GangByHierarchy Version 1

Description

GangByHierarchy creates collections or gangs of jobs based on the combination of job hierarchy values and an accumulated count of PDF pages or sheets, or a configured job property. You will configure a desired gang size along with a maximum gang size. When enough jobs are accumulated, the jobs will be moved to a single job folder. Optionally, metadata can be created in XML and/or JSON format. Additionally, a "flush" schedule can be configured that will trigger the creation of gangs with all currently waiting jobs or move the waiting jobs to a *redirection connection*.

Additional functionality:

- Jobs are processed based on a configured sorting method that can utilize a combination of each job's priority, sorting date and configured sorting key.
- In the case of an individual job exceeding the configured gang size, several options are available to handle these jobs – *see below*.
- Because all the jobs are copied into a new job folder, any jobs with duplicate names would overwrite each other. Several options are available to handle duplicate jobs – *see below*.
- By default, the metadata includes basic information about each gang. The app provides properties to allow the addition of custom gang and job information.
- The redirection of jobs when flushed can further be controlled by a *threshold* value. Below the threshold, jobs are redirected; above the threshold, a "short" gang is created with the remaining jobs in the queue.

Compatibility

Switch 2021 Spring and higher.

Connections

At least one incoming connection required. Each connection should originate from a folder that is immediately downstream from a Submit Hierarchy element. The Submit Hierarchy should mirror the Archive Hierarchy into which the jobs were sorted. Four different outgoing connections can be configured: *move gangs, route duplicate jobs, route over-sized jobs* and *redirect jobs on flush*.

Properties detailed info

Gang name prefix
The gang folder will use this value with a unique 32 character id appended to it. This same unique id is included in the gang metadata. The unique id is created from an md5 hash of the system host name, connection id, the combination of item values in the hierarchy and a time stamp and so should be unique within a set of Switch servers in the same domain.

Gang minimum size
This is the minimum size used for the accumulation of jobs. When this value is reached, a gang will be created. *Note that during a "flush" or in the case of a job being removed from the incoming connection without stopping the flow first, it is possible that a "short" gang could be created.*

Maximum gang size
No gangs will be created with an accumulated total exceeding this value. Exception: if you tell the app to *gang oversized job by itself*, a gang exceeding this value could be produced.

Count unit
Select from *job*, *page*, or *sheet*. The options *page* and *sheet* are for PDFs only. If *sheet* is selected, the value will be set to the equivalent number of two-sided sheets in the PDF. For example, a PDF with 11 *pages* would result in a *sheet* count of 6. When *job* is specified, you must configure how this value is calculated. For example, if your imposition had 12 positions to fill, your calculation could be the number of positions each individual job would require based on quantity needed.

Sorting method
Select the sorting method to be applied to the queue of jobs. Jobs will be added to a gang in this order. Depending on your choice, additional properties will be made available to configure for each job. The values are sorted in ascending order.

Sorting properties…

Job priority
Configure a *numeric* value which will be used to sort the jobs in ascending order. If your metadata contains textual values, you will need to convert these values to a number.

Example script expression:

```
var usePriority;
var workflowXml = job.getDataset("WorkflowXml");
var priorityStr = workflowXml.evalToString( '/dn:JobWorkflow/dn:Priority' );

switch ( priorityStr.toLowerCase() ) {
    case "rush":
        usePriority = 1;
        break;
    case "expedited":
        usePriority = 2;
    break;
    case "normal":
        usePriority = 3;
        break;
}
usePriority;
```

Sort date-time
Configure a date-time value used to sort the waiting jobs. The default is the current system date-time. This should be a valid ISO 8601 formatted date or date-time, examples 2022-02-18 or 2022-02-18T15:30:00.

Sort key
Configure a string from available metadata. *Note that if you include numeric values in the sort string, you may want to pad them with zeros as a string like "job195" would come before "job2" when sorted. Padding the number would fix this as "job002" would come before "job195".*

Attach gang information?
Set this to *yes* to configure custom metadata for the gang.

Custom gang information
This property is evaluated in *timerFired* and so no variables are permitted. Use the *multi-line text editor* to create *key:value pairs* of information with each pair on it's own line. In the resuling XML metadata, each pair will be created as a `<key>value</key>` element contained within an **`<extrinsics>`** element in the **`<ganginformation>`** element. For JSON, each pair will be created as an object value within an **`extrinsics`** object contained in the **`ganginformation`** object.
(See `metadata` examples at the end of this document.)

Attach job information?
Set to *yes* to include additional job metadata.

Job information
You may use either the *multi-line text* or *mult-line text with variables* editor. Place each *key:value* pair on a separate line. In the resulting XML metadata, each pair will be created as a `<key>value</key>` element contained within an `<extrinsics>` element in each **`<jobs>`** element. For JSON, each pair will be created as an object value within an `extrinsics` object contained in each individual object in the `jobs` array.

Include XML metadata?
Set to yes to create an XML based metadata dataset. Indicate the name of the dataset in the *XML dataset name* property.

Include JSON metadata?
Set to yes to create a JSON based metadata dataset. Indicate the name of the dataset in the *JSON dataset name* property.

Large job handling
If a single job comes in with a unit count exceeding the minimum gang size, you can handle the job as follows:

- Gang the job by itself
- Always send the job to the *oversized jobs* connection
- Only send the job to the *oversized jobs* connection if it also exceeds the maximum gang size

Duplicate job handling
An incoming job with the same name can be an issue if both jobs are later ganged together as one job will overwrite the other job. You can handle duplicate jobs with one of these options:

- Keep the job and add a sequence number as a suffix to the jobname
- Keep oldest by arrival date - send duplicate to duplicate jobs connection
- Keep oldest by arrival date - delete duplicate
- Keep newest by arrival date - send duplicate to duplicate jobs connection
- Keep newest by arrival date - delete duplicate
- Keep oldest by sort date - send duplicate to duplicate jobs connection
- Keep oldest by sort date - delete duplicate

- Keep newest by sort date - send duplicate to duplicate jobs connection
- Keep newest by sort date - delete duplicate

Flush jobs?
Choose between the following options:
- Do not flush jobs
- Flush jobs – create gang folder *(these will be “short” gangs)*
- Flush jobs – redirect jobs

Redirection threshold
Use this setting to control whether the flushed jobs are included in a “short” gang or are sent to a *redirection connection* so that they can be processed by a different workflow. A value of 0 *(zero)* will result in the jobs always being routed to the redirection connection, otherwise the jobs will be redirected only if the gang unit count is less than the threshold value. If the gang unit count meets or exceeds the threshold, the jobs are placed in a “short” gang. *See the flow example for ganging sets of business cards that can be ordered in quantities of 250 and 500. It uses this feature to “cascade” jobs through a series of this app with each configured with different count combinations to find a “best fit”.*

Flush schedule
This is a multi-line text field with each day-time designation placed on a separate line. Each value should be in the form *day@time* (ex. MON@13:30 to releases jobs every Monday at 1:30PM). Use these day abbreviations: MON, TUE, WED, THU, FRI, SAT, SUN *(not case sensitive)*

Reset flush date-time on restart
If set to *Yes,* this instructs the app, on a flow restart, to reset the currently saved flush date-time to the next available date-time if the current date-time has passed. This may be desirable if you don’t want the jobs to be flushed immediately when restarting the flow after the current date-time has passed.

Recheck interval (minutes)
Set this to the time interval between checks to see if a gang can be created.

Check for orphan jobs interval (minutes)
Jobs that are removed manually from the incoming connections become “orphans” as they exist in the app’s job information list but are no longer physically available. The app will check at this interval to see if any orphans need to be cleaned up. This should probably be set at 30 to 60 minutes.

Log internals
Set this to *Yes* to have the app log the internal job list and potential gang entries to the *Info* log. Use this to check to make sure your job and gang properties are properly configured.

Note that you should stop a flow before manually removing files from an incoming connection. If not, it is possible that a non-existent job could be included in the ganging count accumulation and result in the creation of a gang that is short of the minimum gang size. However, the metadata and actual files included in the gang folder will be valid. A flag in the gang metadata will identify these gangs as flushed.

Example metadata:

XML

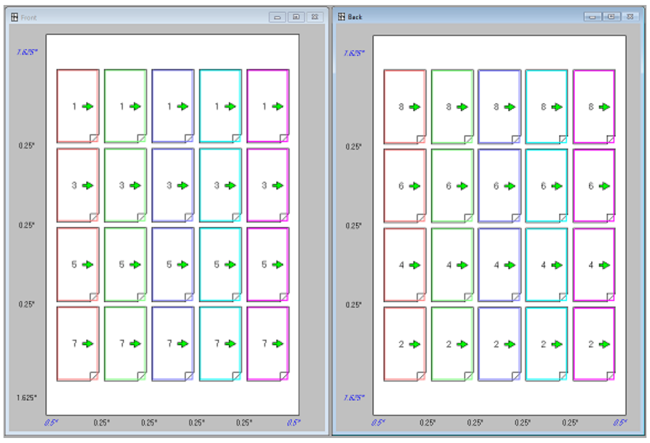
```
<gang>
   <ganginformation>
      <name>Print:Sandshell:5x7:squarecorner</name>
      <uniqueid>20975ac3e6b2cee237da083fb6753251</uniqueid>
      <jobcount>2</jobcount>
      <unit>sheet</unit>
      <unitcount>925</unitcount>
      <minimumgangsize>1500<minimumgangsize>
      <maximumgangsize>2000</maximumgangsize>
      <flushed>false</flushed>
      <heirarchy>Print</heirarchy>
      <heirarchy>Sandshell</heirarchy>
      <heirarchy>5.00x7.00</heirarchy>
      <heirarchy>squarecorner</heirarchy>
      <extrinsics>
         <partnerid>AP</partnerid>
         <partnername>ACME Printing</partnername>
         <producttype>invitations</producttype>
         <workflow>Indigo7000</workflow
      </extrinsics>
   </ganginformation>
   <jobs>
      <jobname>00020920 Flatcard+5x7+Sandshell+squarecorner@405.pdf</jobname>
      <priority>1</priority>
      <sortkey>00020920-607234301-1</sortkey>
      <count>405</count>
      <date>2022-11-02T15:52:05.650-04:00</date>
      <extrinsics>
         <OrderID>00020920</OrderID>
         <LineItemList>c43a36eb-724f-49a7-98dc-af557f12f9f9</LineItemList>
         <ShipBy>2022-11-04T16:30:00</ShipBy>
      </extrinsics>
   </jobs>
   <jobs>
      <jobname>00021178 Flatcard+5x7+Sandshell+squarecorner@520.pdf</jobname>
      <priority>1</priority>
      <sortkey>00021178-612896601-1-1</sortkey>
      <count>520</count>
      <date>2022-11-04T01:45:06.540-04:00</date>
      <extrinsics>
         <OrderID>00021178</OrderID>
         <LineItemList>64ddb6d0-945f-416a-8fe1-08db0646b2d0</LineItemList>
         <ShipBy>2022-11-04T16:30:00</ShipBy>
      </extrinsics>
   </jobs>
</gang>
```

JSON

```
{
  "ganginformation": {
    "name": "Print:Sandshell:5x7:squarecorner",
    "uniqueid": "20975ac3e6b2cee237da083fb6753251",
    "jobcount": 2,
    "unit": "sheet",
    "unitcount": 925,
    "minimumgangsize":1500,
    "maximumgangsize":2000,
    "flushed": false,
    "heirarchy": [
      "Print",
      "Sandshell",
      "5x7",
      "squarecorner"
    ],
    "extrinsics": {
      "partnerid": "AP",
      "partnername": " ACME Printing",
      "producttype": "invitations",
      "workflow": "Indigo7000"
    }
  },
  "jobs": [
    {
      "jobname": "00020920 Flatcard+5x7+Sandshell+normal@405.pdf",
      "priority": 1,
      "sortkey": "00020920-607234301-1-1",
      "count": 405,
      "date": "2022-11-02T15:52:05.650-04:00",
      "extrinsics": {
        "OrderID": "00020920",
        "LineItemList": "c43a36eb-724f-49a7-98dc-af557f12f9f9",
        "ShipBy": "2022-11-04T16:30:00"
      }
    },
    {
      "jobname": "00021178 Flatcard+5x7+Sandshell+ normal@520.pdf",
      "priority": 1,
      "sortkey": "00021178-612896601-1-1",
      "count": 520,
      "date": "2022-11-04T01:45:06.540-04:00",
      "extrinsics": {
        "OrderID": "00021178",
        "LineItemList": "64ddb6d0-945f-416a-8fe1-08db0646b2d0",
        "ShipBy": "2022-11-04T16:30:00"
      }
    }
  ]
}
```

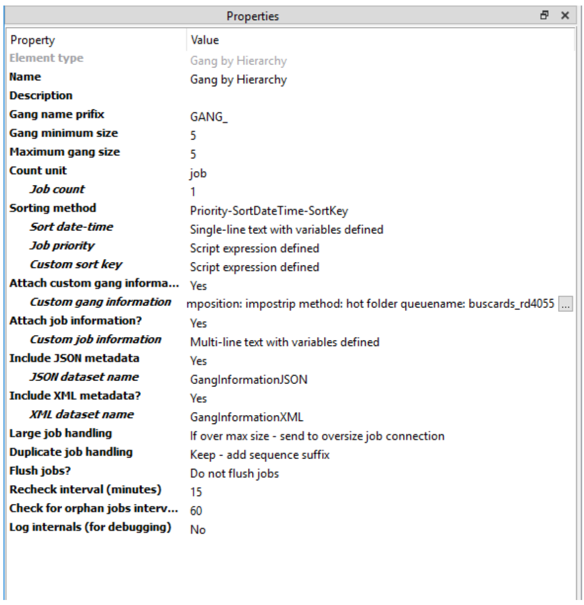
Ganging Examples

Requirement: Gang duplex business cards on 12x18 sheet for cutting on an RD-4055. Each job is a multi-page PDF consisting of a header card, trailer card, re-order insert and 100 cards + additional filler cards. These will be imposed by *Impostrip* using their *Ribbon cut and stack* imposition.



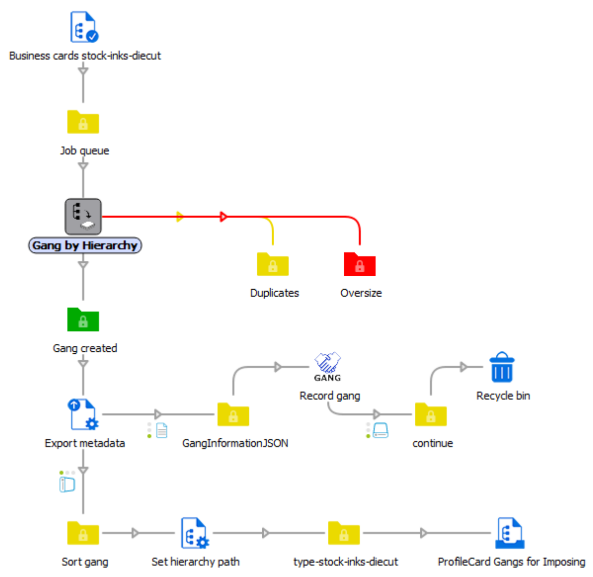
We can gang exactly 5 jobs together so we'll set both the Gang *minimum size* and *maximum* size to 5. The *sort date* is set to the job's ship-by-date. *The sort key* is set to the job's order number plus a set number so that jobs in the same order are printed together. The JSON metadata will be exported and used to register the gang in the database. The hierarchy for these cards is:

size + stock + ink_profile + diecut
(ink_profile can be 4color or 4color+whiteink)



Properties

Property	Value
Element type	Gang by Hierarchy
Name	Gang by Hierarchy
Description	
Gang name prifix	GANG_
Gang minimum size	5
Maximum gang size	5
Count unit	job
Job count	1
Sorting method	Priority-SortDateTime-SortKey
Sort date-time	Single-line text with variables defined
Job priority	Script expression defined
Custom sort key	Script expression defined
Attach custom gang informa...	Yes
Custom gang information	mposition: impostrip method: hot folder queuename: buscards_rd4055
Attach job information?	Yes
Custom job information	Multi-line text with variables defined
Include JSON metadata	Yes
JSON dataset name	GangInformationJSON
Include XML metadata?	Yes
XML dataset name	GangInformationXML
Large job handling	If over max size - send to oversize job connection
Duplicate job handling	Keep - add sequence suffix
Flush jobs?	Do not flush jobs
Recheck interval (minutes)	15
Check for orphan jobs interv...	60
Log internals (for debugging)	No

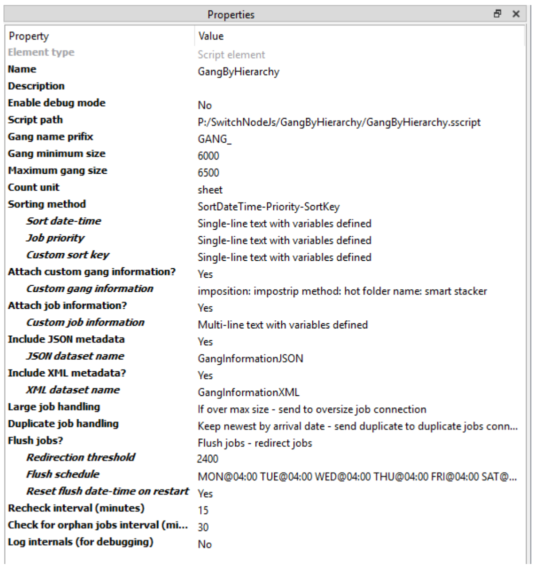


Requirement: Gang high volumes of duplexed card orders that vary in quantities from 1 to thousands (average is around 60 cards) for printing on 29.5 x 20.75 and target approximately 500 imposed sheets. Our press is an Indigo 10k and will be cut down on a SmartStacker unless the number of imposed sheets is less than 200 when we flush the jobs.

This workflow is for 5x7 cards running 12up. Our target gang size is 500 x 12 = 6000 5x7 sheets with a threshold of an additional 50 imposed sheets. We'll set the Gang *minimum size* to 6000 and the *maximum* size to 6600. The *sort date* is set to the job's ship-by-date. *The sort key* is set to the job's order number so that jobs in the same order are printed together. The hierarchy for these cards is:

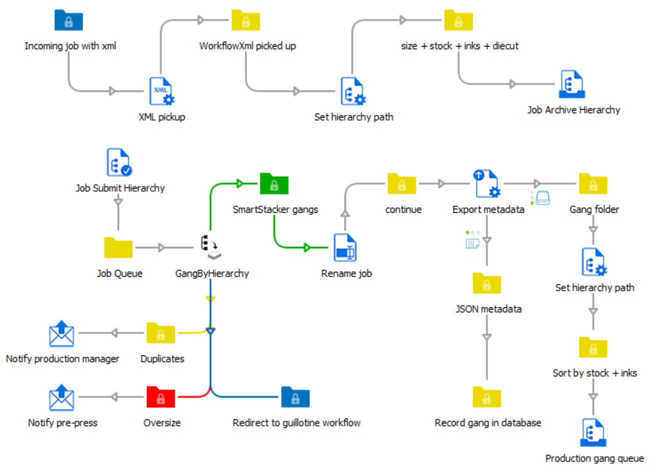
size + stock + ink_profile + diecut
(ink_profile can be 4color or 6color)

Any jobs remaining in the queue will be flushed at 4am. The flush setting is set to *redirect jobs* based on a *threshold of 2400* (2400 / 12 = 200 imposed sheets). So, if a gang results in less than 200 imposed sheets, we don't want to utilize the SmartStacker. Instead, we'll redirect these jobs to a new workflow that gangs these jobs using a cut-and-stack imposition so that they can be cut on a guillotine.



Properties

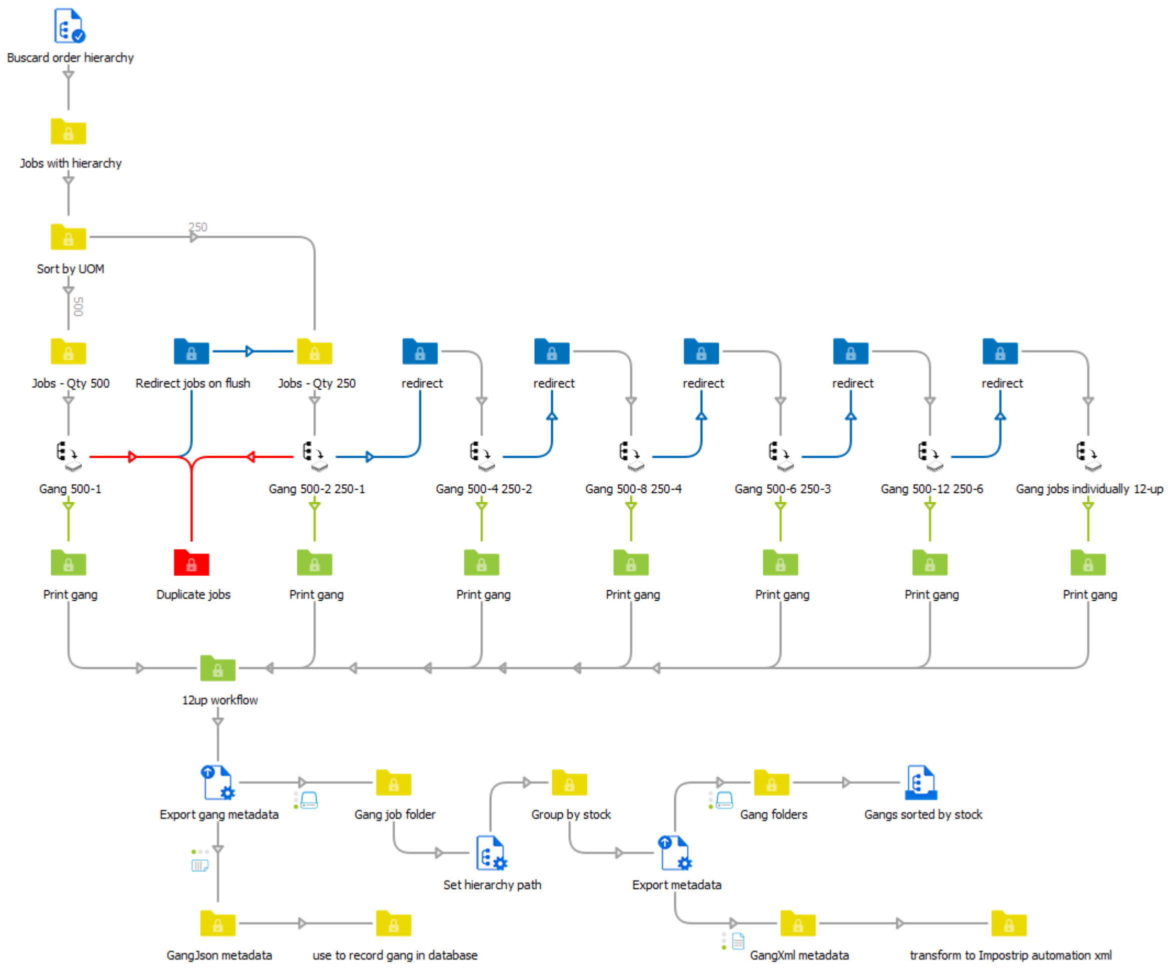
Property	Value
Element type	Script element
Name	GangByHierarchy
Description	
Enable debug mode	No
Script path	P:/SwitchNodeJs/GangByHierarchy/GangByHierarchy.sscript
Gang name prifix	GANG_
Gang minimum size	6000
Maximum gang size	6500
Count unit	sheet
Sorting method	SortDateTime-Priority-SortKey
Sort date-time	Single-line text with variables defined
Job priority	Single-line text with variables defined
Custom sort key	Single-line text with variables defined
Attach custom gang information?	Yes
Custom gang information	imposition: impostrip method: hot folder name: smart stacker
Attach job information?	Yes
Custom job information	Multi-line text with variables defined
Include JSON metadata	Yes
JSON dataset name	GangInformationJSON
Include XML metadata?	Yes
XML dataset name	GangInformationXML
Large job handling	If over max size - send to oversize job connection
Duplicate job handling	Keep newest by arrival date - send duplicate to duplicate jobs conn...
Flush jobs?	Flush jobs - redirect jobs
Redirection threshold	2400
Flush schedule	MON@04:00 TUE@04:00 WED@04:00 THU@04:00 FRI@04:00 SAT@...
Reset flush date-time on restart	Yes
Recheck interval (minutes)	15
Check for orphan jobs interval (mi...	30
Log internals (for debugging)	No



Requirement: Gang orders of 250 and 500 business cards in varying quantities for printing 12up on 8.5x11 stock. Each business card order can be on one of 8 stocks, specify square or rounded corners and can be printed with or without white ink requiring this hierarchy:

stock + die cut + white_ink/no_white_ink (32 combinations)

In this workflow, we first try to gang orders of 500 and 250 by themselves. At the end of the day, we begin to "flush" jobs from left to right by redirecting any remaining jobs downstream to the next app instance. Each new instance of the app is configured to apply different count combinations to *best fit* remaining jobs into a gang. Finally, any jobs still remaining are ganged by themselves. The gang and job JSON metadata is then used to record each gang and what orders it includes in a database. The XML metadata utilizes the Saxon tool to transform the metadata into a redirection xml document for use with Impostrip Automation to impose the jobs in each gang.


Buscard order hierarchy
Jobs with hierarchy
Sort by UOM
250
500
Jobs - Qty 500
Redirect jobs on flush
Jobs - Qty 250
redirect
redirect
redirect
redirect
redirect
Gang 500-1
Gang 500-2 250-1
Gang 500-4 250-2
Gang 500-8 250-4
Gang 500-6 250-3
Gang 500-12 250-6
Gang jobs individually 12-up
Print gang
Duplicate jobs
Print gang
Print gang
Print gang
Print gang
Print gang
Print gang
12up workflow
Export gang metadata
Gang job folder
Set hierarchy path
Group by stock
Export metadata
Gang folders
Gangs sorted by stock
GangJson metadata
use to record gang in database
GangXml metadata
transform to Impostrip automation xml