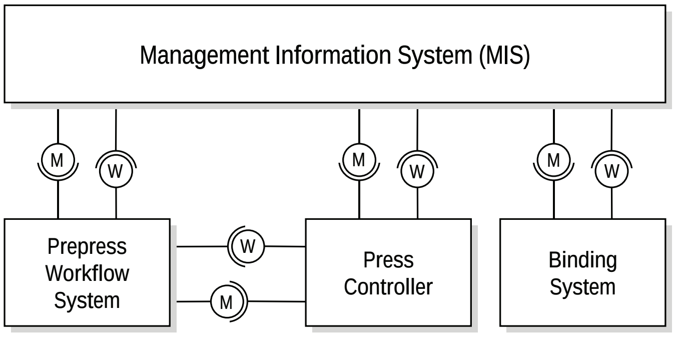
Linköping Studies in Science and Technology
Thesis No. 1285

Integrating Systems in the Print Production Workflow

Aspects of Implementing JDF

Claes Buckwalter



LIU-TEK-LIC-2006:66
Digital Media Division
Department of Science and Technology
Linköping University, SE-581 83 Linköping, Sweden

Norrköping 2006

Integrating Systems in the Print Production Workflow
Aspects of Implementing JDF

© Claes Buckwalter 2006

Department of Science and Technology
Campus Norrköping, Linköping University
SE-601 74 Norrköping, Sweden

ISBN 91-85643-26-2 ISSN 0280-7971

Printed by LiU-Tryck, Linköping, Sweden, 2006

Abstract

The print production workflow consists of various disparate systems—from production equipment to management information systems. During the production of a printed product, information regarding the product must be communicated between the systems in the workflow. Job Definition Format (JDF) is an industry standard that specifies this information interchange. It specifies a digital job ticket format for exchanging administrative and technical information related to a print job, and a messaging protocol for communicating information between the systems in the workflow. This licentiate thesis explores different aspects of integrating systems in a JDF-enabled print production workflow.

Paper III and Paper IV analyze the properties of JDF's messaging protocol—Job Messaging Format (JMF)—and discuss design solutions for a JMF integration layer.

Paper I presents a software tool for simulating systems in the print production workflow. The tool is based on an open source software library, called *the Elk Framework,* which has been developed within the framework of these licentiate studies. The Elk Framework provides the base services required by a piece of JDF-enabled production equipment, called a Device/Worker in JDF parlance.

Paper II presents a software tool that was developed for testing the simulation tool presented in Paper I. The test tool, named *Alces*, can be used for testing if JDF-enabled systems conform to the JDF Specification.

Contents

List of Enclosed Papers

i. **A JDF-enabled Workflow Simulation Tool**
TAGA 57th Annual Technical Conference 2005
Toronto, Ontario, Canada; April 17-20, 2005; p. 271-281

ii. **A Tool for Testing Compliance with**
CIP4's Interoperability Conformance Specifications
2005 International Conference on
Digital Production Printing and Industrial Applications
Amsterdam, The Netherlands; May 9, 2005; p. 83-84; ISBN/ISSN 0-89208-256-9

iii. **Integration Patterns Applied to Print Production**
TAGA 58th Annual Technical Conference 2006
Vancouver, British Columbia, Canada; March 19-21, 2006; p. 54-68

iv. **A Messaging-Based Integration Architecture for**
Print Production Workflow Systems
Printing Technology SPb'06
Saint Petersburg, Russia; June 26-30, 2006; p. 13-17

Related Work

i. **The Elk Framework**
An open source library written in Java that provides the generic infrastructure and services needed by a JDF Device.
Documentation and source code: http://elk.itn.liu.se

ii. **Alces**
Alces is a tool for testing JDF software. Alces plays the role of a Manager and is used for testing the JDF compliance of a Worker, such as a RIP, a printing press, or a binding device.
Documentation and source code: http://elk.itn.liu.se/alces

List of Figures

Preface

When starting out, the focus of my licentiate studies was loosely specified as "studies within the area of prepress, color management, and the graphic arts production process in particular". I had an idea that it would be interesting to specialize in technologies for automating the print production process and integrating the various systems involved in print production. I also knew that I found it stimulating to, and had bit of a knack for, building software systems. Hopefully there would be a way for me to combine these areas.

Before I started my licentiate studies, an emerging industry standard, named *Job Definition Format (JDF)*, that promised to integrate the systems in the print production workflow, had caught my attention. Shortly after I had commenced my licentiate studies, T2F[1] gave me the opportunity to visit *Drupa 2004*, THE tradeshow for everything related to print, held in Düsseldorf, Germany, every four years. I knew that CIP4[2], the industry consortium that developed and maintained JDF, would be at Drupa so I decided to contact them and discuss the possibility of collaborating.

Thanks largely to the marketing departments of the industry's giants, Drupa 2004 become known as "the JDF Drupa". Almost all vendors had "JDF" on their product feature lists. In reality though, many products only had very rudimentary JDF support. CIP4 was aware of this and welcomed contributions that could help speed up the adaptation of JDF and increase the number of products with mature JDF implementations. My proposal to create a reference implementation of a JDF-enabled system was warmly welcomed.

From Drupa on, my licentiate studies have more or less centered around different design approaches to an integration layer for JDF enabling systems, developing test tools for testing JDF-enabled systems, and analyzing JDF from the perspective of generic design patterns for system integration. If I were to attempt to put a label on my work I would call it computer science applied to the graphic arts industry. My licentiate studies have been an opportunity for me to be creative and use software as a medium of expression to help solve interesting problems for an industry in transformation.

My work on JDF and collaboration with CIP4 has been very rewarding. I have had the honor of doing an internship under the supervision of CIP4's chief technical officer Dr. Rainer Prosi, in Kiel, Germany; I have had the opportunity to attend CIP4's technical interoperability conferences in Heidelberg, Pittsburgh, Strasbourg, Tokyo, and Quebec; I have served as chairman of CIP4's Tools & Infrastructure working group; and I have had the pleasure of meeting a lot of intelligent and inspiring people from all around the world.

[1] *Print Technology Research/TryckTeknisk Forskning*, http://www.t2f.nu

[2] *The International Cooperation for the Integration of Processes in Prepress, Press and Postpress Organization (CIP4)*, http://www.cip4.org

I am grateful to all people who have supported me during my licentiate studies. In particular, I would like to thank my supervisor professor Björn Kruse for his guidance and pep-talks; my CIP4 colleagues for their valuable input and sharing; the contributors to my open source projects, in particular Marco, Markus and Ola; my colleagues at ITN for the “fika” breaks—hopefully the horrible coffee hasn’t done too much permanent damage to our bodies; my family and friends for their support; and my girlfriend Susanne for being there.

Finally, the financial support from T2F, CIP4, and ITN is gratefully acknowledged.

1 Introduction and Overview

1.1 Introduction

A technical white paper or article on JDF typically starts with motivating the existence of JDF by mentioning something about the printing industry suffering from shorter run lengths and the need to optimize the print production workflow. This thesis will not. Instead, it is assumed that the reader is already convinced of the need for JDF; or if the reader is not convinced, simply that he has had to come to accept the fact that JDF is an industry standard that has come to stay.

The reader who will probably find this thesis the most interesting is a software developer, not necessarily with a background in developing software for the printing industry, who is trying to get his head around the main concepts in JDF and is looking for inspiration for design approaches for JDF-enabling his software.

1.2 Overview

This licentiate thesis consists of two parts. The purpose of the first part is to give the reader the context in which the licentiate studies have been conducted. It gives a brief overview of the history of system integration in the printing industry, and an introduction to the industry standard *Job Definition Format (JDF)*. Chapter 2 gives an overview of print production and the information flows in a printing plant. In Chapter 3, Job Definition Format (JDF) is introduced. Chapter 4 describes the concepts behind JDF's digital job ticket format, while chapter 5 describes the different system roles in a JDF-enabled print production workflow. Chapter 6 presents two software tools for building JDF-enabled systems that have been developed within the framework of this licentiate thesis project—*Alces* and *the Elk Framework*. Chapter 7 provides a summary of the papers included in the second part of this thesis. Finally, chapter 8 discusses the work presented in the papers and suggests future extensions of the work.

The second part of this thesis consists of the four main papers published during these licentiate studies. *Paper I* describes a JDF-enabled simulation tool based on the Elk Framework. *Paper II* explains the motivation for and design principles behind Alces. *Paper III* and *Paper IV* build on the experiences learned from the development of Elk and Alces and discuss various approaches to implementing a software layer for JDF-enabling systems in the print production workflow.

2 Background

Before delving into the details of *Job Definition Format*, this chapter provides an introduction to print production, the information flows in a printing plant, and the most important standards that preceded JDF.

2.1 Print Production

A printing plant is typically organized into three production departments: prepress, press, and postpress. Overseeing these three departments is a fourth, management-related, department that handles incoming customer orders, plans production, manages inventory, and is responsible for other administrative tasks. Figure 2-1 provides an overview of the departments and the information flow between them.

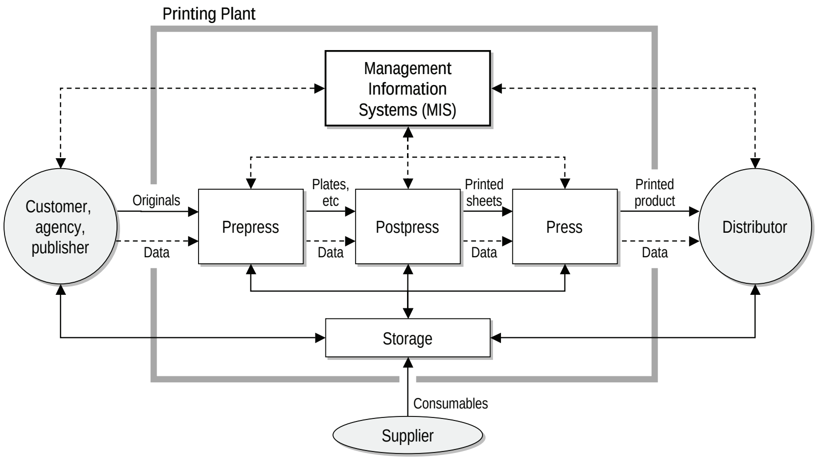


Figure 2-1 Overview of the production flow in a printing plant (inspired by [Kipphan 2001, Figure 1.2-1])

A prepress department prepares customer source material for print. In terms of inputs and outputs, the inputs to a prepress department are the customer's specification of the printed product and the customer's source material, today almost exclusively in digital form, such as PDF. Outputs are printing plates and configuration parameters for the printing press.

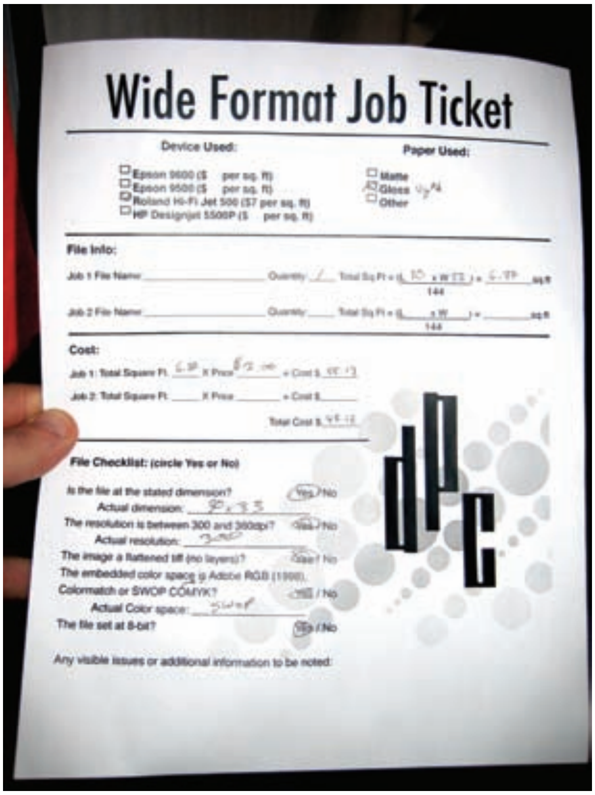
A press department takes the output from the prepress department as input and produces printed sheets of paper. The printed sheets are input to the postpress department that cuts, gathers, folds, and binds them. The postpress department outputs finished printed products.

Traditionally, the three production departments have been relatively isolated from each other, with a minimum of information being communicated between them. The communication between the production departments and the management department, that in theory should have a holistic view of the whole printing plant and up to date information about all customer orders currently in production, has been even more non-existent. The only time information traveled between departments was when one department handed off the printed product to another department in the form of a *job ticket*.

2.2 Job Tickets

In the printing industry, the term *print job*, or *job*, is used to refer to a customer order to produce a printed product. During the production of a job, a *job ticket* is sent around the printing plant together with the printed product and is used to collect technical and administrative information related to the job, such as product specification, customer contact information, scheduling, and equipment configuration parameters [Kipphan 2001, Section 1.9.2.2]. The arrows marked "Data" in Figure 2-1 represent job tickets being transferred between the printing plants departments.

Traditionally, the job ticket was a collection of papers (see Figure 2-2) in a folder. This resulted in error-prone and time-consuming re-entering of information when a job moved between systems. It also made it very difficult to overview the status of all jobs in production since that required manually collecting and analyzing all job tickets.


Wide Format Job Ticket

Device Used:
- ☐ Epson 9600 ($ per sq. ft)
- ☐ Epson 9500 ($ per sq. ft)
- ☑ Roland Hi-Fi Jet 500 ($7 per sq. ft)
- ☐ HP Designjet 5500P ($ per sq. ft)

Paper Used:
- ☐ Matte
- ☑ Gloss [illegible]
- ☐ Other

File Info:

Job 1 File Name: ______ Quantity: 1 Total Sq Ft = (L 10 x W 72) / 144 = 6.79 sq ft

Job 2 File Name: ______ Quantity: ______ Total Sq Ft = (L ___ x W ___) / 144 = ______ sq ft

Cost:

Job 1: Total Square Ft. 6.79 X Price $7.00 = Cost $ 47.53

Job 2: Total Square Ft. ______ X Price ______ = Cost $ ______

Total Cost $ 47.53

File Checklist: (circle Yes or No)

Is the file at the stated dimension? [✓ Yes] No
Actual dimension: 90 x 33
The resolution is between 300 and 360dpi? [✓ Yes] No
Actual resolution: 300
The image a flattened tiff (no layers)? [✓ Yes] No
The embedded color space is Adobe RGB (1998), Colormatch or SWOP CMYK? [✓ Yes] / No
Actual Color space: SWOP
The file set at 8-bit? [✓ Yes] / No

Any visible issues or additional information to be noted:

Figure 2-2 A job ticket for specifying a job for a wide format printer

As computer-integrated manufacturing (CIM) started to make its way into print production in the 1990s [Kipphan 2001, Section 1.2.5] and computer-controlled production equipment became more common, job tickets were digitized.

A *digital job ticket format* is a data format for exchanging print job definitions, and information related to the production of print jobs, between the various production and management systems in a printing plant. Initially, the scope of the open digital job ticket formats did not cover all aspects of print production and different vendors in the printing industry developed their own proprietary digital job ticket formats. Examples of three such formats are *Portable Job Ticket Format (PJTF)* [Adobe 1999], *Print Production Format (PPF)* [Daun et al. 1998], and *IfraTrack* [Ifra 2002]—all of which have influenced the development of JDF.

2.3 Job Tracking

In order to be able make well-grounded decisions related to production, the management of a printing plant needs to have up-to-date information about all jobs in production and be able to monitor the status of all systems in the printing plant. This requires that there is some mechanism in place that allows for the management information systems to track the progress of jobs through production and to monitor the various production systems involved.

Certain sectors in the printing industry, such as catalog printing and packaging printing, have early on adopted a view of their production as a manufacturing process, as opposed to the craftsmanship view used by most of the printing industry. The newspaper industry is another sector that was early to adopt production concepts from the manufacturing industry.

The newspaper industry works under the pressure of tight schedules—each day a new product must be produced and the deadlines met. Real-time updates from production systems are crucial for providing an up-to-date view of the production status of a newspaper. In 1994, *IFRA*[3], an international newspaper association, published a vendor-independent specification, *IfraTrack*, which defined how status and management information should be exchanged between the systems in the newspaper production workflow. The most recent version of the specification, *IfraTrack 3.0*, was published in 2002 [Ifra 2002].

As commercial printing has evolved there has been an increasing demand for the same level of job tracking and monitoring that has been available in the newspaper industry for more than a decade.

[3] Ifra, http://www.ifra.com

3 Job Definition Format (JDF)

Job Definition Format (JDF) is an industry standard that specifies how to integrate the various heterogeneous systems in the print production workflow:

"JDF is an industry standard designed to simplify information exchange between different applications and systems in and around the graphic arts industry." [CIP4 2005b]

The JDF specification [CIP4 2005b] is essentially a middleware specification for the printing industry. It specifies a job ticket format for exchanging administrative and technical information related to a print job, and a messaging protocol for communicating information between the various systems in the print production workflow. In other words, the JDF Specification specifies what information systems should exchange and how they should exchange it.

The JDF job ticket format allows the customer's intent of the printed product to be defined. It also allows the printing plant's production view—what work steps are required to produce the customer's product—to be defined. Chapter 4 explains the concepts behind JDF's job ticket format without going into the details of how the information is represented. For a complete description of the syntax of the JDF job ticket format the reader is referred to the JDF specification [CIP4 2005b].

JDF's messaging protocol—*Job Messaging Format (JMF)*—is used by the systems in the print production workflow to communicate job tracking information; monitor the various production systems, such as prepress equipment, printing presses, finishing equipment; and to a certain extent, control the production systems. An overview and analysis of JMF is given in Paper III and Paper IV. For a complete description, the reader is referred to the JDF specification [CIP4 2005b].

3.1 History and Prior Integration Standards

JDF is not a first attempt at integrating the systems in the print production workflow, but rather the best practices learned from previous attempts.

Portable Job Ticket Format (PJTF) was a proprietary job ticket format developed by Adobe for exchanging processing instructions between systems in prepress. Another important job ticket format that preceded JDF was *Print Production Format (PPF)* [Daun et al. 1998], developed by an industry consortium named CIP3[4]. PPF is used to store and transfer information generated at different production steps in prepress, press, and postpress. A typical usage example is to use PPF data (colloquially referred to as "CIP3 data") generated in prepress to configure the ink keys on a printing press.

The newspaper industry's job tracking specification *IfraTrack* has also influenced the development of JDF, particularly its messaging protocol Job Messaging Format (JMF), used to monitor the systems on the plant floor and track print jobs in production.

It is mainly the models and concepts used by PPF, PJTF, and IfraTrack that have influenced JDF. In many cases there are direct mappings between the data that can be

[4] The International Cooperation for Integration of Prepress, Press, and Postpress (CIP3)

represented in PPF, PJTF, or IfraTrack to a representation in JDF. The structure, syntax, and methods of exchanging data between systems used by preceding standards are however not shared by JDF. Previous attempts used proprietary data formats that required specialized programming tools, and largely relied on embedding metadata in large content files, which required a lot of resources to process. JDF is based on a generic data format named XML [W3C 2006] and data is exchanged between systems using self-contained files and standard network protocols [Fielding et al. 1999].

The first draft of JDF (officially named JDF 1.0) was developed by a group consisting of Adobe[5], Heidelberg[6], Agfa[7], and MAN Roland[8]. In 2001 the group handed over the draft of JDF to CIP4, which in 2000 had been formed out of CIP3. The first version of JDF that could actually be implemented was JDF 1.1, published in 2002. JDF 1.2 was presented at Drupa[9] 2004 and received a lot of attention in the media thanks to effective marketing from CIP4 and its member organizations. At the time of writing JDF has reached version 1.3.

3.2 The CIP4 Organization

JDF is developed and maintained by an industry consortium named CIP4—*The International Cooperation for the Integration of Processes in Prepress, Press and Postpress Organization.* CIP4 publishes the JDF specification [CIP4 2005b] and develops a number of tools that can be used for building systems that use JDF.

CIP4 is an open organization where all members are welcome to actively participate. The organization's web site [CIP4] acts as a hub through which all members communicate and where all information produced by the organization is collected. CIP4's work is conducted in working groups that specialize in different areas of the print production workflow, for example prepress, flexography printing, digital printing, or binding. It is mostly vendors that develop products for a specific area of the print production workflow that contribute to the corresponding CIP4 working group. Each working group normally meets every other week using a web-based phone conference system.

The working groups suggest changes to the JDF specification and submit them to CIP4's *Technical Steering Committee (TSC)* who decides what should be included in the next version of the JDF specification. The TSC consists of the chairmen of the working groups.

Twice a year a one-week *technical interoperability conference*—"interop" for short—is hosted by one of CIP4's member organization. The purpose of an interop is for CIP4's members to meet to test the interoperability of their JDF-enabled systems, and to have

[5] Adobe Systems Incorporated, http://www.adobe.com

[6] Heidelberger Druckmaschinen AG, http://www.heidelberg.com

[7] Agfa-Gevaert Group, http://www.agfa.com

[8] MAN Roland Druckmaschinen AG, http://www.man-roland.com

[9] Drupa, http://www.drupa.com

face-to-face working group meetings. The atmosphere at CIP4's interops is very informal; it is a perfect opportunity for developers to meet peers and exchange experiences related to implementing JDF.

3.3 The JDF Specification and its Ancillary Specifications

The JDF specification describes the structure and syntax of the JDF job ticket format, the syntax of Job Messaging Format (JMF), the roles of the systems in a JDF-enabled workflow, the routing of jobs through the workflow, the execution of jobs, etc.

Although the 900 page JDF specification provides a complete specification of the different constructs available for defining a print job and how it is to be produced, the specification often allows for more than one way of describing a certain aspect of a print job. This leads to different implementations of the JDF specification providing the same functionality but in different ways, thus hampering interoperability between systems.

To alleviate this problem CIP4 provides a collection of ancillary specifications called *Interoperability Conformance Specifications (ICS)*. These specifications specify subsets of the JDF specification that are to be used when two systems in a specific sector of the print production workflow exchange information. For example, there is an ICS that defines the minimum subset of the JDF job ticket format that a prepress system must use when sending a job to an offset printing press [CIP4 2005a]. When implementing a JDF-enabled system a developer should always start by identifying what ICSs the system must comply with.

All published ICS documents are publicly available on CIP4's web site [CIP4 2006e]. Section 5.2 provides a more detailed description of the ICS documents and system interoperability.

4 The JDF Job Ticket Format

The JDF job ticket format uses a hierarchical tree structure to describe a print job. The structure is very similar to a *Work Breakdown Structure (WBS)* [Haugan 2001] with deliverables and tasks, going from general to specific. At the root of the tree is a specification of the printed product to produce—*product definition*. At the leaves of the tree are definitions of the activities required to produce the printed product—*process definitions*.

The JDF job ticket format is based on *XML* [W3C 2006] and uses *XML Schema* [W3C 2004] to describe its syntax and structure. Extensible Markup Language (XML) is a general syntax for specifying markup languages. The advantage of XML is that there are widely available tools and programming libraries for processing any markup language that uses XML syntax. This has made XML the de facto data exchange format for text-based information, and is why JDF's job ticket format is encoded in XML.

We now leave XML and instead focus on the print job model of the JDF job ticket format from a higher level, explaining the main concepts behind the model, not how it is represented in XML.

4.1 Product Model

As mentioned above, the JDF job ticket format models a print job as a tree of *JDF nodes*. In the general case, the root JDF node of the tree is a *product node* that represents a specification of the printed product that the customer wants produced—the customer's *intent* of the product. If the product can be divided into sub-parts that can be given separate but more specific specifications, these sub-parts are represented as product nodes that are children of the root product node. All product nodes in the tree can be recursively divided into sub-parts represented by child product nodes.

The number of levels of product nodes, i.e. the level of granularity of the product specification, depends on the product. In commercial printing a job usually has two levels of product nodes. For example, a customer may want to print a book. The root product node may specify that 500 copies of a book should be printed, that the spine should be glued, and that the finished books should be delivered to a certain address. The root product node has two child product nodes. One child node specifies the media that should be used for the cover, and that the cover should be printed in color; the other child node specifies the media that should be used for the body of the book, and that the body should be printed in black and white. Figure 4-1 shows a possible job structure for a book job taken from an example in the JDF Specification [CIP4 2005b, Section 2.2.1]. Nodes 1, 2, and 3 are product nodes.

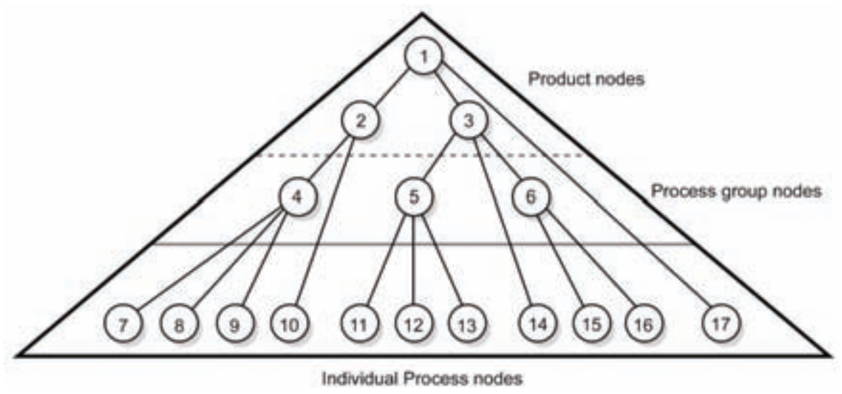


Figure 4-1 JDF job ticket tree structure [CIP4 2005b, Figure 2-2]

4.2 Process Model

The tree structure of the JDF job ticket format describes, from top to bottom, deliverables and the work steps required to produce the deliverables. Deliverables are represented by product nodes. The work steps required to produce a product node are represented by three different types of JDF nodes: *process nodes*, *combined process nodes*, *process group nodes*. The sections that follow describe these three node types in bottom-up order.

4.2.1 Process Nodes

A process node is always a leaf in a job tree and defines the most fine-grained work step (activity) that can be modeled in JDF. The JDF process model is based on the producer-consumer model—a process consumes input resources and produces output resources.

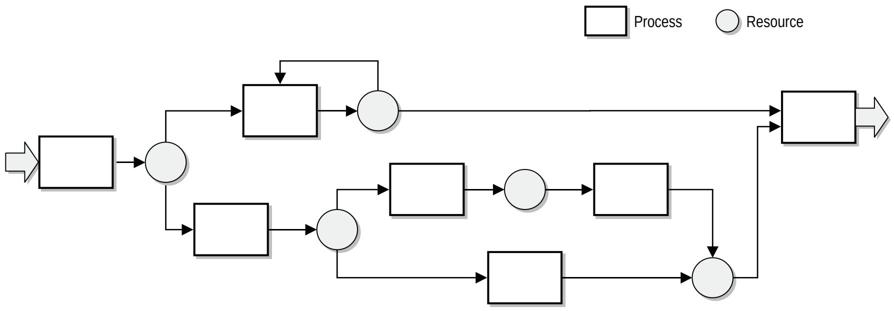


Figure 4-2 Process network

Processes are linked together by their input and output resources. A process can start execution as soon as all, or an adequate subset, of its input resources become available. During process execution input resources are consumed, transformed, or combined to produce output resources. The output resources of one process are the input resources of one or more other processes. As Figure 4-2 illustrates, the linking of processes using resources creates a network of processes allowing serial, parallel, overlapping, and iterative processing to be modeled [CIP4 2005b, Section 4.3; Enlund 1998].

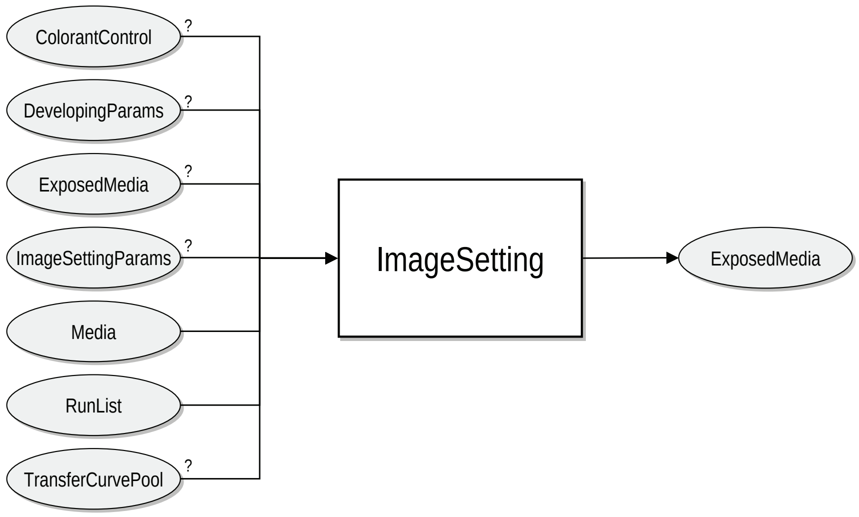


Figure 4-3 The ImageSetting process

The current version of the JDF specification defines more than 90 processes and more than 190 resources that can be used as input or output resources to the processes; most of these are related to prepress and postpress. Figure 4-3 shows the JDF process *ImageSetting* [CIP4 2005b, Section 6.4.14], and its input and output resources. Image setting is a prepress process that exposes a bitmap image on to a film or a plate. The process is typically performed by a device called an imagesetter [Kipphan 2001, Section 3.2.12.1].

4.2.2 Combined Process Nodes

A *combined process node* is a leaf node used for defining several processes that are executed as one atomic process by a single system. For example, a digital printing system typically interprets page descriptions and instructions, renders pages, and then prints the rendered pages on paper. The *Integrated Digital Printing ICS* [CIP4 2006b] requires that conforming digital printing systems be able to execute a combined process node consisting of the following JDF processes: *LayoutPreparation*, *Imposition*, *Interpreting*, *Rendering*, and *DigitalPrinting*.

Before a combined process can execute, all input resources that provide parameters for its internal process must be defined and available. Intermediate resources produced and consumed by the combined process's internal processes do not need to be defined.

4.2.3 Process Group Nodes

A process group node is used to create logical groupings of process nodes, for example grouping together processes in prepress, press, and postpress groups, see Figure 4-1. This allows a system to provide an abstract declaration of a group of processes without defining all the child process nodes. A process group node can also be used to define resources that are used by its descendant nodes, thus removing redundancy. Child nodes of a process group node can be process nodes, combined process nodes or process group nodes.

4.2.4 Gray Boxes

A special case of process group node is a *Gray Box.* A Gray Box is used to specify an ordered process chain consisting of a minimum set of processes, and the input and output resources of the process chain. This allows a sequence of processes to be defined without needing to define all intermediate processes or resources.

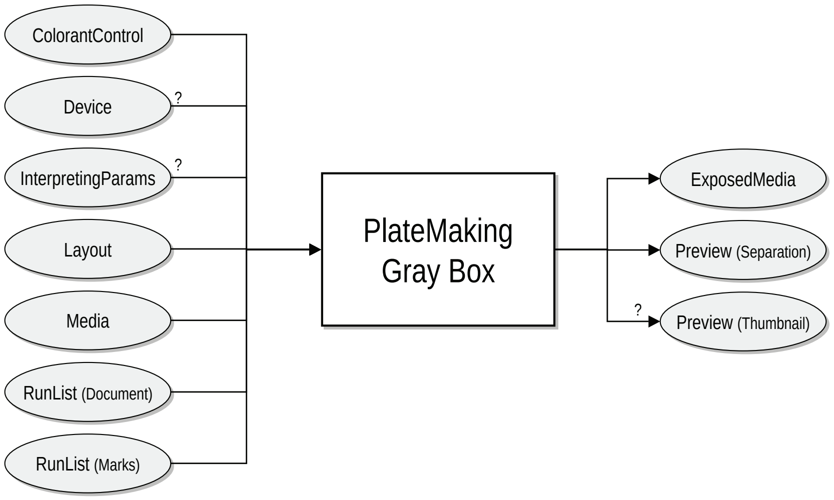


Figure 4-4 Gray Box for plate making

Gray Boxes were first developed for communication between MIS and prepress workflow systems [CIP4 2006g, Section 2]. MIS systems usually do not have knowledge about all production details and often write Gray Boxes instead of explicit process definitions to a JDF job ticket. When a production system receives a JDF job ticket with a Gray Box that it can interpret it “expands” the Gray Box by adding process nodes to the Gray Box process group, linking the processes as necessary. The algorithm for Gray Box expansion is defined in [CIP4 2005b, Section 3.2.2.1].

To summarize, Gray Boxes provide a way of defining a sequence of processes without needing to have detailed knowledge of all the processes and their resources.

4.3 Resource Model

In JDF, resources represent the "things" that are the inputs and outputs of processes. Resources are what link processes together to create the network of processes, see Figure 4-2, that represent the workflow for producing a print job.

During the production of a print job, the resources used go through a number of state changes. For example, when a resource is ready for consumption by a process it changes state to *Available*; when a resource is in use by a process it has the state *InUse*. The resources' states determine when a process can be executed.

4.3.1 Resource Classes

There are five main classes of resources: *parameter resources*, *physical resources, intent resources*, *implementation resources*, and *placeholder resources*.

Parameter resources are configuration parameters or input data, such as files, for a process. For example, *ImageSettingParams* is a parameter resource used to configure an imagesetter, see Figure 4-3.

Physical resources are divided into three sub-classes: *consumable resources*, q*uantity resources*, and *handling resources*. Consumable resources are consumed by processes; for example, *Ink* and *Media* resources are consumed during printing. A consumable resource can be used to create a quantity resource. Quantity resources are semi-finished goods, such as printed sheets from a printing press. Handling resources are resources that are used by a process but not destroyed during the execution of a process; for example, plates (*ExposedMedia*) for a printing press.

Intent resources are used together with product intent JDF nodes (see Section 4.1) to specify various aspects of the printed product to produce. For example, the intent resource *MediaIntent* can be used to specify the characteristics of the paper the product should be printed on.

Implementation resources define the equipment and personnel needed to execute a process. There are only two resources of this class: *Employee* and *Device*.

Placeholder resources are used as temporary stand-ins for other resources. They are used when the actual resource produced or consumed by a process group (see Section 4.2.3) is unknown.

4.3.2 Partitioning of Resources

In a similar way to the product and process models, the resource model of JDF is also hierarchical. Resources of the same type can share properties by having the properties factored out and placed in a parent resource. Child resources can then inherit the parent's properties or choose to override them. This hierarchical grouping of resources of the same type is called *partitioning*.

Partitioning is used to minimize the number of resources and process nodes used to describe how to produce a print job. For example, the *ImageSetting* process (see Figure 4-3) is used to produce a printing plate, represented in JDF by the *ExposedMedia* resource. If four printing plates were to be produced (one for each separation cyan, magenta, yellow, and black) for one side of a sheet, one alternative would be to have four *ImageSetting* processes, each one outputting one *ExposedMedia* resource. Instead, the recommended alternative is to define one *ImageSetting* process that outputs one partitioned *ExposedMedia* resource. The resource's partitions are four *ExposedMedia* resources, one for each printing plate. Each partition inherits all properties (represented by XML attributes and child elements) defined in the parent resource.

4.3.3 Resource Linking

Resources connect processes and determine the sequence of processes that will be executed in order to produce a printed product.

The resources required to produce a product are defined in so-called *resource pools* that are associated with each JDF node in a job tree. A JDF node can use any resource defined in its own resource pool and the resource pools of its ancestor JDF nodes.

Resource pools are only used to house the definitions of resources. The linking of input and output resources to a JDF process node, process group node, or combined process node is defined using *resource links* in the JDF node's *resource link pool*. A resource link references a resource defined in a resource pool and specifies if the resource should be used as input or output to the JDF node. Several JDF nodes can link to the same resource; for example, one process produces a resource as an output and another process uses the resource as an input.

As mentioned in Section 4.3.2 above, resources are hierarchical and can be divided into partitions. Resource links can refer to a specific partition of a resource, for example a specific printing plate.

Physical resources (Section 4.3.1) exist in amounts, such as a stack containing 1000 sheets of paper. A resource link can reference a specific amount of a physical resource. This allows a process to produce a resource while one or more other processes consume the same resource. This is called overlapping processing and the resource linking mechanism used is called a *pipe*. [CIP4 2005b, Section 4.3.3] gives a detailed description of overlapping processing and pipes.

4.4 Comparison with other Workflow Models

An excellent but somewhat dated overview of workflow management in the graphic arts and printing industry is given in [Enlund 1998]. The paper concludes by suggesting a merge of CIP3's and IFRA's standards—something that today has been realized in the form of JDF.

As indicated above, the print production workflow is resource driven. The different activities required to produce a printed product depend on the availability of resources. A printing plate cannot be created until the pages used for the plate are available; the press cannot start printing before the plates are ready; the printed product cannot be bound before all pages have been printed.

Modeling print production with generic workflow models or using more low-level theoretical models is in some cases not possible or becomes overly complex [Tuijn

2004]. In particular, overlapping processing with partially produced objects/resources is difficult to model.

The resource model used by JDF considers all "things" needed to execute a process as resources. Generic workflow models such as the Workflow Management Coalition's (WfMC) [WfMC 1999] typically differentiate between objects and resources, see Figure 4-5. "Objects" are the inputs and outputs of a process, while "resources" correspond to JDF's implementation resources—the equipment and personnel required to execute a process. An example of a model from the printing industry that also separates objects and resources is the newspaper-manufacturing model described in [Nordqvist 1996].

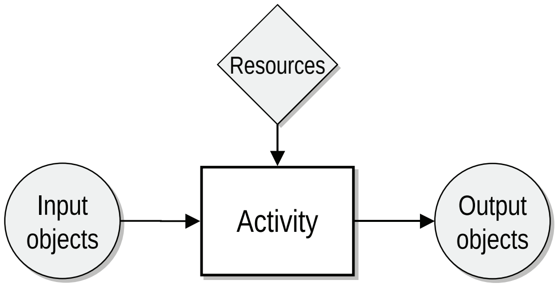


Figure 4-5 Generic process/activity model

5 JDF-enabled Workflows and their Systems

5.1 Workflow Component Roles

A printing plant's production workflow consists of various systems, from management information systems to production equipment in prepress, press, and postpress. The JDF Specification [CIP4 2005b] refers to these systems as *workflow components* and arranges them in a hierarchy. Each system in this model is classified as having one or more of the following *workflow component roles*:

5.1.1 Agent

An *Agent* is a system that writes JDF instances. Most JDF-enabled systems update or add information to the JDF instances they process, and are therefore Agents.

5.1.2 Controller

A *Controller* interprets JDF instances, optionally adds specific information known by the Controller, and then routes the JDF instances to other systems for further processing. A Controller may split a JDF instance into several parts and route each part to a different system for processing—*spawning*. A Controller may also merge several JDF instances and send the new JDF instance for processing—*merging*.

In the JDF workflow system hierarchy, Controllers are located at the top levels of the hierarchy (Figure 5-1). The JDF Specification implies that there is a single monolithic controller at the root level of the hierarchy in the form of a *Management Information System (MIS)*, see Section 5.1.5. Typically, there are two levels of Controllers in the hierarchy: an MIS at the root level, and several sector specific Controllers, such as a prepress workflow system, at the level below.

5.1.3 Device

A *Device* is a system that receives JDF instances, and executes the processes specified by one or more of the process nodes or combined process nodes in the JDF instance.

An example of a Device is a JDF-enabled front-end for a digital printing press. The front-end receives JDF instances, interprets the relevant process nodes, and sends proprietary control commands to the digital printing press hardware.

5.1.4 Machine

A *Machine* is a system that receives proprietary control commands from a Device and does not know anything about JDF. In the Device example above, the digital printing press would be a Machine.

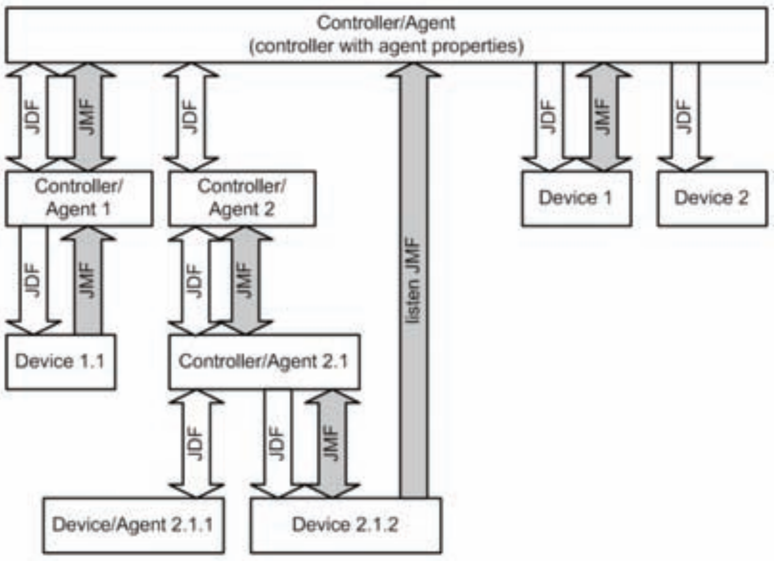


Figure 5-1 Hierarchy of workflow components ([CIP4 2005b], Figure 2-1)

5.1.5 Management Information Systems

A very important concept in a JDF-enabled workflow is the *Management Information System (MIS)*. The JDF Specification defines an MIS as: *"the overseer of the relationships between all of the units in a workflow … in effect, a macrocosmic controller. It is responsible for dictating and monitoring the execution of all of the diverse aspects of the workflow."*

Different MISs vary in complexity and feature set. An MIS normally has some or all of the following functionality [Bellander et al. 1999]:

- Estimating & quoting
- Scheduling
- Order entry & job tracking
- Raw material inventory
- Shop floor data collection
- Invoicing & receivables
- Job history
- Standardized reports
- Receivables
- Payables
- General ledger & financials

The JDF Specification and the ICS documents tend to describe an MIS is a single monolithic system. However, there could be several separate systems in a printing plant that provide different types of, or overlapping, management information system functionality. These systems go under different names [Handberg 2003], such as Production Management System (PMS) or Manufacturing Planning and Control (MPC) system. For the sake of simplicity, we will here adhere to the MIS view used by the JDF Specification and refer to the MIS as a single system.

The MIS concept used in [CIP4 2005b] in many aspects overlaps the Global Production Management System (GPMS) described by [Nordqvist 1996].

5.2 *System Interoperability*

The JDF Specification defines a hierarchical relationship between Controllers and Devices, a messaging protocol for communicating between the systems, and a job ticket file format for exchanging production data. However, it would be unrealistic to require that all systems in a JDF-enabled environment implement the entire JDF Specification. An MIS needs to have more complex JDF/JMF processing capabilities than an imagesetter; an offset printing press controller only needs to know how to execute JDF process nodes of type *ConventionalPrinting*; a folding machine only needs to know how to execute JDF process nodes of type *Folding*. Depending on in what sector of the workflow a system is located, and what it does, it will implement a certain subset of the JDF Specification.

If vendors would choose to implement arbitrary subsets of the 900 pages thick JDF Specification it is unlikely that their systems would be interoperable. To alleviate this, CIP4 provides a set of *Interoperability Conformance Specification* [CIP4 2006e] documents that specify the minimum requirements placed on systems in different sectors of the workflow [CIP4 2006a]: *"Each ICS defines a set of conformance requirements that a conforming JDF-enabled product must meet in order to achieve interoperability with other conforming JDF-enabled products."*

All ICSs build on the *Base ICS* [CIP4 2006a]. It extends on the JDF Specification and specifies the minimum level of JDF and JMF functionality that any JDF-enabled system must have—what values in JDF job tickets and JMF messages a system must read and write, and what JMF messages a system must be able to send and receive.

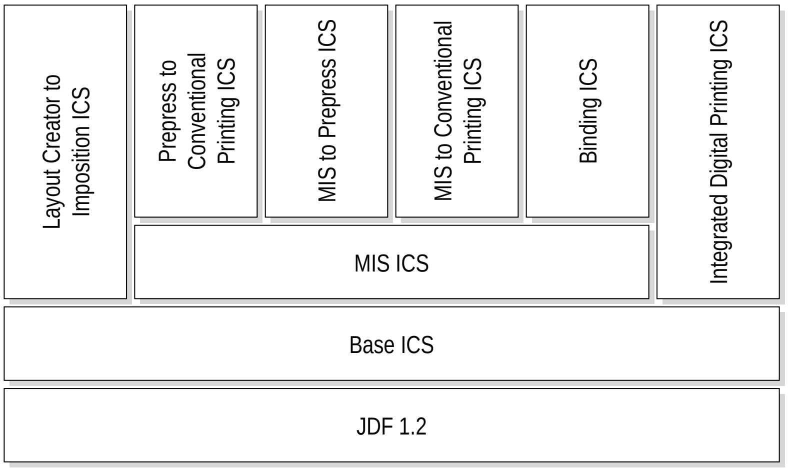

Layout Creator to Imposition ICS
Prepress to Conventional Printing ICS
MIS to Prepress ICS
MIS to Conventional Printing ICS
Binding ICS
Integrated Digital Printing ICS
MIS ICS
Base ICS
JDF 1.2

Figure 5-2 ICS Stack

The other ICSs extend the Base ICS as illustrated in Figure 5-2. They define conformance requirements for specific pairs of systems that interoperate in a specific context. For example, an MIS and a prepress system that communicate must implement the *MIS to Prepress ICS* [CIP4 2006g].

5.2.1 System Roles and Interfaces

When a system conforms to an ICS it conforms to the ICS in the capacity of one of two roles—*Manager* or *Worker*—defined in the Base ICS [CIP4 2006a]. As mentioned earlier, there is a hierarchical relationship between the systems in a JDF-enabled environment. A system that is a Controller and delegates work to other system splays the role of Manager, while the systems that do the work are Workers.

The MIS to Prepress ICS [CIP4 2006g] specifies conformance requirements for the communication between MIS systems and systems in prepress. The ICS specifies that the MIS is the Manager and the prepress system is the Worker.

A JDF-enabled system may conform to multiple ICSs and may play the role of both Manager and Worker. A prepress workflow system is a typical example. A prepress workflow system conforming to the MIS to Prepress ICS plays the role of a Worker—it receives gray boxes from a MIS (Manager). However, in the context of the *Prepress to Conventional Printing ICS* [CIP4 2005a] the same prepress workflow system is a Manager—it notifies a press controller (Worker) about the availability of plates, previews, or other resources.

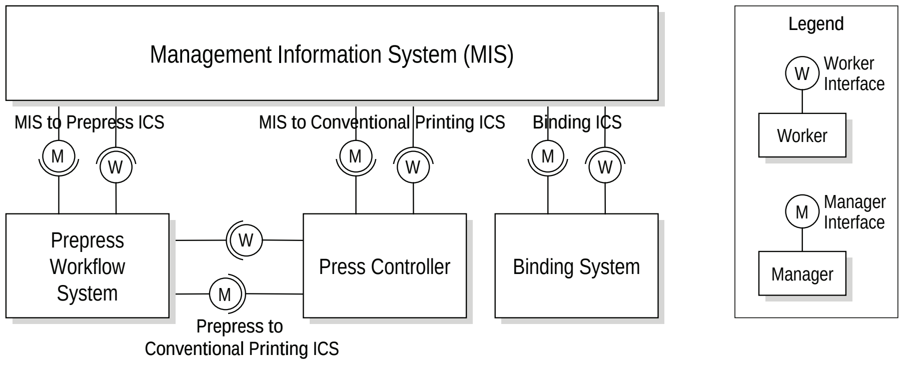


Figure 5-3 Example of systems and ICSs

In ICS parlance, a Manager implements a *Manager Interface,* and a Worker implements a *Worker Interface*. A Manager communicates with a Worker through the Worker's Worker Interface; and the Worker communicates with the Manager through the Manager's Manager Interface. However, the wording in [CIP4 2006a, Section A.2], where these concepts are defined, is extremely confusing. Hopefully the interpretation that follows, illustrated as a UML component diagram in Figure 5-3, is more clear: when a Worker communicates with the Manager it talks to the Manager's Manager Interface; when a Manager talks to a Worker it talks to the Worker's Worker Interface.

Figure 5-4 shows a more detailed hierarchy of systems, showing the Manager and Worker roles used in the ICSs, plus the workflow component roles defined by the JDF Specification and described in Section 5.1 above.

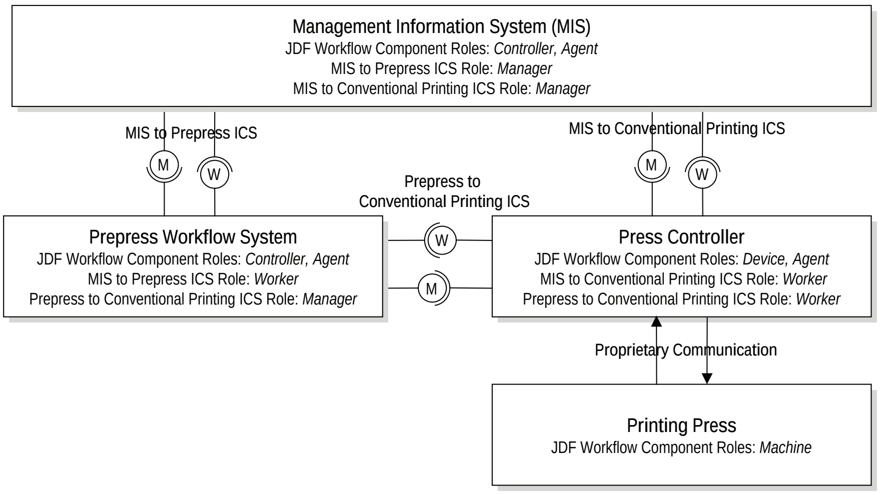

Management Information System (MIS)
JDF Workflow Component Roles: Controller, Agent
MIS to Prepress ICS Role: Manager
MIS to Conventional Printing ICS Role: Manager
MIS to Prepress ICS
M
W
MIS to Conventional Printing ICS
M
W
Prepress to Conventional Printing ICS
Prepress Workflow System
JDF Workflow Component Roles: Controller, Agent
MIS to Prepress ICS Role: Worker
Prepress to Conventional Printing ICS Role: Manager
W
M
Press Controller
JDF Workflow Component Roles: Device, Agent
MIS to Conventional Printing ICS Role: Worker
Prepress to Conventional Printing ICS Role: Worker
Proprietary Communication
Printing Press
JDF Workflow Component Roles: Machine

Figure 5-4 Example of system roles

5.3 MIS Managed Production

The MIS plays a central role in JDF. The concept of a MIS-managed printing plant is implied in the JDF Specification, and the MIS-related ICS documents explicitly describe such an environment—this is the concept that is the foundation for the *MIS ICS* [CIP4 2006f]: *"This ICS describes the data flow in a print shop in a MIS-managed environment. However, this data flow does not necessarily also apply to non-MIS-managed environments."*

In a MIS-managed printing plant the MIS is located between the customer and the production systems on the shop floor. Customers deliver product intent in the form of JDF instances with JDF product intent nodes to the MIS. The MIS has the information necessary to map the product intent to a high-level description of the workflow required to produce the product.

In the cases where an MIS has detailed knowledge about a process and the capabilities of the system that will execute the process, the MIS appends JDF process nodes to the JDF product intent nodes. However, in many cases the MIS does not have detailed knowledge about all the processes and systems required to produce a product. Instead, the MIS uses a special kind of JDF *process group nodes* called *gray boxes* [CIP4 2006f] to provide an abstract definition of a group of the processes. This is common when an MIS specifies the prepress workflow for a product. Rather than specifying all the prepress processes required, the MIS specifies a prepress gray box and then delegates the JDF instance to a prepress workflow system. The prepress workflow system has detailed knowledge of all the systems available in the printing plant's prepress department and can *expand* the gray box into detailed process definitions. The MIS to Prepress ICS [CIP4 2006g] defines the gray boxes that the MIS creates and that are expanded by the prepress workflow system.

6 JDF Tools

The *Tools & Infrastructure* working group of CIP4 is responsible for developing and maintaining a number of JDF software libraries and tools for the organization's members. The CIP4 software is distributed under an open source license [CIP4 2006c] and is available for download from CIP4's web site [CIP4 2006d]. CIP4's members may free of charge use the CIP4 software libraries, or parts thereof, in their commercial products.

6.1 JDFLib

The core software library that all of CIP4's JDF tools are based on is *JDFLib*—an application programming interface (API) for creating, reading, and manipulating JDF job tickets and JMF messages.

JDFLib is available for the Java programming language and for the C++ programming language. The Java version of the library, *JDFLib-J,* is where CIP4 first implements new features before porting them to the C++ library, *JDFLib-C*. Both JDFLib-J and JDFLib-C are used in several vendors' JDF-enabled products.

The other JDF tools developed by CIP4 are based on JDFLib-J. One of these tools is the *JDF Editor* [Andersson 2003; Thunell 2003], a visual editor for editing and visualizing JDF job ticket files and JMF message files. The JDF Editor can also be used to convert files between different JDF versions, and to validate that files adhere to the JDF specification.

Two of CIP4's JDF tools have been developed in collaboration the *Digital Media Division* at ITN, Linköping University, Sweden[10]: *the Elk Framework* and *Alces*.

6.2 The Elk Framework

The Elk Framework, colloquially referred to as Elk, is a software library written in Java that provides the infrastructure and services need by a JDF Device—a Worker (see Section 5.1.3 and 5.2.1). Using the Elk Framework, vendor's can focus on implementing the code that realizes the actual process in their Device and not worry about the generic JDF-functionality needed by all devices, such as a JMF messaging, a job queue, etc.

Elk consists of two parts: an API that defines the programming interfaces for accessing the Device services; and a reference implementation of the services defined by the API. The intention is that vendors can either use the default reference implementations of the Device services defined by the API, or plug in their own implementations of the services.

[10] Digital Media Division, ITN, Linköping University, Sweden, http://media.itn.liu.se

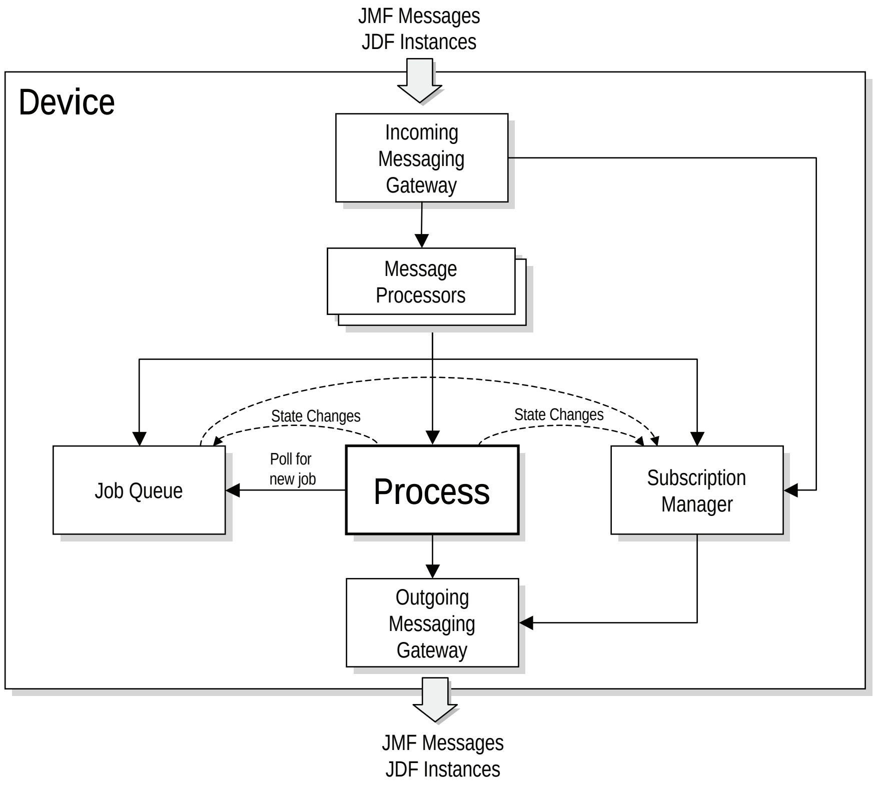


Figure 6-1 The Elk Framework

Figure 6-1 shows the components that represent the most important services provided by Elk, and the interactions between the components. The *Process* component marked in bold represents the JDF process that the Device implements. This is the component that a vendor would implement. For example, a vendor could use Elk as a front-end to JDF-enable their digital printing press. Ideally, the vendor would only have to implement the process component, which would interpret JDF job tickets and call the vendor's proprietary APIs that control the digital printing press's hardware.

Elk's reference implementation includes two Elk process components that implement simulated JDF processes: one component implements the JDF *Approval* process; and one implements the JDF *ConventionalPrinting* process. The simulated processes are intended for demonstrating the Elk Framework and for testing JDF-enabled systems that play the Manager role.

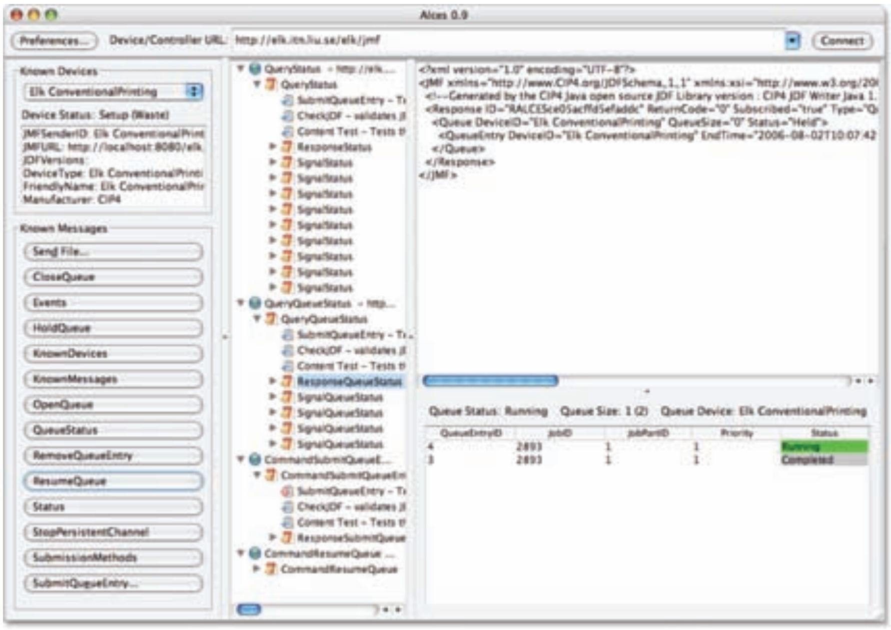
A more detailed description of the Elk Framework and the services it provides are given in *Paper I*. Extensive documentation, the library binaries, and the source code are available at the project web site [Buckwalter 2005a].

6.3 Alces

During the development of the Elk Framework a tool for testing the JMF messaging functionality of the Elk reference implementation was needed. This led to the development of the test tool Alces.

Alces plays the role of Manager (see Section 5.2.1). Using Alces, the JMF messaging functionality of a system can be tested, although Alces was originally designed for testing JDF Devices. Alces connects to a Device and can then send JMF messages to or receive JMF messages from the Device. Tests are applied to all JMF messages sent and received. Developers are encouraged to write new tests for Alces.

Alces software architecture consists of a core module that provides the infrastructure for JDF messaging and the test framework. On top of the core module two different versions of Alces are built: *Interactive Alces* and *Automated Alces*. Interactive Alces has a graphical user interface that allows a user to manually test a Device. Figure 6-2 shows a screenshot of Interactive Alces' user interface.



Device/Controller URL: http://elk.itn.liu.se/elk/jmf

Elk ConventionalPrinting

Device Status: Setup (Waste)

JMFSenderID: Elk ConventionalPrint
JMFURL: http://localhost:8080/elk
JDFVersions:
DeviceType: Elk ConventionalPrint
FriendlyName: Elk ConventionalPri
Manufacturer: CIP4

```
<?xml version="1.0" encoding="UTF-8"?>
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" xmlns:xsi="http://www.w3.org/20
  <!--Generated by the CIP4 Java open source JDF Library version : CIP4 JDF Writer Java 1.
  <Response ID="RALCESce05acffd5efaddc" ReturnCode="0" Subscribed="true" Type="Q
    <Queue DeviceID="Elk ConventionalPrinting" QueueSize="0" Status="Held">
      <QueueEntry DeviceID="Elk ConventionalPrinting" EndTime="2006-08-02T10:07:42
    </Queue>
  </Response>
</JMF>
```

Queue Status: Running   Queue Size: 1 (2)   Queue Device: Elk ConventionalPrinting

QueueEntryID	JobID	JobPartID	Priority	Status
4	2893	1	1	Running
3	2893	1	1	Completed

Figure 6-2 Interactive Alces testing the Elk ConventionalPrinting Device

Automated Alces is run from a command line and automatically sends a sequence of JMF messages specified by the user. When the tests are finished running a log file containing the test results is written. On use case for Automated Alces is in quality assurance: a continuous integration tool [Almagro and Julius 2002] could build and deploy a new version of a Device each time its source code is updated, and then automatically test the Device by running Automated Alces. CIP4 uses a similar approach for testing Elk and its other JDF tools.

Alces is described in more detail in Paper II. The source code and binaries are available for download from the project web site [Buckwalter 2005b].

7 Summary of Included Papers

7.1 Paper I

A JDF-enabled Workflow Simulation Tool

Paper I describes how a system based on *the Elk Framework* can be used for simulating an arbitrary process in a JDF-enabled print production workflow. A description of the Elk Framework's architecture and core components is given, and a first prototype of a simulated process is presented.

7.2 Paper II

A Tool for Testing Compliance with CIP4's Interoperability Conformance Specifications

During the development of the Elk Framework it was realized that a tool was needed for testing the framework's compliance with the JDF and ICS specifications. This led to the development of the test tool—later named *Alces*—presented in Paper II.

7.3 Paper III

Integration Patterns Applied to Print Production

The Elk Framework presented in Paper I and the test tool Alces presented in Paper II both communicate with other systems using Job Definition Format's messaging protocol Job Messaging Format (JMF). This paper analyzes JMF and the different types of system interactions that it supports. The result is the concepts found in JMF expressed using the patterns found in two pattern languages for system integration.

7.4 Paper IV

A Messaging-Based Integration Architecture for Print Production Workflow Systems

The Elk Framework (Paper I) and Alces (Paper II) use different design approaches for the software layer that handles JMF communication with other systems. Elk's design is biased towards handling incoming communication while Alces' design is biased towards handling outgoing communication. Paper IV builds on the experiences gained from developing Elk and Alces and uses the system integration patterns of JMF identified in Paper III to present a more flexible *JMF integration layer* that handles both incoming and outgoing JMF communication equally well.

8 Discussion

The licentiate studies documented in the form of this thesis have focused on different aspects of integrating the disparate systems in the print production workflow—from production equipment to management information systems—using software. The design approaches presented have been based on the industry standard Job Definition Format (JDF), which specifies a digital job ticket format for exchanging administrative and technical information related to a print job and a messaging protocol for communicating information between the systems in the workflow.

One concrete result of these studies is *the Elk Framework* [Buckwalter 2005a], an open source software library that provides the services needed by JDF-enabled production equipment. Programmers developing a piece of JDF-enabled production equipment, in JDF parlance a Device/Worker, can leverage the generic JDF functionality provided by the Elk Framework and instead focus on developing the core functionality of their system. At the time of writing, one commercial product under development is known to be based on the Elk Framework. The product is a JDF-enabled front-end for a digital printing system.

Besides generic JDF functionality, the Elk Framework includes two simulated JDF processes, described in Paper I and [Stering 2006]. These act as reference systems for programmers developing Devices. CIP4's product certification working group and several CIP4 members use the simulation-based Elk Devices for testing purposes. For example, one vendor uses the simulation-based Elk Devices for testing the JDF functionality of their prepress workflow system.

Another result of these licentiate studies is a test tool named *Alces* [Buckwalter 2005b]. Originally developed for testing the Elk Framework, Alces can be used for testing the JMF messaging functionality of any Device/Worker. Alces is used by several CIP4 for testing their systems.

The Elk Framework and Alces represent two opposing system roles in the JDF workflow model: Alces takes on the role of a Manager; Elk takes on the role of a Worker. Elk and Alces each contain an integration layer for JMF messaging. These JMF integration layers use different architectural designs biased towards the requirements of the different system roles Elk and Alces play. A possible continuation of the work presented in this thesis would be to design a new JMF integration layer, equally suitable for both Manager and Worker systems. Papers III and IV provide a starting point for such work.

References

Adobe. 1999. *Technical Note #5620, Portable Job Ticket Format – Version 1.1.* Adobe Systems Incorporated.

Almagro, Alden, and Paul Julius. 2002. CruiseControl – continuous integration toolkit. http://cruisecontrol.sourceforge.net. Accessed 2006-10-26.

Andersson, Anna. 2003. The CIP4 JDF Editor, the Editing Part. http://www.cip4.org/documents/jdf_overview/MasterProject_Anna_Andersson.pdf. Accessed 2006-10-25.

Bellander, Mats, Leif Handberg, and Panagiota Koulouvari. 1999. Identifying requirements for the implementation of production management systems in graphic arts companies. *TAGA 51th Annual Conference* 51, 36-54.

Buckwalter, Claes. 2005b. Alces. http://elk.itn.liu.se/alces. Accessed 2006-10-26.

Buckwalter, Claes. 2005a. The Elk Framework. http://elk.itn.liu.se. Accessed 2006-10-25.

CIP4. CIP4 Website. http://www.cip4.org. Accessed 2006-10-24.

CIP4. 2004. MIS Base ICS – Application Notes, Version 0.5 beta. http://www.cip4.org/documents/jdf_specifications/MIS_APPNOTE_Beta_0_5.pdf. Accessed 2006-10-31.

CIP4. 2005b. JDF Specification, Release 1.3. http://www.cip4.org. Accessed 2006-10-31.

CIP4. 2005a. Prepress to Conventional Printing ICS. http://www.cip4.org/document_archive/documents/ICS-PRECP-1.0.pdf. Accessed 2006-10-31.

CIP4. 2005c. MIS to Conventional Printing – Sheet-Fed ICS, Version 1.0. http://www.cip4.org/document_archive/documents/ICS-MISCPS-1.0.pdf. Accessed 2006-10-31.

CIP4. 2006e. Interoperability Conformance Specifications (ICS) Registry. http://www.cip4.org/document_archive/ics.php. Accessed 2006-10-31.

CIP4. 2006d. JDF Open Source Software. http://www.cip4.org/open_source/index.html. Accessed 2006-10-25.

CIP4. 2006g. MIS to Prepress ICS, Version 1.01. http://www.cip4.org/document_archive/documents/ICS-MIS-Prepress-1.01.pdf. Accessed 2006-10-31.

CIP4. 2006f. MIS ICS 1.0 Errata revision A. http://www.cip4.org/document_archive/documents/ICS-MIS-1.0RevA.pdf. Accessed 2006-10-31.

CIP4. 2006a. Base Interoperability Conformance Specification, 1.0 Errata Revision B. http://www.cip4.org/document_archive/documents/ICS-Base-1.0RevB.pdf. Accessed 2006-10-31.

CIP4. 2006b. Integrated Digital Printing ICS, Version 1.0 Errata Revision A. http://www.cip4.org/document_archive/documents/ICS-IDP-1.0RevA.pdf. Accessed 2006-10-24.

CIP4. 2006c. CIP4 Software License, Version 2.0. http://www.cip4.org/open_source/cip4_software_license_v2.html. Accessed 2006-10-25.

Daun, Stefan, Georg Lucas, and Jürgen Schönhut. 1998. Specification of the CIP3 Print Production Format - Version 3.0. http://www.cip4.org/documents/ppf_specifications/index.html. Accessed 2006-10-31.

Enlund, Nils. 1998. Workflow Management in Graphic Arts Production. *IARIGAI* 25, 191-203.

Fielding, R., J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. 1999. Hypertext Transfer Protocol -- HTTP 1.1. http://www.w3.org/Protocols/rfc2616/rfc2616.html. Accessed 2006-10-24.

Handberg, Leif. 2003. On the implementation of production management systems in the graphic arts industry. KTH, Royal Institute of Technology.

Haugan, Gregory T. 2001. *Effective Work Breakdown Structures*. Project Management Institute.

Ifra. 2002. Ifra Special Report 6.21.3 – IfraTrack 3.0. http://www.ifra.com/WebSite/news.nsf/(StructuredSearchAll)?OpenAgent&IFRATRACK. Accessed 2006-10-05.

Kipphan, Helmut. 2001. *Handbook of Print Media*. Springer-Verlag.

Nordqvist, Stig. 1996. A Model for Global Production Management Systems in Newspaper Production, Real Time Management of Time Critical and Heterogeneous Processes. KTH, Royal Institute of Technology.

Stering, Ola. 2006. Development of a Prototype for JDF Enabled Software. http://www.stering.se/thesis/MasterThesis.pdf. Accessed 2006-10-30.

Thunell, Evelina. 2003. The CIP4 JDF Editor – Visualization of JDF. http://www.cip4.org/documents/jdf_overview/MasterProject_Evelina_Thunell.pdf. Accessed 2006-10-25.

Tuijn, Chris. 2004. Workflow Modeling in the Graphic Arts and Printing Industry. *Color Imaging IX: Processing, Hardcopy, and Applications* 5293, 66-74.

W3C. 2004. XML Schema. http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/. Accessed 2006-10-02.

W3C. 2006. Extensible Markup Language (XML) 1.0 (Forth Edition). http://www.w3.org/TR/xml/. Accessed 2006-10-02.

WfMC. 1999. Workflow Management Coalition, Terminology & Glossary, TC 1011 Issue 3.0. http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf. Accessed 2006-10-11.