\s]{1,}
  0/1 only instance of 1+ digits - ([0-9.]{1,}){0,1} - followed by 0+ spaces - [\s]{0,}
  0/1 only char - ([NSns]{0,1}) - followed by 0+ spaces to end of string - [\s]{0,}$

You are now ready to re-run the transformation:

- first, save the transformation (File->save)
- next, run the transformation (Action->run)
- recommend you set logging level to 'detailed logging', now click [launch]
- at 30k records, the whole process should complete in under a minute
- at the bottom of the screen, a watch window will appear – 'logging' contains the textual execution log, and 'step metrics' displays a tabular state listing each step, and the number of records that have passed
- any errors will be highlighted in red in both windows, and on the transformation pane, the problematic step will have a red border drawn around it