AppWorks Export/Import tool

High level requirements

* Connectivity
  + The tool must be able to make multiple connections to the AppWorks environment (use the SOA layer)
  + All the connections need to be released/disconnected once we’re done with the tool
* Export
  + The tool needs to export metadata to an XML structure (based on an XML schema)
  + The tool needs to export content to a local drive/network share
  + A relation must be available between the exported content and metadata
  + Per batch size
* Import
  + The tool needs to create (= import) new data based on XML metadata
  + Related content needs to be uploaded with the new created data
  + Per batch size
* Configurability
  + The tool must be scalable (by configuration) so multiple connections can be made to the environment
  + The connection string must be configurable from a properties file
  + It must be configurable what entities can be exported
  + It must also be configurable what metadata must be exported
  + There must be an option to include/exclude related content
  + Configure how the mapping is applied from XML data to the AppWorks entity data (and vice versa)
  + There must be a dry-run option with only logging (so you can make an impact decision)
  + The level of logging/monitoring must be set
  + Mail service information must be available
  + Failures allowed in a row
  + Batch size
* Monitoring
  + The tool needs to log what was done (in several levels)
  + A mail notification will be sent out when done
* Failure
  + When the backend system shuts down the tool must stop exporting and finish the current workload
  + When you fail on the export the current item will be logged and skipped so the next item can be picked up
  + When you fail on the import the current item will be ‘rolled back’, log it and skip so the next item that can be picked up
  + On X failures the tool will stop processing
  + When configuration is not properly done
* UI
  + Selection of the action (export/import) to be executed
  + START button to execute the action
  + ‘Know’ about the current status of the current action done
  + CANCEL button to stop the action so the current item will be finished, and the tool stops rest of the action. Tool is not closed; you go back to the initial start screen.
  + The end-user needs to be able to CLOSE/EXIT the tool (current running action will be cancelled!)
  + The CANCEL action will give an extra confirmation message