Digitization Workflow Management System for Massive Digitization Projects

Mohamed Yakout[*]
Email: mohamed.yakout@bibalex.org

Noha Adly[*†]
Email: noha.adly@bibalex.org

Magdy Nagi[*†]
Email: magdy.nagi@bibalex.org

[*]Bibliotheca Alexandrina, El Shatby 21526, Alexandria, Egypt

[†]Computer and Systems Engineering Department, Faculty of Engineering, Alexandria University, Alexandria, Egypt

***Abstract* — The Digitization Workflow Management System (DWMS) is a system developed in Bibliotheca Alexandrina, the Library of Alexandria, to manage the whole process of digitization including its various phases, system users, files movement, archiving, and integration with the ILS and the library digital repository. The system supports workflow dynamic evolutions and deviation to allow for exception handling. It provides history tracking of actions and flexibility to simultaneously manage multiple projects with a diversity of materials. Moreover, it supports ingesting a job in the middle of the workflow and allows easy integration of tools used to perform functions of the workflow.**

***Index Terms* — Digitization, Workflow, Digital Library.**

I. Introduction

When an end user accesses images, PDF, audio, video, or any other multimedia document through the Internet, this means that the primary purpose of the entire digitization effort is met. From start to finish, digital multimedia documents production is a manufacturing and delivery process which should be likened to an assembly line. To date, the digitization process in many libraries has concentrated on the input (scanning) side, and fulfillment via the Internet and Content Management Software. What has not received sufficient attention is automating, tracking, and managing the entire workflow with particular emphasis on what happens between scanning and delivery.

BA accepted this challenge as a part of its Digital Assets Repository (DAR) project to achieve a truly device-independent, integrated and automated workflow. The result was the Digitization Workflow Management System (DWMS), which is a high reliable digitization workflow management that can be customized for large and challenging digitization projects or used out-of-the-box. In either case, BA workflow system improves productivity and therefore reduces both the cost of production and the time it takes to complete a project.

A Digitization Laboratory requires an efficient and highly integrated digitization system consisting of hardware, software and workflow management processes that could fully exploit the unique capabilities of the Digital Lab. Several experiences with large digitization projects taught us the need for a highly integrated system that manages the whole process of digitization with its phases, system users, exception handling, history tracking of actions, files movement, archiving, and integration with the LIS and the library digital repository. Such system would need to be flexible enough to simultaneously manage multiple projects with a diversity of materials covering books, journals, newspapers, manuscripts, unbound materials, audio, video, and slides. The system would also need to seamlessly feed content to the libraries' digital repository to ensure the preservation of the content for years to come.

As any workflow, the digitization workflow is a description of a business process in sufficient details that it is able to be directly executed by a workflow management system [1]. A digitization workflow is composed of a sequence of phases. The phases are undertaken by the digital lab resources, such as digital lab operators and devices (scanners or encoding servers). Production data are information objects. For instance, TIFF, PDF, DJVU, ZIP files or any digital files, whose existence does not depend on workflow management. Several tools are used to execute elementary activities within a digitization phase such as, image processing and OCR suits.

Workflow management systems are used to configure and control structured business processes from which well-defined workflow models and instances can be derived [2, 3]. However, the proprietary process definition frameworks imposed make it difficult to support (i) dynamic evolution (i.e. modifying process definitions during execution) following unexpected or developmental change in the business processes being modeled [4]; and (ii) deviations from the prescribed process model at runtime [5, 6, 7]. Without support for dynamic evolution, the occurrence of a process deviation requires either suspension of execution while the deviation is handled manually, or an entire process abort. However, since most processes are long and complex, neither manual intervention nor process termination are satisfactory solutions [8]. Manual handling incurs an added penalty: the corrective actions undertaken are not added to the system history or transaction log [9, 10], and so natural process evolution is not incorporated into future iterations of the process. Other evolution issues include problems of migration, synchronization and version control [5, 11].

From our experiences, the digitization process is one of the business processes, which is affected by the above limitations. These limitations make it hard to be mapped to a rigid modeling structure [12], due to the lack of flexibility inherent in a framework that, by definition, imposes rigidity. As a result, users are forced to work outside of the system, and/or constantly revise the static process model, in order to successfully support their activities, thereby negating the efficiency gains sought by implementing a workflow solution in the first place. It is therefore desirable to extend the capabilities of workflow systems by developing an approach to dynamic flexibility based on natural work practices [15].

DWMS avoided the above limitations by providing the facility to design a rigid workflow and allow for the dynamic evolution and deviations. This helped in handling exceptions by forwarding the jobs to an appropriate phase, which is not in the rigid defined sequence without the manual intervention. For example when digitizing books, a printed book passes by scanning, processing, OCRing, archiving then encoding to PDF. While for a hand written book, it could not go to the OCR phase and hence should be forwarded to the archiving then encoding phase. Also the jobs can be redirected back to a previous phase due to a quality assurance decision.

The rest of the paper is organized as follow; section II gives an overview on the related work in the digitization workflow systems. Then, the data model of the DWMS will be discussed in section III followed by a description for the system architecture in section IV. In section V, the life cycle of a job will be described then in section VI implementation details will be discussed. Finally, section VII presents a conclusion and future work.

II. RELATED WORK

In this section, we will provide an overview on the digitization workflow systems efforts and directions. Currently, there are three directions followed by most of the existing digitization labs; (1) Manual workflow management using several software packages, (2) Simple tracking workflow system with limited capabilities, and (3) several integrated digitization activities in one software application to perform all the digitization phases.

The manual workflow management is performed using tools such as Excel spread sheets, which could be adequate when the number of digital lab operators is small and digitization projects are also small. When the number of operators increases, some labs use in addition to Excel sheets tools such as Microsoft Share Point and Microsoft Project for managing larger digitization projects.

Other labs may have the ability to hire programmers to develop a customized tracking system for their digitization workflow. In this case, the developed software deals with a rigid defined workflow, where handling exceptions is done manually and a very limited history tracking. The developed system depends on the lab environment and other software tools used within the digitization process. If there is files handling and checking, it will be related to the currently used tools, but the if tools changed the system will need extra programming and development.

In addition to the above directions, several commercial software companies tried to develop digitization workflow systems by integrating several activities. For example, digital capturing, image processing and OCRing in one software.

DOCWorks from CCS [16] integrates several useful activities for digitizing textual documents as books and newspapers. It provides tools for image processing, layout analysis, OCR and metadata extraction. It depends on the OCR software to analyze and extract metadata from a document. If the OCR results are poor, then the software acts as an image processor and PDF encoder. DOCWorks is limited to textual documents only.

BookRestorer from i2s [17] integrates the whole process to digitize a printed book until it is encoded in PDF. It allows for the integration with Photoshop and the currently installed OCR software.

OUPS from the Academic Imaging Associates [18] integrates the image capturing by supporting various scanners, image processing tools, OCRing and metadata extraction. It also provides metadata templates that can be customized.

The above mentioned software tools can be a step or a phase in a bigger digitization workflow system. However, they suffer from several limitation. First, they are tightly coupled with certain tools and do not allow easily other tools to be integrated. For example, the OCR engine in DOCWorks is not working well for the Arabic, and the image processing tools in BookRestorer do not produce the best quality. Second, the systems do not manage the digital lab resources such as *Workstations* and users. Also they lack the management of projects and collections. Third, all the files handling between the storage server and clients is done manually. Finally, the systems lack the handling of workflow exceptions, and do not allow for dynamic evolution and deviations except through manual intervention.

III. SYSTEM DATA MODEL

To achieve a truly device-independent, integrated and automated digitization workflow, the system introduces a data model capable of defining different workflows for various types of objects. Each type of object can have its workflow defined by what we call *Phase Sequence*.

The data model diagram as shown in Fig. 1., consists of six types of entities involved in managing the digitization workflow.

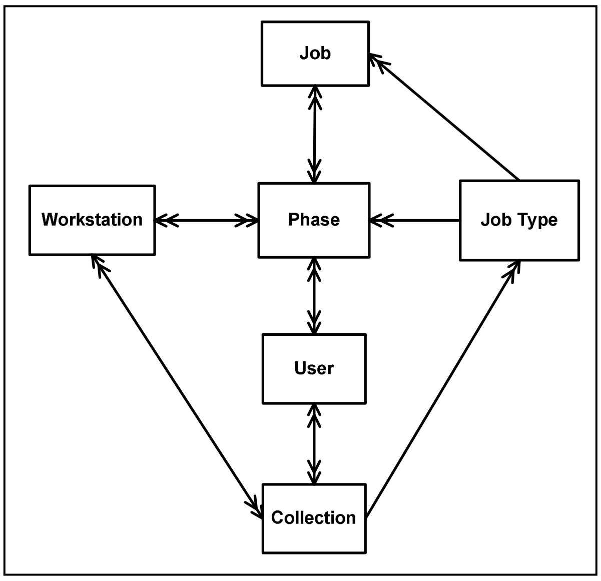

Fig. 1. BA Workflow tracking system data model.

A. The Job

It is the main entity, which represents the object being digitized. For example, a printed book for Naguib Mahfouz, photos for an event, a map of Alexandria, a music sheet for Omar Khayrat, a video film about the High Dam, etc. The *Job* can be one of any *Job Type* in the system. The *Job* should pass through the entire digitization *Phases* required for the *Job Type*. Each *Job* is identified by a unique ID and also can be identified by an External ID, which is the ID of the document in the external source as the ILS ID and/or Barcode. The *Job* has a priority and a life time in the workflow otherwise it will be reported as a *Late Job*.

B. The Job Type

The *Job Type* entity represents all the types of materials that can be digitized. The *Job Type* can be book, map, audio, video, or any other type of document that needs a special digitization workflow.

C. The Phase

The *Phase* entity represents a task or a unit of work that should be applied on a specific *Job Type* in its digitization workflow. Each *Job Type* has its own sequence of *Phases* defined apriori in a *Phase Sequence*, to obtain a digitized version of the *Job*. Each *Job* passes through several *Phases* according to its *Job Type*. For example, for a Printed Book *Job Type*, the digitization *Phases* could be Scanning, Processing, OCRing, Archiving and PDF Encoding. While for Maps *Job Type*, the digitization *Phases* could be Scanning, Processing, and JPG Internet derivatives. The same *Phase* can be done on several *Workstations* in the system. *Workstations* can be assigned for each *Phase* according to its capabilities Scanning is done on *Workstations* with books scanner, Processing is done on *Workstations* with image processing software and so on.. A time period is attached to each *Phase* so that if it took longer time, the *Job* will be reported as *Late Job*. After finishing any *Phase* the *Users* are able to provide information about the *Phase*. This information is divided into *Phase* specific information, general comments, and file level information. This information will help the next operator who is working on the next *Phase* whether it is a new or a previous *Phase*.

D. The User

The *User* entity represents the system *Users* or the Digital Lab operators. Several roles can be defined for the *Users* to manage their access on the *Jobs*. The *User* can perform several types of *Phases* of a specific *Job Type*. The *User* can also be assigned to work on some *Collections* in the system.

E. The Collection

The *Collection* entity represents logical grouping for the *Jobs*. It may represent a digitization project or a private collection. A *Collection* may contain several documents of different *Job Types*. A group of *Users* can be assigned to work on a *Collection*. Several *Workstations* in the system can be allocated for a *Collection*.

F. The Workstation

The *Workstation* entity represents the computer, where the execution of the *Phases* is performed. Several *Phases* can be done on one *Workstation*. The *Workstation* can be allocated for several *Collections*.

IV. SYSTEM ARCHITECTURE

Fig. 2. shows a representation for the architecture of the DWMS. The system provides all the services through five main modules; *Check-In, Phase Manager, Reporting, Archiving and Administration* module. All modules provide the services after passing through an

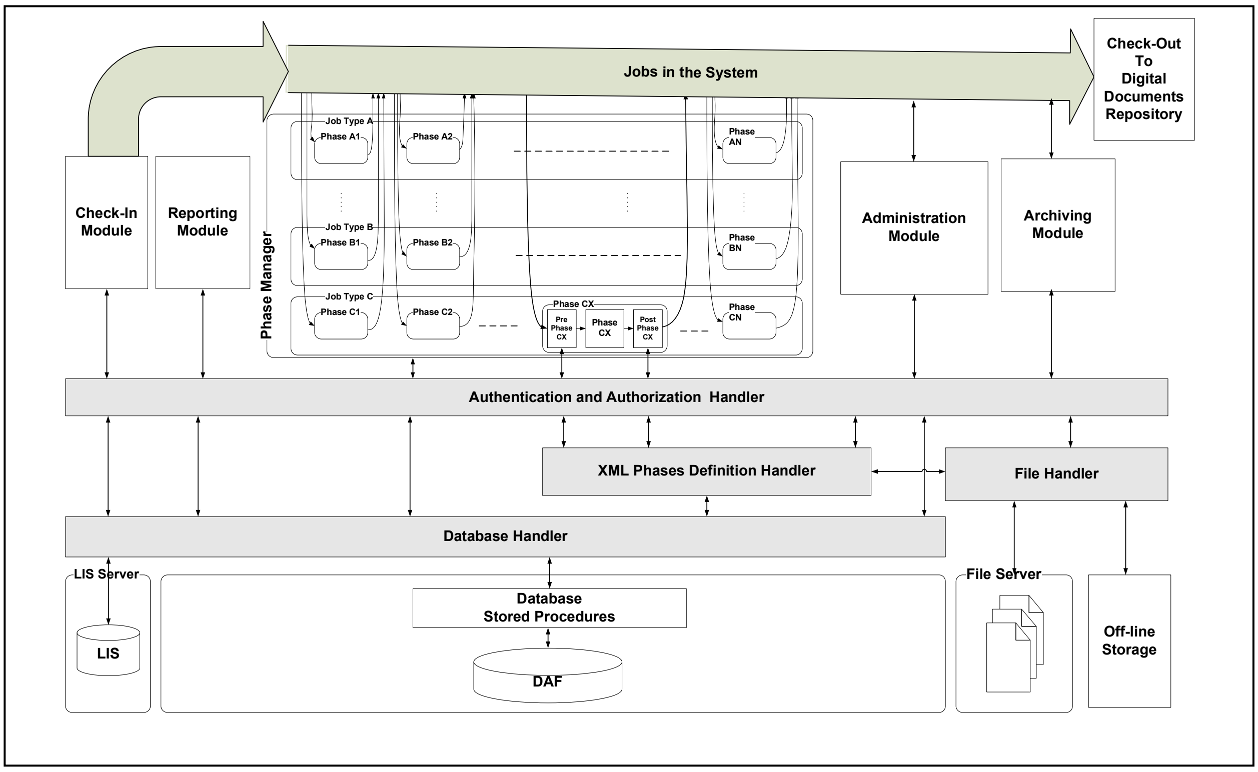

Fig. 2. DWMS system Architecture

Authentication and Authorization Handler. The *XML Phases Definition Handler* accepts the requests related to applying the necessary checks and actions before and after performing a digitization *Phase*. The *File Handler* is used mainly by the *XML Phases Definition Handler* to manage the files checks, copying and movements. All the system configurations, parameterizations and transactions are stored in a database and managed through the Database Handler.

A. The System Handlers

All the provided interfaces and services of the system are accessible through an authentication and authorization handler. This handler is responsible for customizing the application interface to the logged in *User*. Moreover, it authorizes each action or request submitted by *User*.

The *XML Phases Definition Handler* is responsible for interpreting and applying the XML definition of the *Phases.* Each *Phase* has its own *XML Phases Definition* specifying the prerequisites and actions that need to be done before and after each *Phase*. The XML definition contains two main sections; *Pre-Phase* and *Post-Phase*. Each of these sections is composed of three subsections; *Physical, Database and Reflection Call.*

- *Physical* section: In the *Pre-Phase* section, the Physical section allows to describe the necessary folders and files structure required to start work in the *Phase* and which of them should be copied to the client's working folder to execute the *Phase.* For example, it is possible to say that the OCR *Phase* can not start unless there are OTIFF folder with TIFF files and PTIFF folder with TIFF files on the main file server. Only the PTIFF folder is required to be copied from the file server to the client's working folder to do the OCR *Phase*. In the *Post-Phase* section, the *Physical* section allows to describe the necessary files and folders structure required to complete the *Phase*. It also defines which of the folders and files should return to the file server. For example, when finishing the Processing *Phase* there should a PTIFF folder with a number of TIFF files equal to the ones in the OTIFF folder.
- *Database* section: It is usually used in the *Post-Phase* section. It allows to define the structure of database information that should be submitted after finishing the *Phase*. It contains listing and naming for the fields that should be filled by the operator during his work in the *Phase*. This fields are saved as XML text with the *Phase* information in the transaction log for reference and query later using XPath.
- *Reflection Call* section: In this section, DWMS allows to specify the Java function that should be executed either in the *Pre-Phase* or *Post-Phase*. The function can start any process including files management, data entry, zipping, or encoding the files. In this section, it is

possible to write the necessary code to ingest the objects in the digital document repository.

An example of a *Phase* definition is shown in Fig. 3.

```
<Phase Name="Book Arabic OCR">
  <PrePhase>
    <Physical Mode="UnRestricted">
      <Folder Name="OTIFF" Create="false"
              ToDestination="false" NewName="OTIFF"
              Mode="Restircted">
        <File Name="OriginalFiles" Type="tif" Count="+"
              ToDestination="false" Compare=""/>
      </Folder>
      .
      .

    </Physical>
  </PrePhase>
  <PostPhase>
    <Physical Mode="UnRestricted">
      <Folder Name="TXT" Create="false"
              ToDestination="true" NewName="TXT"
              Mode="Restircted">
        <File Name="" Type="frf" Count="1"
              ToDestination="true" Compare=""/>
        <File Name="" Type="art" Count="1"
              ToDestination="true" Compare=""/>
      </Folder>
    </Physical>
    <Database>
      <Field Name="Font" DisplayName="Font Family: " />
      <Field Name="LrnPage" DisplayName="Learn Page : "/>
      .
      .
    </Database>
    <ReflectionCall Method="packageName.doSomething" />
  </PostPhase>
</Phase>
```

Fig. 3: XML *Phase* definition example

The *File Handler* component is used by the *XML Definition Handler* to manage the file copying and movement. It is responsible for the necessary ftp handling with the file server and local file handling on the clients.

All the database interactions are done through the *Database Handler*, who is responsible for interfacing with the database stored procedures.

B. The System Modules

The Check-In Module is responsible for creating a *Job* in the system and fires it to start. Although the *Check-in Module* determines the first *Phase* of a *Job* depending on its *Phase Sequence* , the system allows to handle an exception and allows the *Job* to start from an intermediate *Phase* within the workflow as long as its prerequisites are met. For example, although a printed book *Phase Sequence* is defined to pass through the *Phases* Scanning, Processing, OCRing, Archiving, and Encoding, it also possible to assign it to the Processing *Phase* as long as the TIFF files are ready in the working folder. This allows the accommodations of the system to receive scanned books from external sources.

DWMS check-in has been designed and built to allow for the integration with any metadata source as the Integrated Library System (ILS), document registry, MARC or MODS files. This flexible integration has been achieved by making the check-in module built as plug-in based as described in Fig. 4. The system allows each library to write its own check-in plug-in for its metadata source.

The Check-Out Module is responsible for ingesting the digital objects into the repository. It is implemented to allow for the integration with the institution's repository such that the objects can be ingested into it. The jobs check-out can be written in the *Java Reflection* section of the *XML Phases Definition* explained earlier, see Fig. 3. This will help to customize the *Check-out* Module for the integration with the digital document repository.

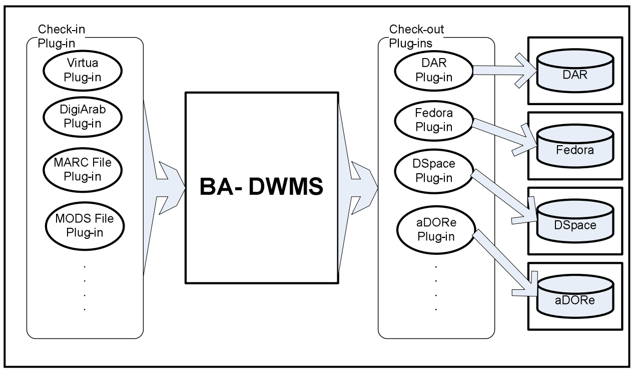

Fig. 4. DWMS Check-in and Check-out

The Phase Manager provides the interface to the digitization laboratory operator. It allows the operator to request a new *Job* to work on, download the working files, and submit the *Job* back to the system to continue in its workflow. Moreover, the *Phase Manager* allows the operator to reject a *Job* after starting working on it. In this case, the operator will have to submit a rejection reason. Afterwards, the system will automatically assign the *Job* to the administrator to review the rejection reason and take the necessary actions. Also the operator can redirect the *Job* to another *Phase* not in its default path. The redirection of *Jobs* is automatically confirmed if the operator has enough permission. Otherwise to redirect *Jobs*, the *Jobs* is automatically assigned to the administrator as pending until he accepts or denies this redirection. The redirection may be due to a problem in a previous *Phase.* For instance, while performing the OCR *Phase*, the operator might have discovered that there were some pages that need further Processing or that there were some pages missing required scanning, thus they need to be returned to the Processing or Scanning *Phase*.

In order to simplify solving problems that happen in previous *Phases* of digitization, DWMS allows the operator to add information on the produced files level. The information contains the files numbers, the required *Phase* that should be revisited, and the problem's reason. This information is saved in the database. Once the *Job* is revisiting a previous *Phase*, the operator will be informed with the files that require reprocessing and the reasons. DWMS allows the administrator to define a list of reasons for each *Phase* to be revisited so that the operator can select the reasons from a drop-down list. This information is propagated for the next *phases* to take the necessary actions on the new produced files. For example, suppose in the OCR *Phase*, files 20 to 25 are missing and require Re-Scanning, the *Job* will be redirected to the Scanning *Phase* with file info about the page numbers that require scanning. The *Job* will be forwarded to the Processing *Phase* with the files level information indicating that files 20-25 require processing and so on.

The Administration Module is responsible for the necessary system parameterization and settings. It allows the administrator to define and manage the *Job Types* with its Digitization workflow and *Phases*, the Roles of the *Users*, *Workstations*, and *Collections*. It also provides the facility to control the matrix covering the relation between *Users*, *Workstations*, *Job Types*, and *Collections*. For example, BA collection contains two *Job Types*; J1 and J2. The collection will be handled on *Workstations* W1 and W2 by the *Users* U1 and U2. U1 will be working on the *Job Type* J1, while U2 will be working on *Job Type* J1 and J2.

The Reporting Module provides the necessary reports to allow for managing the *Jobs* within the workflow. The module provides four types of reports:

- Workflow Tracking reports: provides the status of the *Jobs* in the system. It provides for each *Job Type*, the number of *Jobs* pending , started, and finished within each *Phase*. It also can separate the *Jobs*, which are revisiting a *Phase* to provide information about the new and old *Jobs* in the system
- Pending Items report: provides the redirected or rejected *Jobs* by the operators. The administrator can grant or deny the redirection through this report.
- Late Jobs report: provides a list of the *Jobs* that exceeded its due time either within a *Phase* or in the whole workflow, since there is a due time for each *Job* and a due time for each *Phase*.
- Operators Rate report: helps the supervisors and higher level management to get the laboratory overall production and provide a tool for evaluating the operators.

In addition to the above reports, the module provide a query builder, where the *User* can design his own report on the *Jobs*. The module also provides a search capabilities for the *Jobs* by the various attributes of the job ID, External ID, External ID type, title, creator, *Collection*, *Job Type*, and language. This search can be considered as an access point to the *Job* to assign or retrieve it from an archive.

The Archiving Module is responsible for the archiving process. DWMS allows for the archiving on either online storage, CDs, tapes, or on all of the previous. It also allows to define a new media type with a specific capacity. The archiving information is ingested to DAR in the archiving metadata section. The integration between DWMS and DAR allowed DAR to keep track of the different archived versions of the digitized objects.

C. The Quality Assurance Handling

In this section, we will discuss the Quality Assurance (QA) *Phase*, which is an important step in a digitization workflow. DWMS helps the QA in two stages. First, it allows for providing QA information and decisions during each *Phase*. Second, a QA *Phase* is defined and configured to allow for a complete investigation on the produced digitized objects and all output files.

The first stage is achieved by giving the chance to provide QA information and decisions while moving from a *Phase* to another in the digitization *Phase Sequence*. QA decisions are applied and the *Job* is automatically forwarded to the appropriate *Phases*. During each Phase, DWMS provides an interface to specify the erroneous files, recommended Phase to re-visit and a possible reason for the problem.

The second stage is a QA *Phase* as a separate *Phase* in the digitization *Phase Sequence*. During this *Phase* a complete quality assurance is applied on the produced digitized objects and all output files. Fig. 5. shows an example for the *XML Phase Definition* of the Books QA. The Physical subsection of the *Pre-Phase* section applies the necessary checks on the files and folder structures, denoting that *Phase Sequence* has been successfully completed. The Database subsection of the *Post-Phase* section defines fields to help the operator specifying whether the PDF is Image on Text or not, whether there is Wrong Text and Image Pairing, or there is a Page in Wrong Order or there is an error in the PDF file.

After investigating the objects, the *Job* is automatically forwarded to the earliest *Phase* required to be re-done in the *Phase Sequence*. Once the *Job* started in the re-do Phase, DWMS can display two types of information to help the operator in the re-doing. First, the QA information on the files level, which contains the erroneous files with a possible reason and the QA information defined in the XML Phase definition fields. Of course this type of information will accelerate the re-doing of the *Phase*. The files level QA information is propagated for the next *Phases* in the digitization *Phase Sequence* to take the necessary actions. For example, In the QA of a printed book, files from 10 to 15 requires Re-Scan. The QA files information will contain this information. After finishing the Re-Scan, the QA files information will say that files 10 to 15 requires Re-Processing, then files 10 to 15 requires Re-OCRing and so on.

```xml
<Phase Name="Book QA">
  <PrePhase>
    <Physical Mode="UnRestricted">
      <Folder Name="OTIFF" Create="false"
            ToDestination="false" NewName="OTIFF"
            Mode="Restircted">
        <File Name="OriginalFiles" Type="tif"
            Count="+" ToDestination="false"
            Compare=""/>
      </Folder>
      <Folder Name="PTIFF" Create="false"
          ToDestination="false" NewName="PTIFF"
          Mode="Restircted">
        <File Name="ProcessedFiles" Type="tif"
          Count="+" ToDestination="false" Compare=""/>
      </Folder>
      <Folder Name="TXT" Create="false"
          ToDestination="false" NewName="TXT"
          Mode="Restricted">
        <File Name="OCRedFiles" Type="afn" Count="+"
            ToDestination="false" Compare=""/>
      </Folder>
      <File Name="" Type="pdf" Count="1"
          ToDestination="true" Compare=""/>
    </Physical>
  </PrePhase>
  <PostPhase>
    <Database>
      <Field Name="isIOT" DisplayName="Image on Text:"/>
      <Field Name="WrongTextImage"
        DisplayName="Wrong Text and Image Pairing: "/>
      <Field Name="WrongOrder"
        DisplayName="Pages in Wrong Order: "/>
      <Field Name="PDFerror" DisplayName="PDF Error: "/>
    </Database>
  </PostPhase>
</Phase>
```

Fig. 5: Books QA phase definition in BA Digital Lab.

D. *Achieving flexibility using DWMS*

Workflow management systems provide support for business processes that are generally predictable and repetitive. However, the prescriptive, assembly-line frameworks imposed by workflow systems limit the ability to model and enact flexible work practices where deviations are a normal part of every work activity [13]. For these environments, formal representations of business processes may be said to provide merely a contingency around which tasks can be formulated dynamically [14], rather than a prescriptive blueprint that must be strictly adhered to. In this sense, a workflow process model may be considered a resource which mediates activities towards their objective.

Rather than continue to try to force the digitization workflow processes into inflexible frameworks with limited success, a more adaptable approach is needed that is based on accepted ideas of how people actually work. DWMS has been developed and implemented to be a flexible workflow management system that:

- Considers the defined digitization workflow for a *Job Type* in terms of a *Phase Sequence* as a guide, rather than a prescription for it;
- Provides the facility to define a list of *Phases* that can or can not be included in the default *Phase Sequence* of digitization. The operator can assign the *Job* to any of all of these *Phases*;
- Provides the ability to forward dynamically the *Jobs* to another *Phase* in the default *Phase Sequence*; and
- Allows for changing the digitization *Phase Sequence* by adding or removing *Phases*. The new sequence will be applied on the current and new *Jobs* in the system, leading to natural process evolution

V. LIFE CYCLE OF A JOB

Of particular interest for a digital lab operator is the manner in which *Jobs* are advertised and ultimately bound to specific operator for execution. Fig. 6 illustrates the lifecycle of a *Job* in the form of a state transition diagram from the time that a *Job* is created by the check-in module to final completion. It can be seen that there are a series of potential states that comprise this process.

Each node in Fig. 6 represents a possible state of a *Job*. Each edge within this diagram is provided with a text describing how this transition is initiated. It also indicates whether the transition requires a permission granted to the operator or not.

- Initially a *Job* comes into existence in the *Assign* state to start its execution in a digitization *Phase*. A *Job* can be assigned to a specific operator, or to a group of operators or to any operator.
- The *Job* moves to the *Start* state once the operator used the DWMS to download the required *Job's* files and folders in his working folder. The required files are defined in its *XML Phase Definition*. Started *Jobs* can be either rejected or completed. The operator can reject the *Job* because of a problem and accordingly the *Job* will be automatically assigned to the administrator, in the Pending *Jobs* report, to investigate and assign the *Job* again to the appropriate *Phase*.
- If the operator completed the *Phase*, he may submit the *Job* back to the system to be in the Assign state for the next *Phase* in the *Phase Sequence*. If it is the last *Phase*, the *Job* will be ingested in the repository and check-out from the DWMS. If the operator recommends a *Phase* other than the normal flow, then the *Job* will move to the *Redirect* state, where it is automatically assigned to the administrator to approve or deny such action and put the *Job* in the Assign state either to the system or to a specific operator.

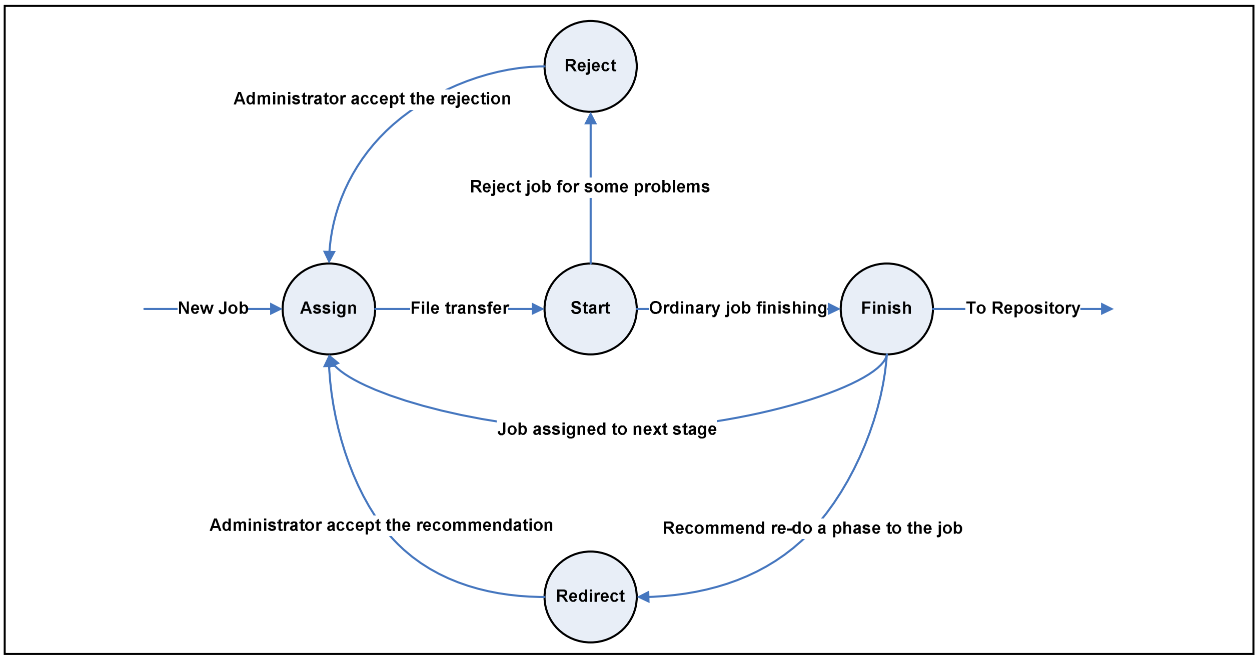
Reject
Administrator accept the rejection
Reject job for some problems
New Job
Assign
File transfer
Start
Ordinary job finishing
Finish
To Repository
Job assigned to next stage
Administrator accept the recommendation
Recommend re-do a phase to the job
Redirect

Fig. 6: Life Cycle of a *Job*

VI. IMPLEMENTATION

The system was implemented on an open source platform. It is written using Java using Eclipse 3.1 IDE and requires JRE 1.5. The system can run on any operating system with a Java Virtual Machine. The database used is MySQL 5.1, with support for stored procedures and XPath query. The MySQL stored procedures were used for all the database interactions. The *Database Handler* interfaces with the database through a MySQL JDBC driver 5.0.

The system allows for the use of the Java Reflection call technology to allow for writing special code to handle difficult *Post* and *Pre-Phase* actions.

A Check-in plug-in has been implemented to import document's metadata from Virtua VTLS, ILS used in BA, and a Check-out plug-in has been implemented using the Java Reflection to ingest digital objects into DAR, the Digital Assets Repository of BA.

VII. CONCLUSION AND FUTURE WORK

In this paper, we presented the DWMS implemented in the Bibliotheca Alexandrina. The system introduces a data model capable of defining different workflows for various types of objects. A flexible integration with both any source of metadata such as ILS and a library digital document repository has been achieved by building the Check-in and Check-out modules of the system as plug-ins to allow each library to write its own modules. Moreover, the Check-in Module supports ingesting a *Job* in the middle of the workflow. The system adapts a flexible *Job* life cycle with history tracking of actions to supports dynamic evolutions and deviations to allow for exception handling. DWMS provides all the necessary tools required to manage the whole process of digitization including its various *Phases*, system *Users*, files movement and archiving. It provides flexibility to simultaneously manage multiple projects with a diversity of materials covering books, journals, newspapers, manuscripts, unbound materials, audio, video, and slides and allows easy integration of tools used to perform functions of the workflow. DWMS is a highly reliable digitization workflow management system that can be customized for large and challenging digitization projects or used out-of-the-box. The system is based on an open source platform and is fully deployed in BA digitization laboratory.

Future work includes:

- Check-out plug-in for Fedora. The system is currently integrated with the DAR repository system of BA. We are planning to build the plug-in modules for the integration with popular repositories especially Fedora.
- Check-in plug-ins will be implemented to support various metadata standards formats MODS, DC, VAR, etc.
- Enhance the software interface with graphical tools to help design and follow the digitization process.

Acknowledgement

Many thanks go to Fadi Edward, Shehab Kamal, and Mohammed Abuouda for their efforts in the design and implementation phases of the system.

References

[1] Russell, N., van der Aalst, W. ter Hofstede, A. & Edmond. , D., Workflow resource patterns: Identification, representation and tool support., in O. Pastor & J. Falcao e Cunha, eds, *Proceedings of the 17th Conference on Advanced Information Systems Engineering (CAiSE'05)*, Vol. 3520 of Lecture Notes in Computer Science, Springer, Porto, Portugal, pp. 216—232, 2005.

[2] van der Aalst, W., Weske, M. and Grunbauer, D. Case handling: A new paradigm for business process support. *Data & Knowledge Engineering*, 53(2):129-162, 2005.

[3] Joeris, G. Defining flexible workflow execution behaviors. *In Peter Dadam and Manfred Reichert, editors, Enterprise-wide and Cross-enterprise Workflow Management: Concepts, Systems, Applications, volume 24 of CEUR Workshop Proceedings*, Paderborn, Germany, October 1999.

[4] Borgida, A. and Murata, T. Tolerating Exceptions In Workflows: A Unified Framework For Data And Processes. *In Proceedings of the International Joint Conference on Work Activities, Coordination and Collaboration (WACC'99)*, pages 59-68, San Francisco, CA, February 1999. ACM Press.

[5] Rinderle, S., Reichert, M. and Dadam, P. Correctness Criteria For Dynamic Changes In Workflow Systems: A Survey. *Data and Knowledge Engineering*, 50(1):9-34, 2004.

[6] Casati, F. A Discussion On Approaches To Handling Exceptions In Workflows. *In CSCW Workshop on Adaptive Workflow Systems*, Seattle, USA, November 1998.

[7] Ellis, C.A., Keddara, K. and Rozenberg, G. Dynamic Change Within Workflow Systems. *In N. Comstock, C. Ellis, R. Kling, J. Mylopoulos, and S. Kaplan, editors, Proceedings of the Conference on Organizational Computing Systems*, pages 10-21, Milpitas, California, August 1995. ACM SIGOIS, ACM Press, New York.

[8] Hagen, C. and Alonso, G. Exception Handling In Workflow Management Systems. *IEEE Transactions on Software Engineering*, 26(10):943-958, October 2000.

[9] Ackerman, M. S. and Halverson, C. Considering An Organization's Memory. *In Proceedings of the ACM 1998 Conference on Computer Supported Cooperative Work*, pages 39-48. ACM Press, 1998.

[10] Larkin, A. K. P. and Gould, E. Activity Theory Applied To The Corporate Memory Loss Problem. *In L. Svennson, U. Snis, C. Sorensen, H. Fagerlind, T. Lindroth, M. Magnusson, and C. Ostlund, editors, Proceedings of IRIS 23 Laboratorium for Interaction Technology*, University of Trollhattan Uddevalla, 2000.

[11] van der Aalst, W.M.P. Exterminating The Dynamic Change Bug: A Concrete Approach To Support Workflow Change. *Information Systems Frontiers*, 3(3):297-317, 2001.

[12] Bardram, J. E. I love the system - I just don't use it! *In Proceedings of the 1997 International Conference on Supporting Group Work (GROUP'97)*, Phoenix, Arizona, 1997.

[13] Strong, D. M. and Miller, S. M. Exceptions and Exception Handling In Computerized Information Processes. *ACM Transactions on Information Systems*, 13(2):206-233, 1995.

[14] Bardram J. E. Plans As Situated Action: An Activity Theory Approach To Workflow Systems. *In Proceedings of the 1997 European Conference on Computer Supported Cooperative Work (ECSCW'97)*, pages 17-32, Lancaster U.K., 1997.

[15] Adams, M, ter Hofstede, A. H. M., Edmond, D., and van der Aalst, W. M. P. Implementing Dynamic Flexibility in Workflows using Worklets. *BPM Center Report BPM-06-06*, BPMcenter.org, 2006.

[16] Content Conversion Specialists. http://www.ccs-gmbh.de/

[17] i2s DigiBook. http://www.i2s-bookscanner.com/

[18] Academic Imaging Associates http://www.academicimaging.com