Document Workflow Support: Applications and Controllers

Roelof Hamberg
Océ-Technologies B.V.
Venlo, The Netherlands

Abstract

An integral approach for support in document workflow is presented. The approach exhibits a number of unique properties, one of which is the representation of tasks in terms of document states. As a result the system becomes more flexible with respect to multiple workflows, because production-specific knowledge is not dispersed in the approach itself, but the focus is on the description of the available input and the requested output of the workflow. Further, the communication framework is less error-prone, because the capabilities of the system can be incorporated in the possible document states without the need to know all possible workflows.

Introduction

What is a document? According to Webster's dictionary, it is anything written that gives information or supplies evidence. In our case, I'd say that any carrier of visual representation of information could be called a document. Explicitly, a document can be an analogue or a digital medium to carry that information. It is tempting to state that documents are mostly deliverables of projects, meant to convey information from one person to another. This information has to be encoded into a document format by the source (called writing), while the target must decode it (called reading). The only constraint that I'd like to set for now is that the coding channel should be visual. Further, information can change, multiple pieces of information can be compiled into one document, and one document can provide information for a large audience – all these properties are equally valid for both analogue and digital documents.

What is workflow? It may be difficult to give a one-sentence definition, but aspects of it can be detailed. First of all, when a person does his/her daily work, with different tasks executed after one another, this constitutes the workflow of that person. It is well known that many changes of context decrease the overall productivity, while too little tasks make life boring.

Secondly, one can talk about devices executing tasks in very much the same situation: devices perform a number of tasks in a certain order. Some devices don't like switching of context either, but this might be optimized. In general, a device becomes more productive if it can switch its context quickly and is independent of other entities.

As a third example, one can consider a project being executed. A project, whether small or large, is defined by having deliverables which are due within certain time limits and with limited resources. There is always a stakeholder of the project. The deliverables mostly aren't there in one iteration cycle; usually, they can be divided in smaller pieces or intermediate results can be defined. In either case, interdependencies exist between the intermediates. The definition of intermediates and their interrelations lead to a workflow, i.e., a series of tasks has to be carried out in a certain order to achieve the desired end result. Mostly, the end result is more important than the process, although impeccable working methods can drastically improve the quality level of the deliverables (cf. CMM model).

In a document production environment the three examples are combined. A number of persons are simultaneously busy to get a number of projects, i.e., document production jobs, done with the aid of a number of devices.

Considering the area of document workflow, a number of observations can be made just on basis of the above definitions:

- If the number of people involved is just one and also only one device is involved, we have encountered the simplest case of workflow. An example is someone making a simple copy.
- The interface in the above case can be explained as telling the device what the source document looks like, telling it what you want to have as a result, or telling it what you want it to do with the source document ("I want 10 copies" versus "make 10 copies").
- If the number of people is larger than one, at least at one place one person is explaining someone else what he/she wants. Furthermore, document(s) is/are exchanged in some form. Such interfaces are essential for enabling a smooth workflow. The requirements of the client have to match the capabilities of the worker. The same story holds for multiple devices, and for mixed situations.
- Analogue document interfaces are harder to deal with than digital ones if device-device or person-device interrelations are considered. For person-person interrelations this depends on the physical distance. If the document interface is digital, automation is more straightforward although not trivial.

- If a smaller number of workers can execute a project, less communication is needed. Therefore, the chance for errors or productivity loss is smaller.
- Productivity of the complete set of workers (people and devices) has several aspects:
  - It takes a certain amount of worker time to carry out a project. Faster workers need shorter time.
  - A higher degree of independence of the workers enables a tighter time schedule. This holds for a single job as well as for handling multiple jobs simultaneously.
  - Running out of resources and making errors dramatically decrease overall productivity.
  - First-time-right production helps tremendously.

The above observations partially form starting points for some Océ solutions on document workflow support.

Key Principles for Solutions

As sketched above, a document workflow is seen as a set of tasks that is carried out by persons and devices in order to create output documents from input documents. In general, one stakeholder, i.e., one client, defines the input document and a description of the required output document, while multiple workers do the production of the output. It is essential to realize that the client not always knows how any worker does his/her/its work, nor which workers are there. Moreover, a worker may outsource the work to other workers, which puts him/her/it in the role of a client. Therefore, the communication between clients and workers can best be done by descriptions of input and output documents, and not by task instructions because the latter are not known.

When multiple workers have to deliver a required output document that was defined by a client, it is not clear at job definition what each worker has to do. Furthermore, it would be advantageous for productivity and system flexibility if the workers can do their work independently of each other. Hence, the need for a work divisor and a work scheduler has been identified. These components together are called workflow management.

In order to complete a client-initiated order, a variable number of workers can be required in variable sequences. Therefore, in order to make life easy for the work divisor, it is useful to apply the same task paradigm for workers as the clients. This means that a worker task is specified by descriptions of input and output document. An additional advantage of this is that the job divisor does not have to know the way in which a worker performs his task. The information models for documents as well as for client and worker tasks are unique, i.e., they are the same for analogue and digital documents for any worker and any client. When an input and output document convey the same information, they can be interpreted being the same document in two different states. A worker task consists of transforming a document from one document state to another.

The strength of one information model for analogue and digital documents is that task modeling becomes independent of the type of documents that are used as input and delivered as output. All clients and all workers of the system can use the same model. Further, the information model is designed in such a way that it closely relates to the way people talk about analogue documents. This is useful for clients and workers that are representing humans.

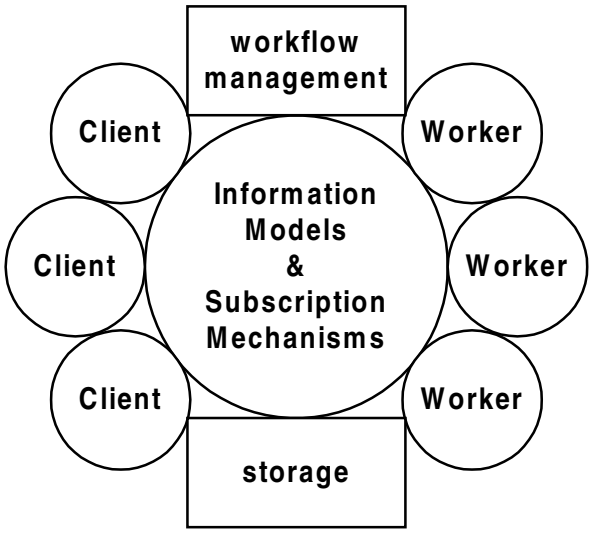

Figure 1. The generic architecture supports division of client-initiated work over different workers.

The architecture as sketched in Figure 1 is truly made generic when all entities only communicate through the information models in the center. As a consequence of that requirement, the workers have to indicate their capabilities and constraints in an information model as well. The job divisor can use this information for determining the worker task sequence that is needed to generate an output document for a client. In view of the above, the job divisor searches a path through a document state transition network. A found path consists of a series of document states that can be transformed into each other by a series of workers. Each of these transformations is possible based on information that was published by the respective worker itself.

For a system that is built around a central information repository, a so-called blackboard, that contains the essential information to execute tasks, a subscription and notification mechanism of new tasks and of information changes is indispensable. Only then the responsibilities of client, blackboard, worker, and workflow management can be addressed in a proper way and full independence of the modules can be guaranteed. Clients can monitor document and worker states; moreover, the clients can subscribe on these states' changes. This facilitates tracking in an easy way. Workers can monitor and adapt document states as well; this controls their progress.

Océ Document Production System Controller

A number of new multifunctionals of Océ, e.g. the Océ DPS400, will have the Océ DPS (Document Production System) Controller as their controller. This controller has been designed according to the generic scheme sketched in Figure 1.

The clients that are known to this controller, are the local user interface, the network connection client, a key operator client, and a service client. Each of these clients can formulate jobs on the central blackboard, independently of each other, but readable for each other. The local user interface client provides user feedback about job settings by which the user at the machine can check whether his/her intentions are well understood.

Workflow management as well as storage has been implemented in software. The storage has been built on the standard file system of Windows 2000. The job divisor breaks client tasks into smaller worker tasks, while the job scheduler implements worker queue management.

The workers are either software layers that connect to physical devices, i.e., the scan station and the print engine, or software modules that carry out specific tasks, like ripping a PostScript file. These workers deduce their tasks from worker queues at the central blackboard that contain worker tasks, which are defined in terms of specifications of input and output documents.

The information model for a document in this controller is a flexible object-oriented model that very closely reflects the physical as well as the information representation of the document. The representation of a document reflects both the intended output document and the (partial) physical existence of the output document. The physical existence of the output document enables tracking functionality in a natural way. Missing resources can be spotted as well at an early stage by combined inspection of the intended output document of each task and the actual system state of resources.

Océ Doc Works: A Server

Next to existing and new devices a new server application has been launched, i.e., Océ Doc Works. This server addresses the remote connection of end users to print rooms and support for print operators to do their work. The logical architecture of Océ Doc Works follows the same line of reasoning as sketched before in Figure 1.

The clients are end users in this case. They can formulate a job in terms of the information model that is provided to them. This is called a job ticket, the definition of which is transferred to them via http at the start of the job creation process. This ensures matching of capabilities of the print room with the requested output document. In order to be sure that their provided input document is correct, preview functionality is offered.

The operator fulfills the role of workflow manager. Currently, this is not fully automated, but only supported through partial tools, which assist in formulating the worker task(s). Storage is implemented through database technology in which the job ticket plays a central role. In fact, the document itself is the input document, while the job ticket reflects the output document specification.

The workers consist of printer drivers, i.e., software layers that connect to physical devices, and Acrobat plug-ins. An example of another worker is the scan-input module that cooperates with the Océ 3165 with scan option. The queues of these workers are very small, because the operator performs the tasks of workflow management and he/she is in general not very multitasking-enabled.

The information model for a document is currently its PDF representation itself. The document state can be interpreted by the operator (fulfilling the task of workflow management) by viewing the document in Adobe Acrobat. Nevertheless, when this task, an obvious candidate for automation, is to be implemented in software, the document will have to be represented in a more explicit way which is currently only done for the requested output document. For document state tracking the job ticket can be used as is; it provides the possibility to indicate which tasks in the workflow have been done already.

Division of Responsibilities

Up to now, our approach of document workflow support seems too generic in the sense that no criteria are indicated that help in deciding what task should be supported at what location in the print room, nor how workers should be designed that can handle a well-defined set of tasks.

At this point the workflows of people and devices themselves, handling multiple tasks that involve different documents, enter the discussion. These tasks relate to the role and responsibilities that the respective people or devices have. Humans have problems with too many context switches, but they are more flexible than devices with respect to performing a sequence of tasks to obtain a sensible output document. They do not like tasks that are too repetitive. On the other hand, devices are less error-prone and very good in performing repetitive tasks. Devices are also faster and cheaper in general. Although the client role is played by a human in most cases, the repetitiveness issue indicates that a client-side specification of a job should be facilitated by providing coarse-grained settings (e.g. a document-level setting that holds for all pages) that reflect fine-grained properties (e.g. at page level). Humans should fulfill the role of worker as less as possible for the same reason. Furthermore, humans can fulfill the role of workflow management as long as the information model for tasks in terms of document states is not sufficiently rich or too difficult to give a concrete form through an understandable user interface.

One of the starting points for improving workflows often consists of automating as much as possible. Nevertheless, handling of analogue documents in view of reproduction is in general more difficult than handling of digital documents. Often a person is required to assist in handling the analogue documents. Therefore, the system is designed such that tasks that involve the transformation of documents to or from the analogue domain always take place at the border of the system. Whether or not the digital processing in between is done at the controller only or requires the server application, depends on the need for human intervention and feedback. If the required document state is too complex to specify in detail or the task is too complex to automate (e.g. spot removal), humans should specify subtasks (i.e. play the role of job divisor) or even act as worker (e.g. in the case of spot removal in scanned documents).

External Standards and Our Approach

In the domain of digital documents PDF is serving our approach better than PostScript. The problem with PostScript is that is does not reflect a document state, but a mix of an input document and a requested output document. Consider the production of a booklet. Nowadays, this is often done using a specific print driver setting. The print driver creates a representation of the input document in combination with the transformations that are needed to obtain the booklet, both in the digital domain and in the analogue domain. The problem with this approach is that preview is unreliable, switching to other output document formatting is hardly possible, and the interface between driver and device is not open.

Alternatively, booklet production is either done on the server, where the viewable PDF indicates the document state (it is a booklet with particular properties), or on the controller, where the input document can be described independently from the output document (the latter of which is a booklet). The degree of complexity of booklet characteristics determines whether the controller alternative is feasible or not. Of course, in the case of PDF it is easier for the user to play the role of the job divisor because the feedback on the document state is immediate. Currently, the Océ DPS Controller does support booklet production through the PostScript route, while booklet production in Océ Doc Works is done through a plug-in for Acrobat.

PDF fits our model better, but at present only the actual document state can be reflected with this representation format. It provides no support for a future analogue document state, nor can the format easily be interrogated for a report on the actual document state, i.e., it can only be viewed. The job definition format is one way to address this problem. It provides a way to formulate the intent, which is a representation of the requested output document. Nevertheless, the JDF modeling of the tasks that are needed to produce the output document does not provide an easy way to inspect the current document state, nor to build flexible software on top of it to handle a configurable set of workers, nor to alter the intent in an easy way. JDF seems to address automation of the current situation very well, whereas the degree of flexibility with respect to adding new software modules is questionable.

Conclusion

A new Océ controller and server application are introduced in one general view that explains how document workflow is being supported. The sketched approach provides good opportunities for an automation of task division that is sufficiently flexible to deal with arbitrary configurations of workers.

The document representation is crucial in the presented concept. For the server application Océ Doc Works that is introduced to bridge the gap between end users and the print room, this representation is intentionally flexible and strictly aligned with the current capabilities of the print room. For the Océ DPS controller the document representation is fixed, but sufficiently generic to support all Océ small format devices.

The concrete division of tasks over human, server application, and controller is driven by discerning between client roles, worker roles, and workflow management that consists of job scheduling and job dividing. Handling of analogue documents, too complex document representations, or tasks that are too difficult to automate constitute reasons for involving humans in the workflow. The coexistence of this involvement of humans and the automation of other tasks is well supported by the sketched approach.

It has been argued that job tracking is more easily implemented when following this approach, because the actual document states provide the necessary information. The development of PDF as a successor of PostScript yields a positive contribution to this, whereas some concepts in JDF are less useful in our approach. The automation of the job divisor task at server level is well facilitated by the presented concepts.

References

1. JDF Specification, Spiral 6.0, Candidate for Release Version 1.0. International Cooperation for Integration of Processes in Prepress, Press and Postpress, CIP4, 2001.
2. Océ DPS400 and Océ Doc Works, product descriptions at the corporate web-site, http://www.oce.com/products/.

Biography

Roelof Hamberg got his Ph.D. in theoretical high-energy physics in 1991. He joined the vision group of the Institute for Perception Research in Eindhoven. Research topics were the evaluation of image quality and modeling relations between physical quantities and dimensions of perceived quality. In 1997, this changed to topics in user-centered design of interactive systems. In 1998, he joined R&D of Océ-Technologies, where relations between end-user's tasks and technical architectures are main topics of research.