Chapter 2

The Production and Storage Frameworks and an Architecture

In this Chapter, frameworks for production and storage of digital documents in an office are studied. Based on the frameworks, a conceptual architecture is also designed. The problem of production and storage of digital documents in an e-office can be studied under the following frameworks. These are not formal, but design frameworks.

2.1 The Production Framework

Document production in an office can be studied under a generic framework comprising of three concepts: Document Production Workflow (DPW), Context and Multi-Part Multi-Signature Document (MPMSD).

An office document is an output of an office task. The office task, concerned with a DPW is termed, in general, as review. The review task com-
prises of a set of operations: case examination, composition, signing, filing, linking and flowing. A review task may have specific names at different positions of the workflow. For example, in the travel plan workflow discussed in Chapter 1, the different review tasks may be termed as travel request, departmental approval, financial approval and final approval.

Definition 2.1 An office task \( w_i \) is defined as \( \{OP_i, T_{IN_i}, T_{OUT_i}\} \), where \( OP_i \) is the set of operations to be performed in \( w_i \), \( T_{IN_i} \subseteq TO \) is the set of object types allowed as inputs, \( T_{OUT_i} \subseteq TO \) is the set of object types expected as outputs. \( TO \) is the finite set of object types.

An office work comprises of an ordered set of tasks. Various tasks are usually carried out by several office workers in accordance with the organizational rules relevant to the office concerned. The output object of one task may be the input of the next task. As a result, work objects flow from one task to another. Therefore, office work can be ideally represented by a workflow model. When the work object is a document, we term it Document Production Workflow (DPW) and the output document thus formed from a DPW is termed as a case.

Definition 2.2 A Document Production Workflow can be represented as a directed graph \( W(T, E) \), whose vertices \( T \) represent the set of tasks, \( T = \{tw_1, tw_2, tw_3, \ldots, tw_n\} \) in the workflow and the edges \( E = \{(tw_i, tw_j)|tw_i, tw_j \in T\} \) represent the stages of the flow of the case under review. Both the vertices and the edges are labelled. In the edge \( tw_i \to^k tw_j \), \( k \) signifies that the case under review is in the \( k^{th} \) stage. This signifies that the ordinal number of the latest comment, added to the flowing case so far, is \( k \), assuming the ordinal number of the original request as 1. The label on a vertex signifies the reviewer associated with the task.
All documents in an office constitute a document space and a subspace of it constitute a reference space. The documents in the reference space are referred or cited while producing a new document or a part of it. Only a subset of the reference space may be relevant to a particular DPW. This relevant set of documents constitutes the context. Thus every DPW has a context associated with it. The context comprises of the rules, precedents and other support documents relevant to the DPW.

**Definition 2.3** A Document Space $D$ is the universal set of uniquely identifiable documents in an office.

**Definition 2.4** A Reference Space $R \subseteq D$ is a collection of documents of types rules($R$), precedents($P$) and other support documents($S$), which can be consulted and referred to while generating other documents. Rules are the the documents containing formal guiding principles, policies etc. on all topics of concern in an office. Precedents are the cases handled earlier. Support documents are the documents which are neither rules nor precedents but are consulted or referred to while generating other documents.

**Definition 2.5** A DPW Context $C_W \subseteq R$, of a DPW $W$, is a collection of documents relevant to $W$. The relevance is defined as a predicate based on the different attributes of the documents in $R$.

During composition of a document, a reviewer of $W$ navigates through the context, peruses documents belonging to $C_W$ and draws relevant citations to the document under composition, from the context, in order to provide rules position, precedent position and other supporting documents to complement the position of the document. This process is termed as case examination.
**Definition 2.6** Case Examination is the process of navigation and perusal of the documents belonging to the context $C_W$ of a DPW $W$ done by a reviewer of $W$ as an operation of the task $t_{w_i} \in T$ of $W(T, E)$.

A reviewer of $W$ can add new documents from $R$ to $C_W$ but cannot remove any document from $C_W$. Moreover, as soon as a document is added to $D$, if it satisfies any relevance predicate of $C_W$ then the document will automatically be included in $C_W$. Only the designer of the DPW, who is authorized to modify the DPW can redefine the predicates and thereby remove documents from the context.

Once the document is generated, it is registered in a proper way and related documents are linked for immediate pick up. The document then flows from its point of origin to the target.

If we consider the output of an office task as an elementary document, then the output of a DPW is a composite document, composed of multiple elementary documents, each containing comments of the corresponding reviewer. Such a composite document is termed as a Multi-Part Multi-Signature Document (MPMSD), where each part is an elementary document.

**Definition 2.7** A Multi-Part Multi-Signature Document (MPMSD), $D_W$, produced in a DPW $W$, is an $n$-tuple, $n \geq 1$, such that $D_W = (d_1, d_2, d_3, \ldots, d_n)$.

Each part $d_i$ in turn is defined as a 4-tuple $(m_i, c_i, \sigma_i, s_i)$, where $m_i$ is the comment of the reviewer $s_i$, $c_i$ is the context of $m_i$, based on which the comment $m_i$ is produced, and $\sigma_i$ is the signature of $s_i$.

A case is a MPMSD. Since, the parts of a case are produced at different points of time, the context of the DPW may be at different states at different points of time. As a result, the context of creation of any two parts of the same
case may not be the same. Moreover, during case examination, a reviewer may cite another document, not in \( C_w \), as a support for his comments. This document is specific to the case and may not be relevant to other cases of the DPW. Therefore, this document need not be included in the context \( C_w \) of the DPW \( W \) but it needs to be included in the context of the case.

**Definition 2.8** A case context \( C_{Dw} \) of a case \( D_w \) consists of the context \( C_w \) of the DPW \( W \) and a set documents, \( \delta_{Dw} \), specific to \( D_w \), added to \( C_{Dw} \) at different points of time by the reviewers of \( D_w \).

Therefore, the context \( c_i \) of a comment \( m_i \) of a part \( d_i \) of a case \( D_w \) is defined by the state of \( C_{Dw} \) at time \( t_i \), where \( t_i \) is the time of production of \( d_i \). The state of \( C_{Dw} \) in turn is defined by the states of \( C_w \) and \( \delta_{Dw} \) at \( t_i \).

For example, let us consider the travel plan workflow discussed in Chapter 1. The workflow has a context comprising of documents containing rules on travel permission, on leave to be granted during travel, on TA/DA and precedent cases of travels granted or rejected. It also contains documents on budget provisions etc. as support documents. Each of the documents in the context is a MPMSD. An employee \( A \) composes her application citing some of these documents. \( A \) also wants to cite her leave account report on the date of application in support of her application. Since this report is specific to her case only, so she adds the report in the context of the case and submits an application \( m_A \) to \( B \) with a request to allow her travel to an organization. This is the first part part of a new case. Before forwarding the case to \( C \), \( B \) wants to peruse her forwarding note on a similar case forwarded earlier. When \( B \) retrieves the concerned part \( d_2 = (m_2, c_2, \sigma_2, s_2) \) from the precedent case, the context \( c_2 \) prevailing at the time of creation of the forwarding note \( m_2 \) is regenerated. \( c_2 \) is defined by all the documents and their parts available to the context of the case, of which \( d_2 \) is a part. \( B \) then
appends her forwarding note $m_B$ to the application and sends the same to $C$. $C$ finds that a resolution taken in a recently concluded Finance Committee Meeting is relevant not only to the case in hand but also to all future cases of the workflow. Therefore, she registers the document containing proceedings of the meeting, of which she is the convenor, in the document space of the office first as a rule, which automatically qualifies for a member of reference space. $C$ then adds it to the context of the workflow. Now, the comment $m_C$ is created, citing the newly added document from the context, and is forwarded to the director $D$ and $D$ approves the case.

2.2 The Storage Framework

The main objective of storage of information in an office is to keep track of the history of who did what, when, why and how. Thus storage in an office serve as the organizational memory, where the documents are the neurons. Therefore, the central issue is to store the documents in such a way that they can be identified, located and retrieved in an efficient way. Moreover, from a document thus retrieved, all the related documents should be reachable in a simple and straightforward way. The framework of storage and retrieval of office documents comprises of the following concepts: life-cycle, representation, identification, organization, relationship, efficiency and reliability of office documents.

2.2.1 Life-Cycle

Normally, an office document has a long life time and during its lifetime it passes through different states. The life-cycle of a MPMSD can be represented by a state transition diagram. A state transition diagram is a digraph
$LC(S, O)$, where $S$ is a finite set of states represented as vertices and $O$ is the finite set of operations on MPMSDs represented as labels on the arcs.

$S = \{\text{born, active, reference, archived, expired}\}$

$O = \{\text{registration, addpart, close, reopen, archive, de-archive, burn}\}$

The states and the arcs of $LC(S, O)$ are labelled and the labels are the corresponding elements of $S$ or $O$. On execution of an operation on a MPMSD, a transition occurs from a state to the next state of the life-cycle. The life-cycle is shown in figure 2.1. When a new MPMSD is created, it is in the born state. On registration it transits to the active state. Registration of a new document means inclusion of a document in the document space $D$ with a unique document identifier. Multiple parts can be added to a MPMSD in the active state. On execution of the addpart operation, the MPMSD re-

Figure 2.1: Life Cycle of a MPMSD

born state. On registration it transits to the active state. Registration of a new document means inclusion of a document in the document space $D$ with a unique document identifier. Multiple parts can be added to a MPMSD in the active state. On execution of the addpart operation, the MPMSD re-

35
mains in the same state. The addpart operation links a part to the MPMSD. If the part is a new one, the addpart operation also includes registration of the new part and it precedes the linking. On closing a MPMSD in the active state it transits to the reference state. Parts cannot be added to the MPMSD in the reference state. On reopening a MPMSD in the reference state it transits back to the active state again. On execution of an archive operation on a MPMSD in the reference state it transits to the archived state. Archival may include compression of the document. The reverse transition occurs on the operation de-archive. On execution of the burn operation, an archived document transits to the expired state, which is the final, and no return state. The burn operation physically deletes the document from the system. An archived document may be associated with a life-tag. A life-tag signifies the life of an archived document based on the values: permanent, semi-permanent, temporary, immediate etc. of the information contained in the document. On the expiry of the life in the tag the document will be automatically burned. Therefore, the de-archive operation is applicable only before the expiry of the period mentioned in the life-tag of an archived document. The main characteristic that differentiates these states is the accessibility. Archive objects have no access privilege for the reviewers. The reference objects are read only and the active objects may have privileges like read, write, modify etc.

2.2.2 Representation

An office document has three aspects: profile, content and a presentation. The profile represents a document.

Profile

A profile is the bio-data of the office document. It contains meta-data of the
document which provides a detailed description of the document. The profile comprises of a set of keywords and three types of records: production record, storage record and flow record.

- **keywords:** This is a limited collection of representative terms from the vocabulary of the office concerned, which represent the content of the document. Most existing text retrieval techniques rely on indexing keywords or indexing terms. There are standard models for keyword based retrieval, like the vector space model [16]. We excluded keywords from the discussion of our model but it can be easily incorporated. Unfortunately, keywords alone cannot adequately capture the office document contents, resulting in poor retrieval performance. We need other attributes, like record attributes to complement the keyword description of an office document. The record attributes can be categorized as follows:

- **production record:** This record consists of production related attributes like, *class, type, topic, date of production* etc., of an office document. A document may belong to one of the classes like *rule, case, support document* etc. A document may be created using some templates, called types or forms. For example, *office order, notice, casual leave application* etc. are different types of documents. Moreover, a document may be on one or more topics. The attributes are used in designing multi-dimensional Page Cube model, discussed in Chapter 3.

- **storage record:** This record consists of storage related attributes like address, size, authorization, state etc. of a document.

- **flow record:** It is a record pertaining to the flow of a document from one point to the other. The attributes may be senderId, receiverId,
time of sending, time of receiving the document.

Content
The content of a document may be multimedia information. But, for the present work, we assume that it contains only text. The simplest type of digital document is plain text, which contains only the natural language text of the document with not much restricted formatting and structural information. The advent of word processing and text formatting systems introduced "tagged" or "marked up" documents to substitute for plain text documents.

Presentation
The content of a document is presented for display or for printing in a layout framework. The layout framework associates the contents with a hierarchy of layout objects such as pages, columns etc. The layout structure is also hierarchical in nature. It also includes presentation rules, like a chapter should be in a new page, the content should be justified both left and right etc. HTML is now a de-facto layout framework for digital pages. Thus the layout framework provides the get up of a document. For storage and retrieval of office documents, the focus of our discussion is on the profiles of the documents. Therefore, content encoding and presentation aspects of a document are excluded from the rest of our discussion.

2.2.3 Organization
There are different states of office documents as described in section 2.2.1. For persistent documents we are concerned with the active, reference and archived states. The documents can be organized in such a way that all documents in a particular state will be stored in the same data-storage. Accordingly, we can have three types of storages: active storage, reference
storage and archive storage. A document may be stored in the respective category of storage based on the state in which it belongs. Categorization of storages provides a level of access control. For example, any attempt to access an archived document directly by a reviewer other than some authorized users like the storage manager will be denied straight away.

State transition of a document may induce migration of the document as well as other related objects like digital certificates etc from one storage to the other. The migration of objects from one storage to the other are recorded in the migration table maintained in the respective storages.

2.2.4 Identification

A document may be represented by the attributes of the profile but for identification of a document during storage and retrieval a unique address is required. In a paper-based system such a unique string is generated manually combining some of the attributes of the profile, which serves as the unique document identifier. In an e-office an equivalent identifier is to be generated automatically. In our framework, a document may migrate from one storage to the other as soon as it changes its state and accordingly the physical address of the document may change from time to time. But at a particular time a document will have one and only one physical address. Therefore, each document in the document space will have a pair of addresses: a time varying physical address and an invariant logical address.

2.2.5 Relationship

As soon as a document is produced in an office, the document is implicitly related with a set of documents belonging to the document space \( D \). The implicit relations are set up by virtue of production of the documents in the
office concerned. For example, a newly produced document of a particular type on a particular topic will be closely related to other documents of the same type and/or on the same topic. Again, the types are hierarchically linked to each other. Thus, the documents are related further as siblings, ancestors and successors etc. Similar is the case with topics. On the other hand, a document may be explicitly related with other documents. For example, when a document is cited as reference in another document, then these two documents are explicitly related. In our framework, the relations among documents are set up with links. In DPW, links are of two types: implicit link and explicit link.

Definition 2.9 Implicit links are the links which are set up among documents implicitly based on the attributes of the profiles of the documents.

For example, key-word links, type links, topic links etc. are the implicit links.

Definition 2.10 Explicit links are the links which are set up explicitly among documents by the reviewers or by other entities of the system.

For example, citation links, part link etc. are explicit links. The links in our framework are bidirectional. Every link is a conjugate pair of directed hyperlinks: one forward and one reverse.

2.3 The DPW Architecture

A software architecture can be viewed as a style or a method of design and construction or strategic policies and patterns that shape a system. It provides a common definition in abstraction for different components involved in a system [23]. In this section, we present a conceptual architecture for
production and storage of digital office documents. A preliminary version of the architecture was published in [36].

A common method in software architecture design is based on the multi-layering principle. Layers are popularly termed as tiers in software architectures. Each tier is an abstract representation of a perspective of the application. In the present architecture, different tiers, and different components in each tier, are identified and discussed. Common business application architecture of today comprises of three tiers, based on three perspectives: corporate data representation, business logic to manipulate the data and presentation of input output data or information. The architecture we propose here is also a three tier architecture consisting of the following tiers: view, logic and data storage. The view consists of user interfaces, the logic consists of processes to manipulate data and data storage stores data. These three tiers are again grouped into Client and Arbiter with split logic. The logic is split into client logic and arbiter logic, because the data manipulation will be done by the client and the arbiter cooperatively. The client contains the view and the client logic and the arbiter contains the arbiter logic and data storage. The client and the arbiter will communicate over a network using some protocols. This architecture is based on a centralized arbitration mechanism, where all the documents must be routed through the arbiter. The architecture is shown in figure 2.2.

The components of the architecture are described in the following sections.
Figure 2.2: The Conceptual Architecture for DPW
2.4 The View

This tier gives us basically the Graphical User Interfaces (GUIs) of the system. The physical metaphor conceived here, for this architecture, is the office desktop. View comprises of three interfaces: Production Interface, User Interface, and Workflow Interface. A user interacts with the system through these interfaces. The interfaces are discussed in brief below:

2.4.1 The Production Interface

Through this interface a reviewer interacts with the production agent of the client logic module to review a case received. The reviewer examines the case in the context of existing rules, precedents and other support documents, composes his own comments, cites references from the context into the comment to establish the rules position, precedent position etc. of the comment, and forwards the case to the next reviewer. The main elements of this interface are:

- **Workspace**: It is the area where a document under creation or a document retrieved from the arbiter, will be displayed.

- **Inlog and Outlog**: These are the indices of documents incoming to and outgoing from this desktop respectively. The indices include the time of receiving / sending and the user ids of from whom received or to whom sent etc. Thus the Inlog and the Outlog of a reviewer keeps the record of flow of documents.

- **Context**: It is the dynamic context attached to the workflow. It is basically a document which contains a list of hyperlinks to the documents referenced: rules, precedents and support documents. It also contains
templates based on the predicates defined on the profiles of documents to enable the reviewer to search for documents to be included in the context, if required.

2.4.2 The Workflow Interface

Through this interface a user interacts with the workflow agent of the client logic module. The interaction may be different for different roles. A reviewer interacts to find out the status of an active case of the workflow. The status includes the reviewer with whom the case is lying and for how long, the time taken to complete a particular task etc. A work manager inherits a reviewer and additionally can create a new DPW and can modify an existing workflow in different aspects. For example, a manager can change the route of the DPW, can temporarily reassign a task to another office worker of the same role in case one in the review line is absent, can create and modify the context template and the default context as need arises. The main elements of this interface are:

- **DPW design:** Through this a work manager interacts with the workflow agent to create a new DPW or to modify an existing DPW.

- **Case status:** Through this a user interacts with the workflow agent to find out the status of a case.

- **Load balancing:** Through this a work manager interacts with the workflow agent to find out the load of an office worker. The load may be calculated based on the number of DPWs the user is participating as a reviewer, the number of cases flowing into and the number of cases flowing out from the desk of the user, average time of completion of the tasks for the cases handled etc.
2.4.3 The User Interface

This is a standard interface through which a user interacts with the user agent for mutual authentication of the user and the arbiter and also to set up a signed session key.

2.5 The Client Logic

This module is populated by client side processes. The processes in this module are termed as agents. Main agents are: user agent, production agent, workflow agent and crypto

2.5.1 Crypto

This component is responsible for all security activities in the client side. The responsibilities include: encryption and decryption of messages, verification of origin, confidentiality, content integrity of the messages.

2.5.2 Production Agent

This agent is responsible for client activities during production of digital documents. It interacts with the production manager of the arbiter using the production protocol, discussed later. The responsibilities of the production agent include:

- create original request documents using a template, submit it to the arbiter via crypto.
- receive the document sent by the arbiter and verified by the crypto and forward it to the view module for display.

45
• get the context of the DPW from the production manager,

• get the reference document selected out of the context by the user from the production manager,

• update the context by inclusion of reference items in the context during review

• submit the updated context, basically the new inclusions, to the reference manager

• draw citations from the context to the comments and thus form a part and submit it to the arbiter via crypto.

• receive the evidence of submission from the arbiter and update the inlog and outlog indices.

2.5.3 User Agent

This agent is responsible for client activities during user authentication and session setup. It interacts with the user manager of the arbiter logic using the authentication protocol described later. The responsibilities of the agent include:

• request the user manager for registration of a new user

• speak for the user during mutual authentication of the user and the arbiter

• collect session key(s) from the user manager during a login session

• return signed session key(s) to the user manager via crypto.
2.5.4 Workflow Agent

This agent is responsible for client activities regarding design and management of document production workflows. The responsibilities include:

- interact with the authorized user through the workflow interface to get the required parameters to create a new DPW or to modify an existing DPW. Modification includes reassignment of tasks, change of rules etc.
- get the request for workflow status from the user and display the status info received from the arbiter.

2.6 The Arbiter Logic

The arbiter is the central hub of the architecture. It certifies subjects and objects, authenticates subjects and objects, manages storage and retrieval of objects, manages authorizations of subjects on objects, handles time-stamping etc. Since it is the arbiter, the resolutions of disputes based on evidences stored in the data storage of the arbiter will be final and binding. The services of the arbiter are provided by the following components called managers.

2.6.1 Crypto

It performs the same function as the crypto component of the client logic in the arbiter side.

2.6.2 Production Manager

This is an important component of the arbiter. The services provided by this manager include:
• forwarding a document to the next reviewer (client)

• addition of the part submitted by the client to the MPMSD under production,

• sending evidence of submission (NRS) of parts to the client

• recording document flow in the log books

• time-stamping the evidences of occurrences of events in the communication (sending, receiving, authoring etc.)

• authorization flow management (access control) during production

• rendering context and the reference documents to a client when requested for

• communication with other components

2.6.3 User Manager

This component provides the user management services. The services include:

• registration of a new user,

• de-registration of users (employees transferred, terminated, retired, suspended etc)

• generation of conjugate pair of private-public keys per user

• issuance of credentials (digital certificates) to users

• revocation of digital certificates
• maintenance of multi-version digital certificates

• generation and collection of signed session keys from users during a login session

• communication with other components

### 2.6.4 Storage Manager

The services of the component include:

• encryption of documents with a master key known only to the storage manager while storing in the data storage and decryption after retrieving.

• storing and retrieving objects from different storages of the data storage.

• migrating objects from one storage type to the other as soon as criteria are satisfied.

• maintenance of migration tables in the storages

• archiving and removal of reference objects

• deletion of archived objects after the expiry of the life of the objects

### 2.7 The Data Storage

This tier is basically a repository of office objects. There are three storage types - active, reference and archive.
2.7.1 Active Storage

Here, all the active objects are stored. Active objects are basically operational data. MPMSDs which are under production are the main objects stored here. Apart from MPMSDs, workflow documents, context documents, user credentials, log books, session keys, program codes, program credentials, migration tables are also active.

2.7.2 Reference Storage

Here all the reference documents are stored. An active MPMSD becomes a reference document as soon as it is closed and it is migrated to the reference storage. Other reference documents are: older versions of workflow documents, and program codes, rules, precedents, revoked user credentials, log book entries related to the flow of the closed MPMSDs and the signed session keys of closed MPMSDs that are not associated with any other active document. The objects stored here are read only to authorized users.

2.7.3 Archive

The reference objects which are not accessed for a long time are archived and are migrated to the archive storage. The objects stored in the archive cannot be read even. The objects are stored in compressed form.

2.8 The Protocols

There are three main protocols by which client agents interact with the corresponding managers in the arbiter. The protocols are:
• **Workflow Protocol:** This is a protocol used by the workflow agent and the workflow manager during interaction. The interaction is mainly during design, modification and maintenance of DPWs.

• **Production Protocol:** This is the major protocol. It is used by the production agent and the production manager during production of MPMSDs. The protocol will be discussed in detail in Chapter 4.

• **User-Arbiter Protocol:** This is a standard protocol used for authentication of entities. Here the user and the arbiter will be authenticated through this protocol. The protocol provides peer authentication of the user and the arbiter.

There are standard solutions for workflow design. Even graphical workflow design tools are available. Example products are Lotus Notes, Office.IQ, Oracle Workflow, WorkMAN, Visual WorkFlow, FlowMark etc. In the present work we assume that standard workflow design tools, suitable for DPW are available. Therefore, no further discussion on the components of workflow design, like workflow interface, workflow agent, workflow manager and workflow protocol has been done in this work. There are standard solutions for user authentication, like Smart Card technologies. Moreover, in depth theoretical study in user authentication, including peer authentication and secure protocols, are done in [10, 21]. Therefore, we also assume existing user authentication components are sufficient for entity authentication. Only additional concept which needs to be added to peer authentication is the establishment of a signed session key, which is a prerequisite for production protocol to start. It is discussed in Chapter 4.
2.9 Discussion

In this chapter production and storage framework for DPW is presented. The frameworks are design frameworks. The objective of the frameworks is to provide a clear perimeter of the production and storage perspective of the DPW problem. A conceptual architecture is also provided in the present chapter. While the frameworks provide vertical perspectives, the architecture provides a horizontal broad perspective across all the vertical perspectives. The architecture is based on a central arbiter. The need for central arbitration comes from security perspective of the problem, which is the subject matter of Chapter 4.