

Академия медиаиндустрии



Thomas Hoffmann-Walbeck

Sebastian Riegel

Der JDF-Workflow

Lehrbuch zur Automatisierung
in der grafischen Industrie



Verlag Beruf und Schule

Томас Хоффман — Вальбек

Себастьян Ригель

JDF - Рабочий поток

Учебное пособие

Перевод с немецкого языка

2012 г.

Уважаемые читатели!

Компьютеры и информационные технологии произвели революцию в полиграфической и издательской сферах. Процессы цифровизации затронули не только технику и технологию, но и управление производством, включая его организацию, менеджмент, финансы, работу с клиентами и некоторые другие области. В этой связи для работающих или готовящихся к работе в полиграфической индустрии становится важным знание об информационно-управляющих системах, использовании соответствующих программных средств для управления производством. Книга профессора Томаса Хоффмана-Вальбека и дипломированного инженера Себастьяна Ригеля выгодно отличается от изданных ранее переводных пособий. Авторы полностью прослеживают полиграфический технологический процесс на предмет его автоматизации, начиная с приема заказа и заканчивая выпуском готовой печатной продукции. Приводимые примеры использования формата JDF конкретны и наглядны. Материал системно описывает методологию внедрения «Рабочего потока» с целью максимального удовлетворения потребностей клиента, изготовления качественной печатной продукции с наименьшими затратами, обеспечения гибкости производства, решения проблем устойчивости предприятий и фирм на рынке печатных услуг.

Проф. А.М.Цыганенко

Предисловие

Почему книга о JDF- рабочем потоке? Прежде всего, почему JDF? Это не просто формат данных, что стоит за ним, чтобы машины в типографии получали правильные задания? Кому интересно, как используются протоколы в основе телефонной связи? И можно ли даже ездить на машине, без знания технических деталей поршневых двигателей?

На деле, эти аргументы не совсем корректны. Тем не менее, мы считаем, что применение JDF должно интересовать не только специалистов в области информационных технологий, потому что:

- Job Definition Format (JDF) описывает большинство процессов в промышленности печати и представляет собой, следовательно, важную модель для использования в полиграфической индустрии;
- технология еще достаточно новая и ее применение функционирует еще не так надежно как телефон или автомобиль. При проектировании, создании, тестировании систем и возможных случаях применения JDF-рабочего потока, чтобы избежать ошибок, вполне необходимы определенные базовые знания формата.

Мы не намерены углубляться во все тонкости данного формата данных. Важным является, по сути, построение рабочего потока, который может быть реализован с помощью JDF. Поэтому, например, во многих главах описываем процессы с применением JDF, как основы для их моделирования в JDF. Мы старались объединить обе стороны. Так, Вы знаете, что только при сопоставлении можно что-либо создать.

В качестве читателей этой книги мы видим лиц, повышающих квалификацию в области полиграфической индустрии, студентов специальностей полиграфических и медиа-технологий, проходящих практику или занятых непосредственно на производстве, а также специалистов в прикладных областях применительно к медиа.

Мы не используем в данной книге глубоких знаний о формате JDF, рассчитывая также, что основные производственные процессы создания печатных продуктов читателям известны. В Глоссарии имеются некоторые пояснения, которые помогут читателям при необходимости понять глубже материал книги.

Конечно, в книге не все этапы процессов создания печатного продукта рассмотрены до микроскопических деталей. Для этого есть специальные труды по обработке изображений, компьютер - печатная форма (СtP), офсетной печати и другие. Мы хотели лишь дать обзор возможностей применения формата на основе технических примеров. С другой стороны, показать, что можно сделать и не философствовать в общем смысле о применении формата.

И еще одно предупреждение: мы хотели осветить основополагающие принципы описания рабочих процессов, а не давать советы и рекомендации для конкретного рабочего потока по производству печатного продукта. Мы не описали *Workflowsysteme* различных производителей – для этого имеется специализированная литература.

Мы учим на протяжении нескольких лет студентов университета допечатным процессам в полиграфии. Средств информации в г. Штуттгарте. Занятия проходят как в виде лекций, так и практических занятий с выполнением определенных заданий. Составляя списки рекомендуемой литературы, мы обнаружили, что на эту тему имеется недостаточно источников. При этом следует добавить, что с одной стороны, имеется более чем 1000-страничная спецификация по JDF- формату [13], но она, в первую очередь, адресуется специалистам и является довольно сложной. С другой стороны, в списке литературы нами приводятся источники, содержащие общие описания экономических преимуществ JDF- автоматизации, например, в [32], [33]. Конечно, имеются обширные руководства, написанные разработчиками JDF-рабочего потока по использованию их систем (см., например, [16]) и, как правило, по применению подсистем на отдельных участках производства (см., например, [10], [22]).

Мы твердо уверены, что JDF-рабочий поток вызывает новые формы производства в графической индустрии и способствует ее развитию. И поэтому мы считаем, что заниматься этой темой, безусловно, стоит. Об этом говорилось еще в *Klimsch Jahrbuch* (Klimsch - ежегодник) за 1924/25г.г. (стр. 109):

„Неумолимо и неотвратимо экономическое развитие идет своим путем. Старое разрушается, и новая жизнь возникает на его руинах. А медленное сменяется почти взрывным развитием, которое в последующие десятилетия ждет графическую технику“.

Во введении (глава 1) мы рассматриваем общие свойства формата JDF и историю его возникновения, а также применение при создании рабочих потоков. Те, кто уже в течение определенного времени знаком с этой темой по статьям о JDF, может этот материал уверенно опустить.

Во второй главе описываются три сценария построения производственного процесса в типографиях. Все три предприятия находятся на современном уровне развития техники, однако только два используют JDF-технологии. Мы хотели показать читателям их различия и особенности. В параграфе 4 этой главы раскрываются такие термины, как рабочий поток, его составляющие и др., а в параграфе 5 представлены общие характеристики бизнес-процессов на основе рабочего потока.

Модель процесс - ресурс и, соответственно, модель производитель-потребитель являются темой главы 3. В ней описываются основы и преимущества данных моделей с точки зрения применения в области управление заказами, допечатного, печатного и отделочного производств.

В 4 главе рассмотрены классические метаданные и их применение.

В главе 5 дано краткое введение в XML (*Extensible Markup Language*) в JDF-коде.

В следующей шестой главе рассмотрены основные JDF-структуры. Глава 7 посвящена работе с сообщениями в Job Messaging Format (JMF), иными словами „SMS полиграфической промышленности“. Это формат данных и протокол, используемый для коммуникации в JDF- среде. Они оба лежат в основе глав с 8 до 12, в которых речь идет о деталях рабочего потока и JDF- аналогах при управлении заказами, допечатными процессами, печатью и послепечатным производством, а также при производстве упаковки.

В главе 13, обсуждаются два проекта, которые могут заинтересовать читателей.

В первой части речь идет о создании рабочего потока с использованием модулей одного или нескольких производителей. Вторая часть дает краткое введение в JAVA - программирование JDF - приложений.

В конце некоторых глав, мы приводим перечень заданий, которые помогут углубленно освоить материал.

В JDF терминологии используется английский язык. При необходимости в книге делается перевод некоторых терминов на немецкий, если это имеет смысл. В этом случае термины на английском языке выделяются курсивом в скобках.

Авторы благодарят за помощь и поддержку при создании данной книги:

г-на Дитера Адама (MB Bauerle), г-на Яна Брайтхолда (HELL Gravure Systems),

г-на Рубена Кагние (EskoArtwork), г-жу Анну Даннхорн (Fujifilm), г-на Готфрида

Грасла (Heidelberger Druckmaschinen), г-на Стефана Копес (тип-я Mack), г-жу Ульрику

Курц (MBO), г-на Бернда Лаубенгайера (типография Laubengaier), г-жу проф., д-ра

Кристу Нес (Высшая школа медиа), г-на Ливена Плеттника (EskoArtwork), г-жу Ульрику

Зеетхалер (Heidelberg Druckmaschinen), г-на Матиаса Зигеля (MB Bauerie), г-на Клауса Штоклаза (MBO).

Авторы примут все предложения и замечания, направленные на совершенствование книги.

Томас Хоффманн-Вальбек, Себастьян Ригель

Оглавление

Предисловие.	4
Введение.	
Глава 1. Введение в тему.	10
1.1. Развитие формата JDF	10
1.2. Основы формата JDF	13
1.3. Реализация JDF - рабочего потока	15
Глава 2. Основы формирования рабочих потоков.	18
2.1. Пример фирмы, которая не использует JDF –рабочий поток	18
2.2. Пример фирмы, которая частично использует формат JDF	20
2.3. Фирма, которая полностью использует JDF/JMF-сеть	21
2.4. Определения	23
2.5. Классификация рабочих потоков	25
2.6. Взаимодействие системы управления рабочим потоком и форматом рабочей карточки (Jobtiken)	30
Глава 3. Модели рабочего потока печатного производства.	35
3.1. Управление заказом	38
3.2. Рабочий поток в допечатной подготовке	40
3.3. Процесс листовой офсетной печати	47
3.4. Пример модели послепечатной обработки	50
Глава 4. Классические метаданные и их применение.	54
4.1. Метаданные для фотографий и документов	55
4.2. Формат для печати PPF	61
4.3. Формат PJTF	68
Глава 5. Краткое введение в XML.	74
5.1. Структура XML - документа	74
5.2. Именное пространство XML	76
5.3. Resource Description Framework (RDF) и XMP	79
5.4. сXML-документ	80
Глава 6. Введение в JDF.	83
6.1. Структура JDF - документа	83
6.2. Примеры для JDF - узлов	88
6.3. Разделяемые ресурсы	94
6.4. «Черные ящики» и комбинированные процессы	96
6.5. JDF - рабочий поток — архитектура	99
6.6. Разделение и объединение	104
6.7. ICS	108
Глава 7. Формат рабочих сообщений (JMF).	113
7.1. Модели коммуникаций	113
7.2. JMF - семейства	116
7.3. JMF - ICS (интегральная коммуникационная система)	124
Глава 8. Системы управления заказами.	127
8.1. Основные функции АМС	128
8.2. Использование JDF для процессов резки	130
8.3. Документы MIS ICS	145
8.4. Print Talk и JDF –интерфейс клиентов	148
Глава 9. Стадия допечатной подготовки.	154
9.1. Связи между MIS и Prepress	156
9.2. Монтаж	163

9.3. Треппинг	168
9.4. RIPing и изготовление печатной формы	172
9.5. Proof и разрешение на печать	176
Глава 10. Печать.	181
10.1. Традиционные технологии печати	182
10.2. Цифровая печать	198
Глава 11. Отделочные процессы.	206
11.1. Одноножевая бумагорезальная машина	208
11.2. Фальцевальная машина	212
11.3. Вкладочно-швейно-резальный агрегат	214
Глава 12. Печать упаковки.	221
12.1. Дизайн формы для штанцевания, монтаж и изготовление формы для штанцевания	225
12.2. Штанцевание и придание формы складным коробкам	229
12.3. Штриховой код	234
Глава 13. Проекты по внедрению JDF/JMF.	237
13.1. Создание рабочего потока с использованием модулей одного производителя	239
13.2. Создание рабочего потока с использованием модулей нескольких производителей	242
13.3. Программирование JDF/JMF	244
Некоторые термины и определения (гlossарий)	250
Список литературы	254
Авторы	260

Глава 1. Введение в тему.

Введение

Под термином "JDF-Workflow" понимается общее взаимодействие технологических процессов в полиграфической индустрии, которые базируются на форматах JDF (*Job Definition Format*) и JMF (*Job Messaging Format*). Основная цель применения таких форматов - это автоматизация процессов путем внедрения различных приложений и систем.

Основная идея очень проста — это сбор актуальных данных о состоянии процессов и их передача на различные участки производства в целях управления. Например, параметры печатного листа задаются только один раз для изготовления печатной формы, но эти данные остаются актуальными для всех участков процесса, таких как: калькуляция заказа, формирование заказа на печать, резка и др.

"JDF-Workflow" применим и в других целях, среди которых - сокращение ошибок в процессе производства и времени выполнения заказа, прозрачность всех участков в процессе выполнения заказа.

1.1 Развитие формата JDF

Проиллюстрируем развитие формата временными этапами (обозначим, не нарушая нумерации последовательности схем, таблиц и рисунков, как рис.1.1)

- 1993 - Развитие концепции формата PPF;
- 1995 - Презентация формата PPF Version 1.0 и основание организации CIP3;
- 1996 - PPF Version 2.0;
- 1998 - PPF Version 3.0;
- 2000 - Презентация формата JDF и основание организации CIP4;
- 2001 - JDF Version 1.0;
- 2002 - JDF Version 1.1;
- 2004 - JDF Version 1.3;
- 2008 - JDF Version 1.4.

Незадолго до проведения выставки DRUPA 2000, формат JDF был предложен инициаторами - консорциумом фирм: Heidelberg Druckmaschinen, manroland, Agfa и Adobe, а непосредственно на ней состоялась его презентация. В сентябре того же года объединением CIP4 было произведено дальнейшее усовершенствование формата JDF. На сегодняшний момент организация CIP4 - это союз, насчитывающий более 300 членов, среди которых поставщики, пользователи, консультанты и исследовательские институты полиграфической отрасли.

Уже упоминалось, что организация CIP4 возникла из созданной в 1995 году организации CIP3. CIP3 - это аббревиатура, которая расшифровывается как „*International Cooperation for Integration of Prepress, Press and Postpress*“, что в переводе на русский означает: "Международное сообщество для интеграции процессов допечатной подготовки, печати и послепечатной обработки". Организация CIP3 пропагандировала формат PPF (*Print Production Format*), который был принят за основу для развития формата JDF организацией CIP4. Представительство организации CIP4 находится в городе Цюрих в Швейцарии.

Как формат PPF, так и форматы JDF/JMF- это форматы интерфейсов для объединения в сеть всех участков производства. Оба этих формата служат автоматизации промышленного полиграфического производства, они отменяют ручной принцип управления им. Можно сказать, что лозунг этого нового принципа - *Computer Integrated Manufacturing (CIM)*, что в переводе звучит как "Компьютерные технологии, интегрированные в производство".

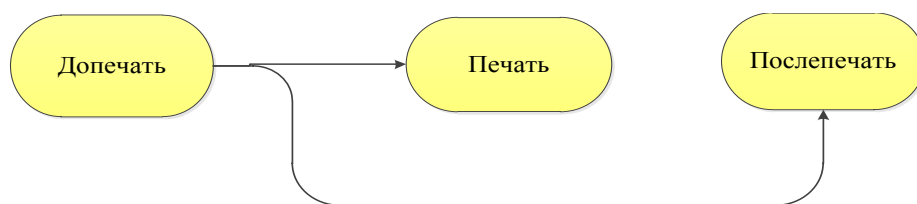


Рис. 1.2 Поток PPF-информации

Формат PPF в полиграфической индустрии, известный так же как CIP3-Format, осуществляет передачу технических данных из отдела допечатной подготовки в печатный цех, либо из отдела допечатной подготовки в отдел послепечатной подготовки. Другая его важная функция - это возможность предварительного просмотра будущего оттиска (*Preview*) в формате PPF и параллельное формирование задания RIP на изготовление

печатной формы. Предварительное изображение оттиска также содержит в себе информацию об установках цвета, которая затем передается печатной машине.

Ещё один пример функционирования PPF - Workflow, это передача размеров печатного листа, параметров резки и фальцовки из отдела допечатной подготовки в отдел послепечатной обработки. При вёрстке печатного листа устанавливаются соответствующие метки и параметры для послепечатной обработки и когда вёрстка листа полностью закончена, происходит формирование файла данных в формате PPF. Такой PPF - файл передаётся на резальную или фальцевальную машину, после чего происходит установка соответствующей программы резки или фальцовки, и машина автоматически может выполнять введенные задания.

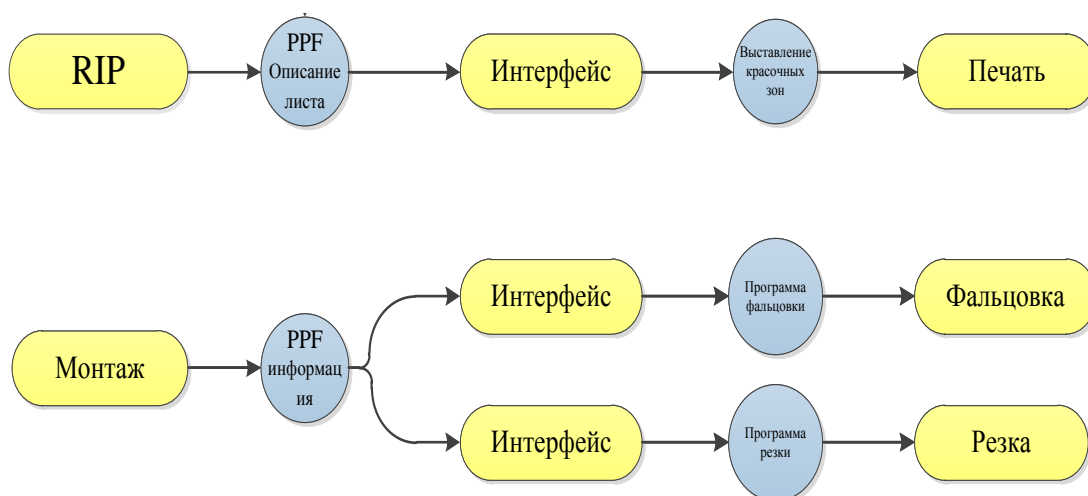


Рис. 1.3 Детализация PPF - рабочего потока

Обобщая изложенное можно сказать, что организация производства с применением PPF - Workflow имеет следующие преимущества:

- технические данные, содержащие в себе параметры для работы оборудования, могут передаваться из одного подразделения в другое;
- подборка данных в одном и том же формате может передаваться на различные устройства (например, как на устройства печати, так и на устройства послепечатной обработки);
- отпадает потребность в выполнении некоторых операций (например, в выполнении операции ручной цветокоррекции);
- благодаря стандартизации печатного производства и универсальности формата *Print Production Formats*, в производстве могут быть использованы модули разных производителей.

Более детальную информацию о данном формате Вы можете найти в параграфе 4.2.

1.2 Основы формата JDF

Иногда формат JDF (*Job Definition Format*) определяется как "Электронный пакет заданий", но на самом деле он несет в себе гораздо больше функций. В действительности, этот формат является не только "пакетом заданий", он также может формировать рабочие потоки, передавать настройки работы оборудования и вести протокол полного производственного процесса.

JDF осуществляет поддержку:

- при передаче данных о заказе;
- при передаче данных о параметрах настройки оборудования;
- при учете производства и учете работы оборудования;
- при диспозиции;
- при контроле выполнения заказа.

Формат JDF обладает не только всей функциональностью формата PPF, но и выходит далеко за его рамки.

При JDF - рабочем потоке осуществляется:

- информационная поддержка при взаимодействии программ калькуляции заказа (MIS - информационно-управляющая система) с производством. Информационные файлы о заказе в формате JDF могут содержать в себе такие "тонкости" как, например, информацию о растре. Одним словом, такой файл содержит в себе полностью всю ту информацию, которая требуется абсолютно на всех участках производства:

- сбор данных о работе оборудования и иных производственных сведений ("*Job Tracking*"), контроль выполнения заказа, отчетная калькуляция;
- учет производительности труда и ведение протокола;
- информационная поддержка в режиме "он-лайн" при общении заказчика и исполнителя (*eCommerce*);
- наличие разделов "Ассортимент", "Предложения", "Подтверждение заказа" и др.;
- формирование требований и установок к интеграции программного обеспечения (*Plug-and-play*) на основе форматов JDF/JMF на соответствующем участке производства;
- протоколирование абсолютно всех производственных процессов.

Рис. 1.4 Пример записей в портфеле заказов, который при поступлении заявки от клиента пополняется новой информацией

Изменения JDF - документа системы, позволяющие выделять необходимую информацию из всего пакета данных и передавать её на необходимый участок производства в виде отдельного файла, а затем снова собрать в общий трансформированный файл, представлены на рис. 1.5.

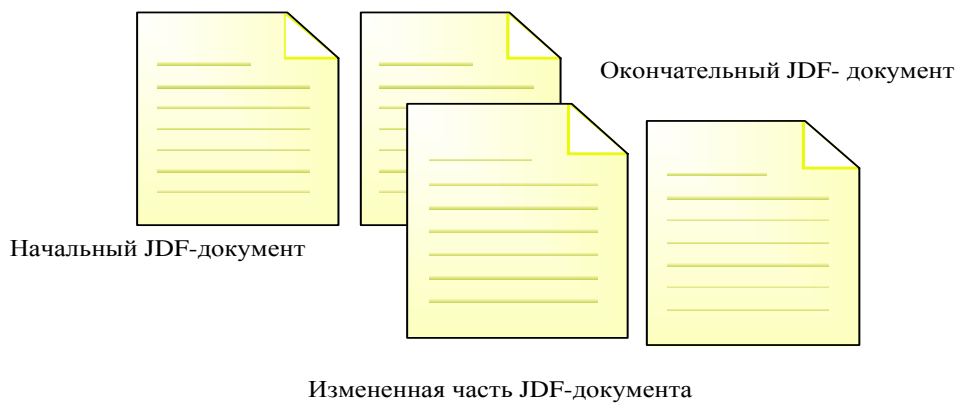


Рис.1.5 Изменение JDF-документа при поступлении заданий

Из сказанного следует, что JDF - рабочий поток в сравнении с PPF - рабочий поток приводит к большей оптимизации производственного процесса (рис. 1.6).

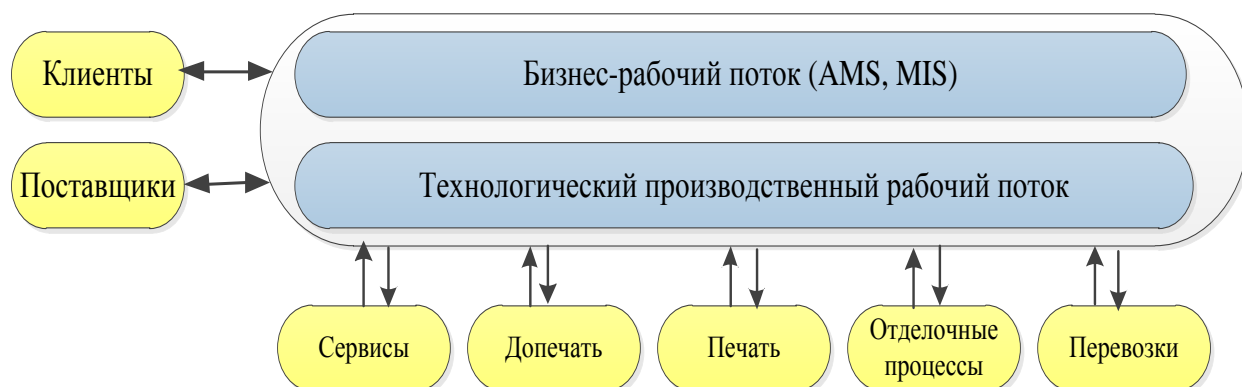


Рис. 1.6 Коммуникационные пути при JDF - рабочем потоке

Эти и другие возможности будут подробнее обсуждаться в следующих главах этой книги.

Так всё же, почему аспект оптимизации производства становится всё более важным? Ответ на этот вопрос очень прост. На сегодняшний день наиболее влияющими на производство являются такие факторы, как уменьшение тиражей и необходимость повышения производительности труда, а также сокращение времени для подготовки комплекса технологического оборудования к работе. Все эти факторы приводят к необходимости оптимизации производства и повышению рентабельности предприятия.

1.3 Реализация JDF - рабочий потока

JDF - это только формат данных, но никак не рабочий поток. Спецификация формата JDF не содержит никаких указаний для разработчика о том, какие модули должны быть включены в систему. Это характерно и для PDF - спецификации. Несмотря на это, все же имеются несколько общедоступных моментов, которые в целом актуальны для формата JDF - рабочего потока.

Данные в формате JDF не передаются свободно от одного программного обеспечения к другому или от одной машины к другой. Теоретически это было бы возможно, но на практике стало бы "страшным сном" как для разработчиков, так и пользователей. Вместо этого существует центр хранения данных - банк данных, где собраны данные в формате JDF выполнения производственной работы. Такой центр может быть присоединен к

системе MIS (*Management-Information-System*) или другой системе управления производством.

Использование форматов JDF/JMF пока не может быть осуществлено во всех процессах полиграфического производства, или, по крайней мере, в ближайшее время. Некоторое оборудование, чаще всего машины для послепечатной обработки, вообще не могут быть подключены к общей сети на базе JDF/JMF, но данные с таких машин всё равно должны быть доступны, при этом не важно, происходит ли это с использованием формата PPF или иными способами. Классические системы рабочих потоков для устройств компьютер – печатная форма (CtP или *Computer-to-Plate*) были во многом "эмбрионами" современных систем JDF - рабочий поток, потому что такие системы имели с одной стороны связь с MIS, а с другой стороны связь с печатными отделениями. Разумеется, здесь также существуют ограничения, и информация о некоторых процессах проходит мимо JDF/JMF. Например, в подразделении рекламного агентства, где происходит допечатная подготовка, в большинстве случаев напрасно искать рабочие процессы, базирующиеся на формате JDF.

Спецификация формата JDF определяет возможное содержание тех файлов, которыми могут обмениваться различные участки производственного процесса. Однако эта спецификация ни в коем случае не говорит о том, какие участки производственного процесса предоставляют необходимую информацию, и кто должен её оценивать. Для рабочего потока на базе формата JDF существуют отдельные описания и пояснения, которые касаются коммуникации и взаимодействия различных модулей. Так, содержатся описания коммуникаций между допечатной ступенью и системой MIS или системы MIS и печатной машины. Организация CIP4 располагает для этого полным комплектом документов, которые соответствуют спецификациям ICS (*Interoperability Conformance Specifications*). Познее рассмотрим этот вопрос в параграфе 6.7.

Первая публикация спецификации формата JDF (*JDF Spezifikation 1.0*) появилась в январе 2001 года. В это же время на рынок вышли первые приложения, совместимые с форматом JDF. Между тем, с 2008 года существует спецификация *JDF Spezifikation 1.4*, а также очень много приложений, которые совместимы с форматом JDF. Организацией CIP4 регулярно публикуется каталог („*The JDF Marketplace*“) [14] о различных приложениях JDF и применениях. Например, в издании каталога за 2008 год значились несколько сотен продуктов и руководств по их использованию.

Кроме того, программы на основе формата JDF сертифицированы, то есть они соответствуют ICS - требованиям. Список сертифицированных продуктов можно найти на сайте организации CIP4. Тем не менее, существование ICS списков не дает оснований

для вывода, что можно без разбора использовать программные модули различных производителей, слепо и без тестирования. Всегда имеются ловушки. ICS – стандартизация всегда ориентируется на типовые решения, что не всегда подходит для специальной печатной продукции. И, поэтому, при создании или расширении JDF интеграции в типографии всегда есть много вопросов, требующих уточнений и проверки.

Советы для практической реализации проектов можно найти в [35], более общие рекомендации приводятся нами в главе 13.

Задания для самостоятельной работы:

- найдите официальный сайт организации CIP 4(www.cip4.org). Прочитайте при этом внимательно введение спецификации [1.3].

Глава 2. Основы формирования рабочих потоков (Workflows).

В этой главе рассмотрим две совершенно противоположные друг другу позиции. В первых темах главы нам будут представлены методы работы трёх типографий, которые мы назовём X, Y и Z. Фирма X установила систему рабочего потока (Workflow), которая полностью современна, но не содержит приложений формата JDF. Фирма Y, напротив, реализовала несколько приложений, использующих формат JDF. Фирма Z полностью использует систему на основе JDF/JMF.

После этих конкретных примеров мы рассмотрим формат JDF сначала абстрактно. В параграфе 2.4, в частности, ознакомимся с такими основными понятиями, как рабочий поток (Workflow), системы управления рабочим потоком (Workflow-Managementsystem) и рабочая карточка (Jobticket). В параграфе 2.5 рассмотрим несколько общих свойств рабочего потока, а также представим классификацию рабочих потоков, которая будет привязана к типографиям.

2.1 Пример фирмы, которая не использует JDF - рабочий поток

Фирма X является типографией полного цикла. Её штат насчитывает 12 сотрудников. В отделе допечатной подготовки уже несколько лет назад была установлена современная система управления рабочим потоком (Workflow-Managementsystem). Типография располагает фотонабором и устройством фотовывода. Действующая на типографии система управления рабочим потоком способна формировать данные в формате PPF о цветовых настройках и приводке красок, но до сегодняшнего времени эта функция ещё не использовалась. Парк печатного оборудования насчитывает три машины офсетной печати (пятикрасочная, четырехкрасочная и двукрасочная). Кроме того, имеется одна цифровая печатная машина со встроенной секцией послепечатной обработки. В отделении послепечатной обработки есть резальная и фальцевальная машины, а также стоп-цилиндрическая машина для тиснения фольгой и высечки. Средний недельный оборот типографии составляет около 50 заказов, многие из которых сложносоставные и имеют много различных особенностей.

Запросы на заказ поступают в типографию чаще всего по телефону или электронной почте, реже факсом или с курьером. Сотрудник, ответственный за выполнение заказов, вводит важнейшие их параметры в систему управления. Расчеты и калькуляцию заказа

чаще всего выполняет сам владелец типографии. При калькуляции он использует только одну ценовую таблицу и карманный калькулятор. Примерно в одной трети всех случаев в ценовой таблице отсутствуют параметры, необходимые для калькуляции, такие вопросы каждый раз решаются индивидуально с заказчиком.

Разнообразие вариантов заказываемых услуг, которых требует потенциальный клиент, сильно возросло в течение последних лет. Прежде всего, рекламные агентства заказывают легко модифицируемые варианты исполнения.

Разница между запрашиваемым и предлагаемым ассортиментом решений составляет 10 %. К заказу, приходящему по электронной почте, чаще всего прикреплены файлы на печать. 80 % всех заказов присылаются в формате PDF, остальные 20 % приходят в форматах InDesign, QuarkXPress, OfficeSuite. Сотрудник, ответственный за выполнение поступившей заявки, делает заказ на материалы в соответствии с ее объёмом, согласовывает производственное задание и формирует программу загрузки печатной машины, а также производит распечатку бланка из системы управления заказами. При стоимости работы на сумму более 500 Евро, клиент получает письменное подтверждение о приеме заказа.

Используя программу Preflight – Programm (*предпечатная*), работники отдела допечатной подготовки проверяют полученные файлы. Как правило, две трети полученных файлов содержат ошибки и должны корректироваться. Все ошибки, которые могут быть устранены в течение короткого периода времени, исправляются без согласования с клиентом и это не сказывается на общей стоимости заказа. После вёрстки выводится предварительный макет, который по почте или с курьером посылают на подпись клиенту. Когда макет подписан, в типографии приступают к изготовлению печатной формы. Только в тех случаях, когда заказ очень срочный или клиент является надежным, макет в формате PDF посылают клиенту по электронной почте. После того, как клиент утверждает направленные ему материалы, происходит изготовление печатных форм. Готовые печатные формы передаются в печатный цех. План работы над конкретным заказом заносится в общий план-график, где фиксируются конечные сроки его выполнения.

В процессе выполнения заказа у клиента бывает необходимость внесения изменений. При поступлении изменений, ответственный исполнитель заносит их в бланк, описывающий заказ. Формирование счетов-фактур и другой бухгалтерской документации осуществляется с помощью вышеупомянутой системы управления заказами. Типография имеет интерактивный интерфейс с налоговым органом.

Отчетная документация готовится в этой типографии нерегулярно и выполняется от руки. Рабочие бюллетени заполняются лишь периодически и только для того, чтобы время от времени проверять предлагаемую клиенту ценовую таблицу. Выполнение заказа для цифровой печати аналогично выполнению заказа офсетной печати.

С одной стороны, владелец думает, что в его фирме используется автоматизированная система управления производством, но с другой стороны производственный процесс не имеет достаточного уровня автоматизации. В действительности в Германии такая ситуация является типичной для малых типографий.

2.2 Пример фирмы, которая частично использует формат JDF

Фирма Y печатает, в основном, каталоги, журналы и журнальные приложения. Для производства каталогов, журналов и журнальных приложений используются две машины рулонной печати, а на листовой офсетной машине печатают конверты для этой же продукции. Фирма получает заказы либо от постоянных клиентов, либо от представительской службы, которая организует привлечение новых. Для заказов, поступающих от представительской службы, есть специальный формуляр в бумажной форме, где указываются сведения об объеме издания, тираже, бумаге, цветности и другая информация. Данные, как от постоянных клиентов, так и от клиентов представительской службы, вносятся в базу системы управления заказами. Калькуляция производится также с помощью этой системы. Представительская фирма отправляет потенциальным клиентам предложения по услугам типографии, сведения об ассортименте, отвечает на возникающие вопросы.

Если заказ подтвержден клиентом, то из системы управления заказами делается распечатка об объеме печатной работы, сроках исполнения и др. Этот документ отправляют в службу планирования и управления производством. Сотрудники этой службы составляют план-график работ, вносят соответствующую информацию в документы по загрузке всех подразделений типографии. Отдел планирования и управления производством не имеет обратной связи с системой управления заказами, поэтому не существует автоматического получения данных от подразделений системой управления заказами. По этой причине система управления заказами не получает, например, данных о превышении сроков их выполнения. Поставки бумаги и других материалов оформляются после согласования требуемых объемов и запрашиваются по электронной почте у поставщиков. Через систему управления заказом клиент получает подтверждение о принятии его к исполнению.

Макет печатного листа (разметка, метки фальцовки, нумерация страниц) формируется в системе управления заказом. В него вносится также дополнительная информация, которая необходима для последующего исполнения заказа (например, сведения о растре и пр.).

Система управления заказом формирует электронную версию программы исполнения заказа на каждый вид операции, причем в формате HTML - документа, к которому имеют доступ все участки производства. Специально для отдела допечатной подготовки дополнительно в бумажной форме распечатывается документ, где перечисляются все операции, которые должны выполняться. После ознакомления с документом, сотрудники подписывают его. Посредством формата JDF информация передается в систему рабочего потока допечатного отдела (Prepress-Workflowsystem). Как видим, отделы получают большинство данных о заказе (например, данные о стоимости и др.) из автоматизированной системы. Печатные машины получают задание на печать (главным образом, установки на приводку красок) в формате PPF.

Посредством Интернет - портала налажена обратная связь с клиентами. Они могут вносить в заказ изменения и коррективы, которые через систему рабочего потока допечатной ступени поступают сначала в отдел допечатной подготовки, а затем передаются далее на соответствующий участок производства.

Сбор производственных данных проходит с помощью специальных терминалов, в которые сотрудники вводят необходимую информацию для единой автоматизированной системы технологического процесса. Однако у этой сети нет обратной связи с системой управления заказами, поэтому ввод данных происходит вручную. Только после всех перечисленных операций система управления заказами производит уточнение калькуляций и выявляет величины издержек.

На этом примере становятся понятными два момента. С одной стороны, осуществляется оптимизация производства посредством работы системы управления заказами и появления интерфейса на основе JDF. Но с другой стороны, этот пример показывает типичный недостаток: множество интерфейсов и отсутствие общего интерактивного пространства с обратной связью между подразделениями, системой управления заказами и отделом планирования производства.

2.3 Фирма, которая полностью использует JDF/JMF - сеть

Штат фирмы полного технологического цикла Z насчитывает 150 сотрудников. Предприятие Z - это типичная универсальная типография, которая выпускает не только книги, брошюры и журналы, но и печатает визитные карточки и другую

представительскую продукцию. Заказы поступают по факсу и электронной почте. Калькуляция производится в системе менеджмента MIS. Система имеет интерактивную связь с поставщиками расходных материалов, что позволяет автоматически обновлять ценовую таблицу и получать актуальную информацию о стоимости бумаги, пластин, краски и др. Заказ бумаги осуществляется только после проверки собственных бумажных запасов.

После получения заказа система автоматически формирует его бланк в виде HTML-документа в портфеле заданий и выкладывает его в открытую сеть, к которой имеют доступ участники производства.

Для каждого заказа система MIS генерирует документ в формате JDF, который поступает на сервер рабочего потока, а затем в виде заданий и в производство. На этом сервере всегда содержится актуальная информация о состоянии выполнения того или иного заказа. На основании оформленного JDF - документа производится автоматическое распределение заданий по участкам, в том числе на стадии работ по допечатной подготовке.

Клиенты загружают файлы данных для печати на сервер типографии. Система управления получает данные о поступлении новых файлов, проверяет и обрабатывает их. Результаты обработки протоколируются и записываются в JDF-документе, а также отправляются клиенту.

Через сеть JDF/JMF четыре из шести печатных машин получают необходимую информацию о формате печатного листа, цветности и об установках на приводку красок. Посредством обратной связи информация от печатных машин о состоянии выполнения работ поступает на сервер рабочего потока. Две оставшиеся печатные машины данного производства - это устаревшие модели более ранних годов выпуска, поэтому они не могут быть подключены к общей сети. Таким образом, функционирует двусторонняя связь между всеми участками производства и системой управления заказами.

Оборудование отдела послепечатной обработки, как дальнейшей логистики в настоящее время не подключено к общей сети, поэтому выдача заданий и информации о состоянии работ ведётся через специальные терминалы. После завершения всех работ по выполнению заказа, система управления производит распечатку документов на выпущенную продукцию. В этой типографии, как мы видим, связь системы с отделом логистики не реализована, так как логистические операции осуществляют подрядные фирмы. Зачастую это маленькие структуры, которые не имеют необходимого для этого обеспечения.

На примере данной фирмы становится понятно, что, несмотря на высокое оснащение в целом производства необходимым обеспечением и наличием общей сети, многие его участки не имеют доступа к системе на основе форматов JDF/JMF. К таким участкам можно отнести, в частности, отдел послепечатной обработки, для которого на рынке существуют специальные решения по рабочему потоку. Но это уже вопрос инвестиций и времени.

2.4 Определения

Workflow - это "протекание производственного процесса" или "рабочий поток", а *Workflow-Managementsystem* - это система управления рабочим потоком. Для того чтобы принцип работы такой системы был понятен, можно рассмотреть следующий пример. На гравюре 1555 года показано, что механизмы приводились в движение с помощью воды (рис. 2.1). На сегодняшний день вода, которая приводила в движение механизмы, в современных системах управления условно заменена информацией на рабочей карточке или рабочим заданием (*Jobticket*). В принципе, в этом и состоит суть рассматриваемых вопросов.

Рассмотрим ещё несколько понятий. Начнем с термина *компьютерная поддержка совместной работы (Computer Supported Cooperative Work, CSCW)*. При этом речь идет о междисциплинарной области исследований, связанной с совместной работой устройств и, в частности, в коммуникационных технологиях. Различные точки зрения на CSCW приводятся, например, в [41]. *Продукты для групповой, совместной работы (Groupware)* - это CSCW-приложения: программные и технические средства. Их много и они различного типа. Например, e - Mail - система используется в качестве важного инструмента не только обмена сообщениями, но также и для проведения конференций, передачи информации и т.д.

Под термином "*рабочий поток (Workflow)*" мы понимаем совокупность всех действий по управлению производственными операциями, которые были заданы, выполнены, а также проконтролированы системой. Рабочий поток, сам по себе, не обязательно должен быть оснащён компьютерным управлением. Например, процесс приготовления пищи - это, своего рода, тоже рабочий поток. То же самое можно отнести и к печатной машине, где рабочим потоком является последовательность действий: подготовительные работы, установка печатных форм, приладка и т.д.

Управление рабочим потоком (Workflow-Management) - это совокупность нескольких действий. Сюда относят: моделирование потока, управление потоком и контроль над ним.

Примером документа управления рабочим потоком можно назвать большой план-график, по которому выполняются работы в печатном цехе.

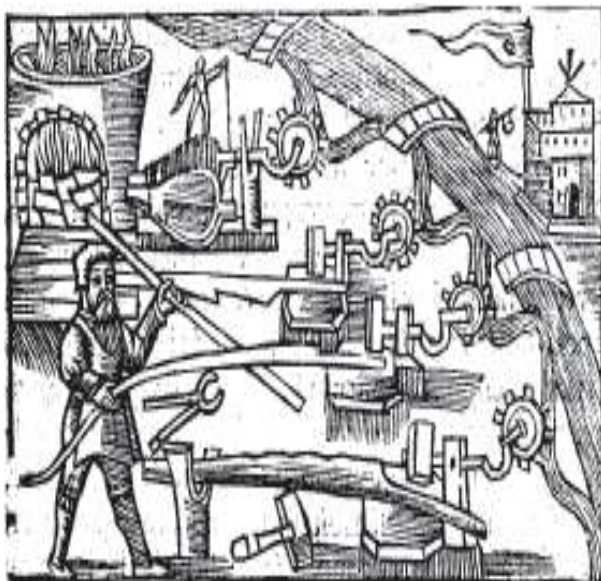


Рис. 2.1 Олаус Магнус. Историческая аналогия системы рабочего потока

И, наконец, *система управления рабочим потоком (Workflow-Managementsystem, WMS)* - это аппаратно-программное обеспечение коллективного пользования, разработанное для управления рабочим потоком. С помощью программного обеспечения задают и определяют последовательность выполнения операций и осуществляют контроль. Материалы по этой теме можно найти в [31],[43]. Консорциум *Workflow Management Coalition (WMC)* на своём сайте в Интернете приводит следующие определения:

- *рабочий поток (Workflow)* - это полная или частичная автоматизация производственного процесса, при котором документы, задания и другая информация одним и тем же установленным методом передаётся в виде команд, заданий, отчетов и др. от одного участка производства к другому;

- *система управления рабочим потоком (Workflow-Managementsystem)* - это система, которая с помощью программного обеспечения формирует рабочий поток (*Workflow*) и управляет им, причем все модули программного обеспечения этой системы функционируют в единой сети. Такое определение не совсем корректно. Лучше было бы

сказать, что *Workflow-Managementsystem* - это система управления рабочим потоком, которая базируется на формате JDF (JDF-Workflow).

Рабочая карточка (Jobtickets) имеет значение некоторой электронной информационной единицы, программы для поддержки систем рабочего потока в полиграфической промышленности. Jobticket может быть, например, инструкция или параметр для контроля печати при управлении ею. Естественно следует, что Jobtickets может описываться Job Definition Format. Другими словами: JDF- основа для Jobticket-формата.

WMS-состоит из нескольких программ-двигателей (*движков*) рабочего процесса (*Workflow-Engins*), называемых Jobticket-процессоры (*Jobticket-Prozessoren*). Это программные модули (как *тренинг-двигатель (Trapping-Engine)*, *модуль цветокоррекции (Color Transformation Modul)* или программа для пульта управления печатной машины), которые Jobtickets создает, интерпретирует и должна выполнять.

Jobtickets представляет в допечатной ступени форму „метаданных“ (*Metadaten*) или по-иному данные, которые содержат информацию о других данных. Метаданные противостоят и в то же время находятся во взаимосвязи с данными содержания (*Content-Daten*). Последние данные могут быть как открытые приложения в программах InDesign, Quark и т.д., или в закрытых форматах обмена, таких как PDF. Метаданные разделяются на „описания объектов“ и „инструкции“. Описание, например, содержит утверждение, что разрешение изображения от 400 пикселей на дюйм. Однако может быть и указание, что изображение должно быть пересчитано на 300 ppi.

Метаданные могут быть вместе с *Content*-данными в файле (см. XMP в параграфе 4.1) или отдельно, как в формате JDF. Когда метаданные находятся за пределами данных контента, тогда делается ссылка, это будет, например, указание на имя файла.

2.5 Классификация рабочих потоков (Workflows)

Итак, мы дали определение термину *система управления рабочим потоком (Workflow-Menagementsystem)*. Теперь необходимо рассмотреть классификацию таких систем непосредственно для полиграфической индустрии.

Выделим три разновидности систем управления (*Систем управления рабочим потоком* - СУРП):

- СУРП (*WMS*) – подразделения;
- СУРП (*WMS*) в целом предприятия;

- СУРП (*WMS*) на основе интерактивной связи ("Online-Portale") с поставщиками и/или заказчиками.

СУРП (*WMS*) между подразделениями внутри предприятия в свою очередь также может быть:

- СУРП (*WMS*) со связью с общей системой управления производством (*MIS*);
- СУРП (*WMS*) без связи с общей системой управления производством (*MIS*).

Эта классификация в действительности позволяет говорить о Workflow- интеграции.

В реальности данные системы управления (СУРП) не всегда удается классифицировать настолько четко. Понятно, что описанные категории в реальности не всегда так ясно выделяются и отличаются друг от друга. Соответственно конкретная СУРП (*WMS*) связывает, в основном, только часть всех звеньев, что называется, находящихся рядом и работающих в ее зоне и которые отстоят недалеко. Часто имеются подсистемы некоторых подразделений, которые не интегрированы в СУРП (*WMS*), и не от нее осуществляется поддержка их рабочих операций. В реальности есть в технологическом процессе предприятий и целые отделы, которые не интегрированы в систему. Таким образом, встает вопрос о диапазоне охвата СУРП (*WMS*) производственного процесса. Так RIP, который определяет, в том числе, задания для расчета красочных зон в печатной машине, может стать подсистемой *WMS*, как ключевое звено всей системы, интегрированной на основе JDF!

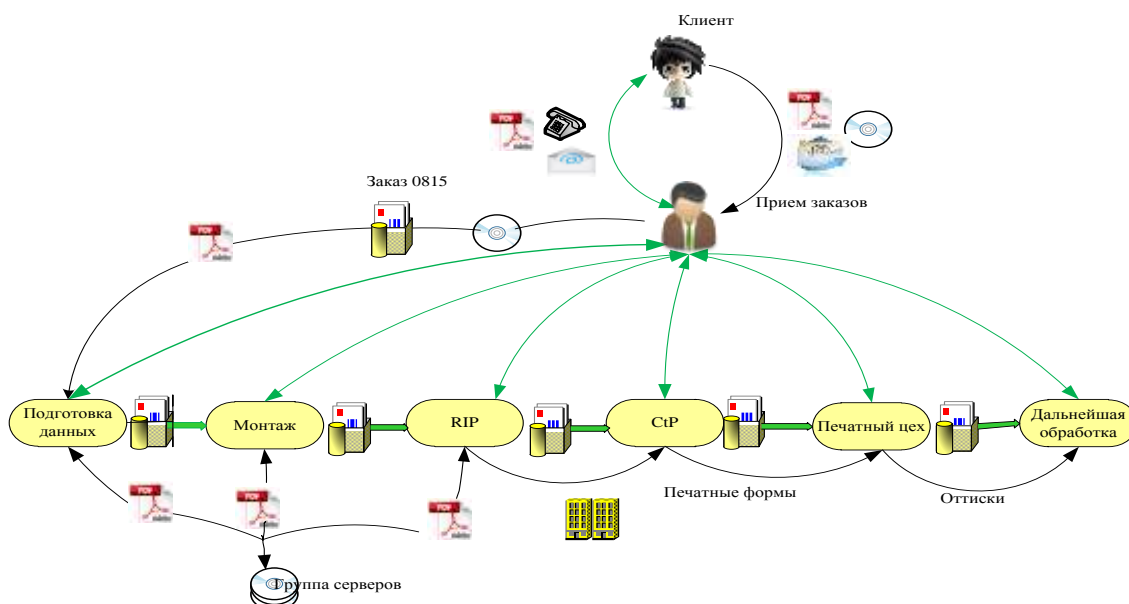


Рис. 2.2 Модель классического рабочего потока в печатном производстве

Ниже будут рассмотрены три категории систем. Классический пример приведен на рис. 2.2. Клиент и соответственно, исполнитель взаимно общаются посредством передачи метаданных (зеленая стрелка). Содержание данных (черная стрелка) обозначает их передачу и управление в типографии. Метаданные передаются в типографию по телефону или по электронной почте. Данные контента доставляются с помощью электронной почты в качестве вложений. Возможна еще передача их на CD/DVD. После того, как заказ введен в систему управления рабочим процессом, он проверяется, распечатывается и направляется на обработку и соответственно по цепочке далее в подразделения производственного процесса. В приведенном здесь примере отдельные компоненты процесса их преобразуют и обрабатывают как, например, на участках “монтажа”, “RIP” и т.д. В процессе задействовано несколько человек и несколько устройств обработки данных, много серверов. Заказчик контактирует по вопросам выполнения работы с ответственным сотрудником типографии, который должен выдавать и собирать информацию по многим участкам. Каковы очевидные недостатки этого бизнес-процесса?

Их несколько:

- содержание данных и задание по e-mail передаются по многим каналам и их можно трактовать по-разному, что влияет на эффективность производства;
- сбор производственных данных происходит, либо не происходит в определенном временном отрезке, который часто не совпадает с производственным процессом;
- задание на последовательное выполнение заказа (*Job-Tracking*) затруднено в силу наличия барьеров между подразделениями;
- установочные данные для машин должны периодически вводиться заново.

Данная схема не соответствует строго определению эффективной системы управления рабочим потоком.

Иная система представлена на рис.2.3. На базе клиент-сервера системы управления рабочим потоком (WMS) все производство автоматизировано. На начальном этапе создается рабочая карточка с заданием (Jobtickets), которая базируется на серверном процессоре. Только системный администратор может изменить задание в рабочей карточке, чтобы, например, установить стандартную частоту раstra для печатной формы в 70 линий на см.

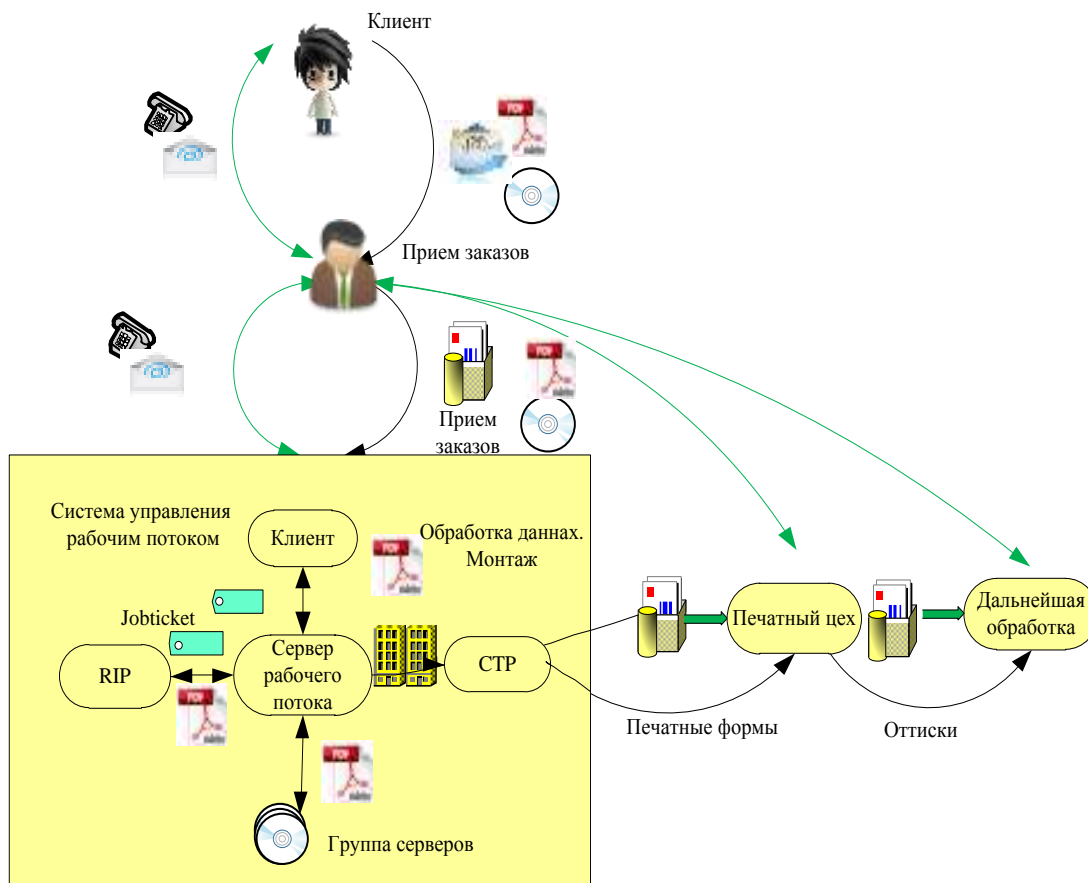


Рис. 2.3 Клиент - сервер, как основа системы управления рабочим потоком (WMS)

Каждый оператор допечатного подразделения этого сделать не может, только отдельным из них разрешается вносить коррективы. Через эти строго определенные задания рабочей карточки, которая индивидуально настроена на каждый заказ, протекает процесс производства продукции и получается необходимый результат и высокое качество. Черные стрелки на желтом поле системы управления рабочим потоком (WMS) на рис.2.3 показывают как передачу контента, так и управляющих воздействий. Данная концепция представляет собой значительное улучшение предыдущей модели, представленной на рис. 2.2.

Однако и она имеет все же некоторые недостатки:

- не полностью связаны процессы ввода заказа и управления производством;
- не полностью увязан сбор производственных данных (*BDE*);
- не полностью связаны общим заданием программы работы машин в отдельных процессах.

При таком построении системы управления рабочим потоком, уже сегодня возможна электронная коммуникация подразделений в производстве и с заказчиком. Как, например, передача данных для расчета и настройки красочных зон из отдела допечатной подготовки в печатный цех и т.д.

Следующая схема интеграции и объединения подразделений системой управления рабочим потоком в пределах деятельности предприятия представлена на рис. 2.4.

Метаданные, посредством локальной сети, передаются на пультаы управления различных машин и агрегатов, и через нее осуществляется обратная связь. Оформленный заказ поступает сразу на сервер рабочего потока. При этом необходимо еще раз отметить, что такой пример не реализован на практике, он отражает лишь то, что частично есть в настоящее время на предприятиях.

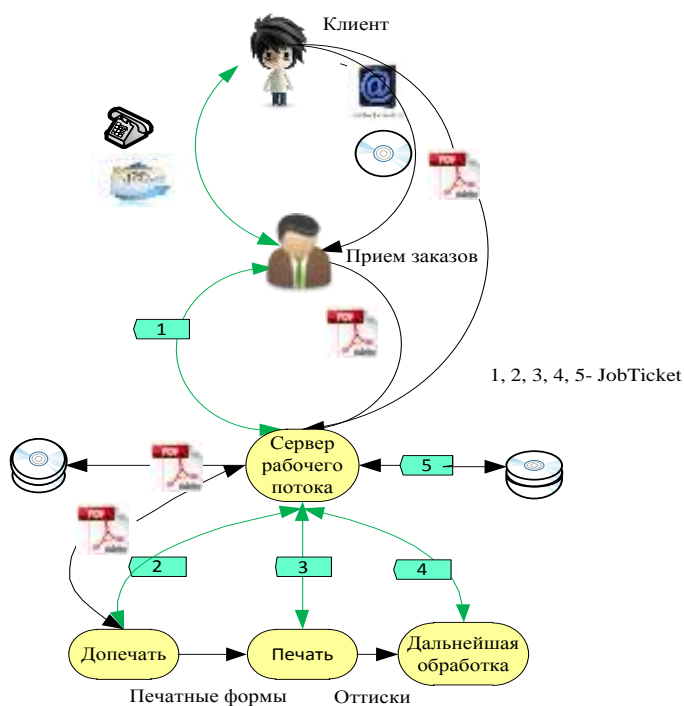


Рис. 2.4 Интегрированная система управления рабочим потоком

На рис. 2.5 представлена система, базирующаяся на обмене данными только через Интернет. Управление заказами, допечатной подготовкой и изготовлением форм может осуществляться из любой точки мира и различными фирмами. Этот процесс реализуется тогда, когда затруднена транспортировка или она требует больших затрат и времени. Хотя такая идея довольно реалистична и востребована, она будет реализована заказчиками и

типографиями в последующие годы с развитием систем рабочего потока на основе Интернета.

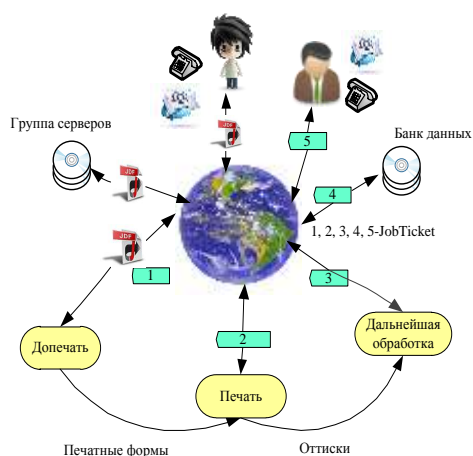


Рис.2.5 Интернет – ориентированная система управления рабочим потоком

2.6 Взаимодействие системы управления рабочим потоком и форматом рабочей карточки (Jobticket)

Последние три примера систем управления рабочим потоком были построены с использованием рабочей карточки (*Jobticket*). Ее можно охарактеризовать как программу или формат данных (в электронном виде) для осуществления работы системы управления рабочим потоком в полиграфической индустрии. Рассмотрим и подискутируем в этой связи, казалось бы, на отвлеченные три темы:

- настраиваемость;
- стандартизация;
- расширение.

Этот перечень тем хотя и является на первый взгляд отвлеченным, но при последующем рассмотрении окажется полезным.

Настраиваемость. Система управления рабочим потоком должна обеспечивать непрерывный рабочий процесс предприятия и не настраиваться постоянно. По-видимому, правильный набор функций и оборудования для выполнения заказов имеет определенные границы. Некоторые работы в типографии нельзя заранее предусмотреть. Требования

заказчика бывают большими, чем это может реализовать определенная система управления рабочим потоком и установленное оборудование.

На деле при создании WMS всегда проводится предварительный анализ и определение особенностей протекания производственного процесса. Это важная и сложная задача при реализации определенного JDF – рабочего потока на конкретном предприятии.

Управление, а также реализация стадий производственного процесса, происходит в определенное время и не всегда стандартно, имеются различные альтернативы, переходы, возвраты (см. рис. 2.6).

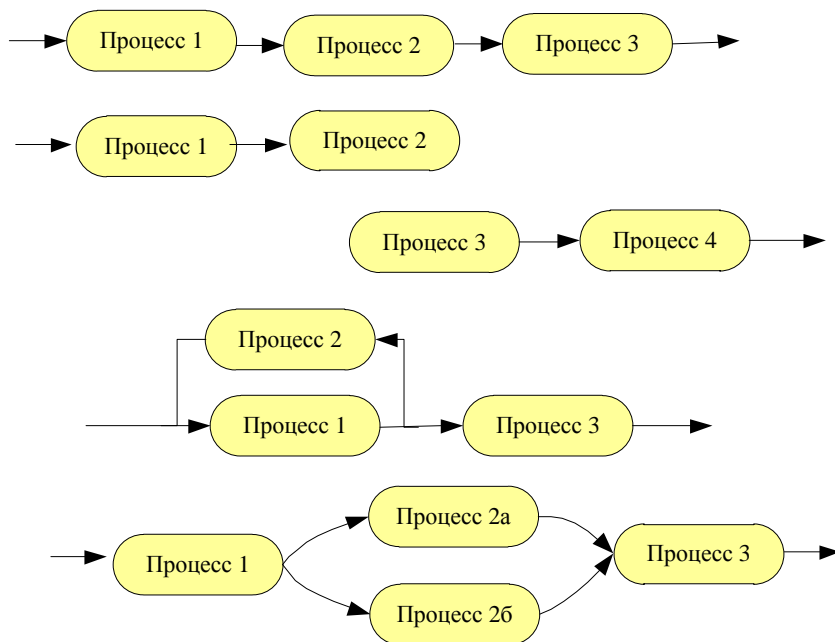


Рис. 2.6 Различные варианты протекания производственных процессов

При производстве печатной продукции в конкретной типографии всегда имеется для этих ситуаций много примеров. Вот некоторые из них:

- последовательность: первоначально печать, потом фальцовка;
- параллельность: текст и изображения готовятся одновременно;
- переходы: формные пластины экспонируют, а затем обрабатывают;
- альтернативы: печать на машине А или на машине В;
- интеграция: проба, корректура, проба, вывод.

Система управления рабочим потоком должна эти ситуации реализовывать. При этом возникает вопрос, что в рабочей карточке должно отображаться. В спецификации Job-формата имеется, например, возможность выбора различных операций. При этом

применяют известный из информатики термин *Pipe (труба)*. Он предусматривает взаимодействие процесса производства (как, например, производство печатных форм), В этом случае предусматривается, что следующий процесс может начинаться, только когда экспонирующее устройство даст команду на установку очередной формных пластины. Иными словами, с использованием правила *Pipe* можно предусмотреть описание поступления пластин из накопителя (аналогично устройству на рис.2.7).

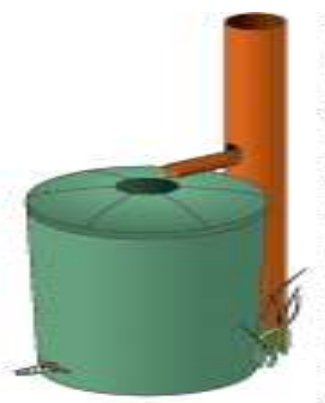


Рис. 2.7 Принцип последовательности процессов по аналогии с резервуаром

При производстве полиграфической продукции есть типичные ситуации, когда первый раз при запуске заказа надо учитывать множество технических и организационных деталей. Это значит, что рабочая карточка должна динамично принимать к решению всю вновь поступающую информацию. Например, установки подача красок по красочным зонам, как известно, нельзя производить одновременно с реализацией процесса. До начала процесса производства продукции необходимо описать детально печатный продукт и, естественно, сделать это правильно. Необходимо, например, задать частоту раstra изображения, которая должна применяться. В последующей реализации процессов производства эта информация может уточняться. Описание этой конкретной ситуации следует отразить в рабочей карточке. Должен быть и вариант, связанный с изменением параметров продукции (например, решения, вносимые заказчиком), однако все это сложно учесть для конкретной системы управления рабочим потоком. Следует еще много поработать над необходимыми решениями.

Стандартизация. При производственной интеграции с использованием системы управления рабочим потоком целесообразно закладывать стандартные

коммуникационные решения. Однако стандартизация имеет определенные границы. Что это значит? Прежде всего, необходимо максимально использовать Jobticket - формат для описания производства полиграфической продукции. Однако не всегда разработчик конкретной системы управления рабочим потоком применяет все возможности JDF – формата и функции рабочей карточки. Вариант эффективного использования готовой системы конкретным пользователем существует в том случае, если разработчик отразил, какие функции в ней блокированы, а фирмы должны учитывать, как в этом случае применять систему к производству своей продукции. Как решить эту дилемму? Ответ заключается в том, что нельзя основываться на стандартизации описания процессов, команд, содержания только одной части производства, так как при этом идея стандартизации нарушается. Выход из ситуации – расширение охвата JDF - форматом областей, для которых в настоящее время отсутствуют решения.

Однако этого недостаточно. Должен ставиться вопрос помимо «Что», но и «Кто» участвует в процессе. В системах рабочего потока возникают проблемы правильного понимания информации при коммуникации между участниками производства. Коммуникации между ними должны правильно функционировать на основе единых описаний и правил. Для этих целей создана спецификация ICS (*Interoperability Conformance Specification*). На ней более подробно остановимся в параграфе 6.7.

Еще один прагматический пункт рекомендаций заключается в том, что при использовании стандартных решений систем управления рабочими потоками в них должны предусматриваться возможности использования ежегодно появляющихся новых программных продуктов.

Расширение. Поставщики WMS должны предусматривать возможность, несмотря на стандартную модель, добавлять расширения. Требования к расширениям особенно необходимы и для Jobticket-формата данных. Что означает это свойство для формата данных?

Jobticket- информация составляется с применением правил и структур (синтаксиса) других языков и форматов. На практике это:

- PostScript (PS);
- Portable Definition Format (PDF);
- Extensible Markup Language (XML).

В главе 4 Jobticket –Formate, *Print Production Format* (PPF) и *PortabelJobticket Format* (PJTF) будут анализироваться глубже. Сейчас только укажем, что PDF кодируется в PS, а PJTF в PDF. Job-Definition Format имеет структуру XML - документа. Расширения XML объясняются в параграфе 5.2.

PS и PDF известны как языки описания страниц, точнее как документальная структура описания страниц в графической индустрии ([4], [5]). Оба формата могут быть расширены в своем языковом объеме. Это означает, что личные ключевые слова могут быть свободно выбраны и их опции могут быть зарегистрированы фирмой ADOBE (во избежание конфликтов торговых марок). Мы хотим представить возможности расширения PS на некоторых примерах. Для формата PDF аналогично. В PS переменные имена это цепочка знаков, которые обозначаются косой чертой (/). Ключевое слово *def* указывает на смысл этого имени. Например, PS-коде можно описать как:

/hev/ Helvetica-Bold def

Новые переменные имена с указанием */hev/* имеют смысл Helvetica-Bold. В этом случае новое переменное имя служит сокращением оригинального полного имени. Точно также можно описать строку:

/CIP3PreviewimageWigth 1425 def

Переменное имя CIP3 PreviewimageWigth разложено в PPF-спецификации и задает разрешение экрана в пикселях. Точно так же можно создавать переменные имена с собственными вложенными в них смыслами:

/Meine Erweiterung/ ganz_toll def

Это стандартный PS-код. Фирмы могут применять PPF с собственными переменными именами и другими PS структурами так, что в рамках их рабочего потока они будут нести дополнительную информацию, которую CIP3 не предусматривает. Эти индивидуальные расширения не нарушают теоретические построения, они часто применяются на практике как в PPF, PJTF так и в JDF. Это, конечно, при определенных обстоятельствах ограничивает совместимость между системами. Поэтому переменные имена должны применяться только для дополнения, а не служить стандартной информацией. Поэтому, если рабочий поток натолкнется на эти личные изменения и их не поймет, то он их проигнорирует. Обобщая, можно сказать, что эти три понятия: настраиваемость, стандартизация и расширение частично друг другу противоречат и нужно постараться при использовании Jobticket – Formaten, как основы системы управления рабочим, потоком найти компромисс между этими тремя направлениями.

Глава 3. Модели рабочего потока печатного производства.

В этой главе на нескольких примерах описываются четыре следующие модели рабочего потока, которые часто имеют место в печатном производстве:

- список действий;
- диаграммы перехода состояния и диаграммы действий;
- диаграммы потока (алгоритмические);
- модель Производитель - Потребитель или модель Процесс - Ресурсы.

Другие подходы моделирования для стандартных производственных процессов можно найти в [18].

Список действий (*ToDo-List*) представляет собой совокупность этапов работы, как, например, обычный список покупок. Расширенный список действий может включать в себя ответственность и время начала и конца выполнения подзаданий. Речь идёт только лишь о том, кто что делает, когда и посредством чего. Не представлены в списках действий зависимости между подзданиями или выраженные критерии решений при возможных разветвлениях рабочих процессов. Последнее - важнейший признак диаграмм переходов состояния. В дальнейших примерах представляются состояния (обозначенные как прямоугольники или окружности) и переходные состояния (помеченные стрелками). Классическими примерами являются представления состояния машины как: *нерабочее состояние*, *рабочее состояние*, *прерывание* и *поломка* (рис. 3.1). В случае ошибки в *нерабочем состоянии* или *рабочем состоянии* происходит переход в *состояние поломки*.

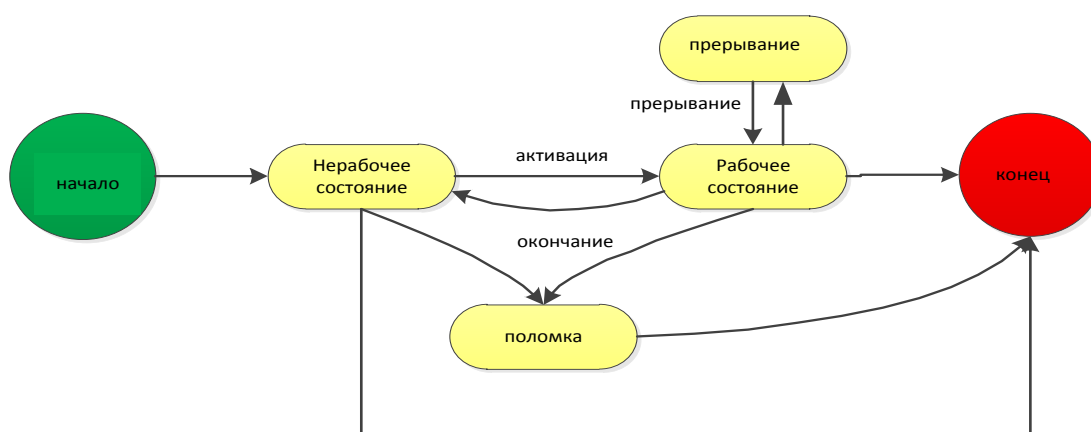


Рис.3.1 Состояния, которые машина может принимать в производстве

С помощью описания рабочего потока можно получить диаграммы похожего вида, только вместо состояний изображается действие. Стрелки представляют собой переход между действиями. В этой книге данные диаграммы называются «диаграммы действий». Модель этапов выпечки пирога, например, позволяет видеть последующие действия на несколько шагов вперед (рис. 3.2).

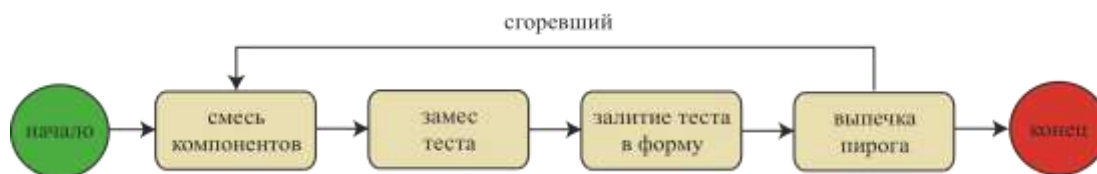


Рис.3.2 Диаграмма действий при выпечке пирога

Диаграммы потока используются сегодня для визуализации алгоритма при разработке программного обеспечения. Однако они находят также применение в процессах работы предприятия в целом, в отдельном производстве и даже могут быть полезными при планировании дел на день в обычной жизни, что можно увидеть на рис. 3.3

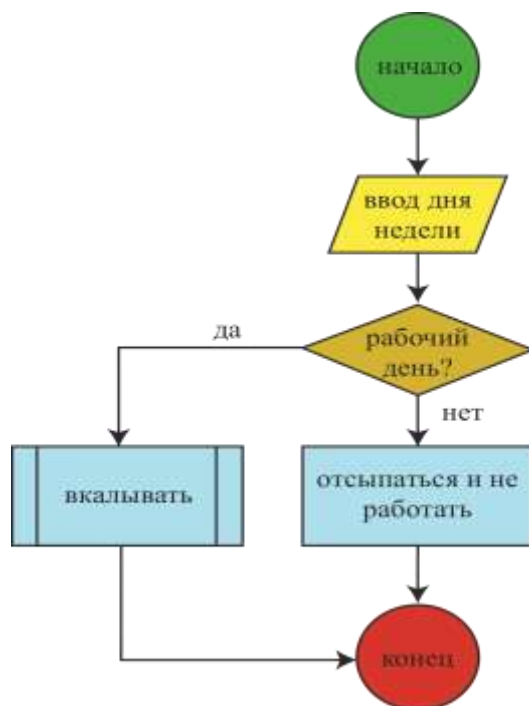


Рис.3.3 Диаграмма потока с различными операциями

Зеленый или красный круги означают соответственно начало или конец действий, желтый параллелограмм – ввод или вывод, оранжевый ромб – запрос, который позволяет

говорить только «да» или «нет». Голубой прямоугольник представляет операцию в целом, серый прямоугольник с двойными, вертикальными линиями обозначает подпрограмму, которая, как правило, обозначает новый план реализации потока. Все геометрические элементы нормированы, кроме, конечно же, цветов элементов, которые выбираются по усмотрению разработчиков и пользователей.

Модель «Производитель – Потребитель» точно описывает, что можно будет ожидать: производитель производит какой-либо продукт и помещает его в буферную память, например, ставит на полку в супермаркете. Потребитель берет один экземпляр из буфера и использует его. Оба процесса являются разделенными во времени друг от друга: производитель должен сначала обязательно что-либо изготовить, перед тем, как потребитель начнет использовать данный товар.

При описании процессов рабочего потока есть очень распространенная модель, в которой используются процессы, продукты и ресурсы. Например, процесс экспонирования и проявки формных пластин при изготовлении печатных форм производит ресурсы, которыми являются формы для печатных процессов.

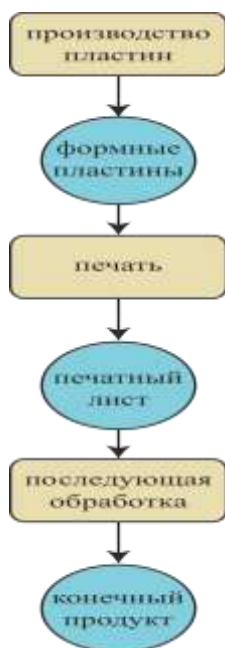


Рис.3.4 Модель Производитель - Потребитель в рабочем потоке печатного производства

Печатный процесс, в свою очередь, также производит ресурс, а именно: стопу оттисков, которые при дальнейшей обработке будут использоваться в последующих отделочных процессах и так далее (рис. 3.4).

Модель для выпекания пирога, представленная на рис.3.2, с помощью диаграммы переходного состояния, даёт нам возможность заглянуть на несколько шагов вперёд. Сами составляющие вроде муки, яиц, масла и молока, а также промежуточные результаты процесса, как тесто или заполненная форма для запекания – ресурсы. В данном случае работает эмпирическое правило: процессы являются глаголами, а ресурсы - существительными.

Модель JDF, основывающаяся на принципе Процесс - Ресурс, будет сопровождать нас по всей книге. Рассмотрим три модели этапов работы с заказом: допечатную подготовку, печать и послепечатную обработку с точки зрения менеджмента.

3.1 Управление заказом

Действия по управлению заказами до их выполнения позволяют составить следующий упрощённый список:

- предложение (от клиента к типографии);
- передача предложения (от типографии к клиенту);
- передача заказа (от клиента к типографии);
- подтверждение заказа (от типографии к клиенту).

Очевидной является зависимость во времени между действиями процессов списка. Пример упрощенного описания данной зависимости представлен на рис. 3.5.



Рис.3.5 Действия менеджмента типографии по приему заказа

Как правило, можно представить рабочий поток с помощью диаграмм более детально. Тогда каждое действие, а также каждый переход между действиями в диаграмме действий, может представлять собой множество запросов и операций согласно терминологии диаграмм потока. На диаграмме рис.3.6 детально изображены действия, представленные на рис. 3.5, (см. также рис. 3 в [40]).

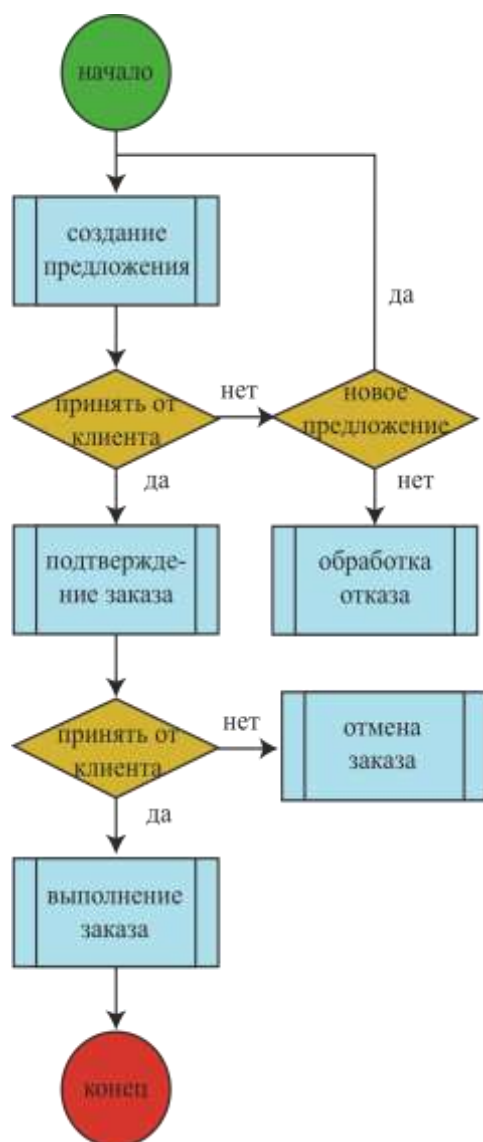


Рис.3.6 Диаграмма рабочего потока с соответствующими действиями по приему и выполнению заказа

3.2 Рабочий поток в допечатной подготовке

Отдел изготовления печатных форм типографии, который получает готовые PDF - файлы, руководствуется определённым набором действий (для уточнения терминов воспользуйтесь глоссарием в конце книги):

- проведение предварительной проверки, а, при необходимости, корректуры поступивших данных;
- нормализация PDF - файлов;
- преобразование цветового пространства;
- треппинг;
- подбор и разработка одного или более макетов (лист-макет);
- спуск полос;
- риппинг (рипование, создание полиграфической растровой структуры) данных для выполнения контрольного спуска полос на плоттере;
- пробная печать;
- риппинг данных для предварительного просмотра изображения перед формными процессами;
- риппинг данных для экспонирования, экспонирование формных пластин и их обработка;
- расчет значений установок красочных зон.

Мы хотим только кратко остановиться на различии между *изготовлением листа-макета* (*монтаж печатного листа - stripping*) и *спуском полос* (*спуск - imposition*), так как очень часто на практике эти два понятия четко не разделяют. Изготовление листа-макета происходит после определения размеров печатного листа (запечатываемой части бумажного листа) и бумажного листа (как результат-позиция печатного листа на печатной форме). Вслед за этим определяются в своих размерах и размещаются при акцидентной печати страницы или сигнатуры на виртуальном листе с помощью схемы спуска полос. Для этого необходимо также учитывать тип переплета и тип печати обратной стороны листа, а также поля захвата и другие поля, такие как шлейф сфальцованного листа, шлейф тетради или поля для клеевого скрепления.

Страницы, а вернее единицы использования площади бумаги, определяются как заполнители без содержания. Заполнители страниц также называются образцами страниц.

Наконец, на лист наносятся шкалы для контроля печати, метки для высечки и фальцовки и др. В итоге получается макет (монтажный лист) или лист-макет.

При спуске полос нумерация страниц или сигнатуры, которые имеются в PDF -формате, «вручную» или автоматически упорядочены, совмещаются вместе с метками в одну структуру. Образцы страниц заполняются контентом, вследствие чего получается файл, который описывает лист (рис.3.7), например, в PDF -формате.

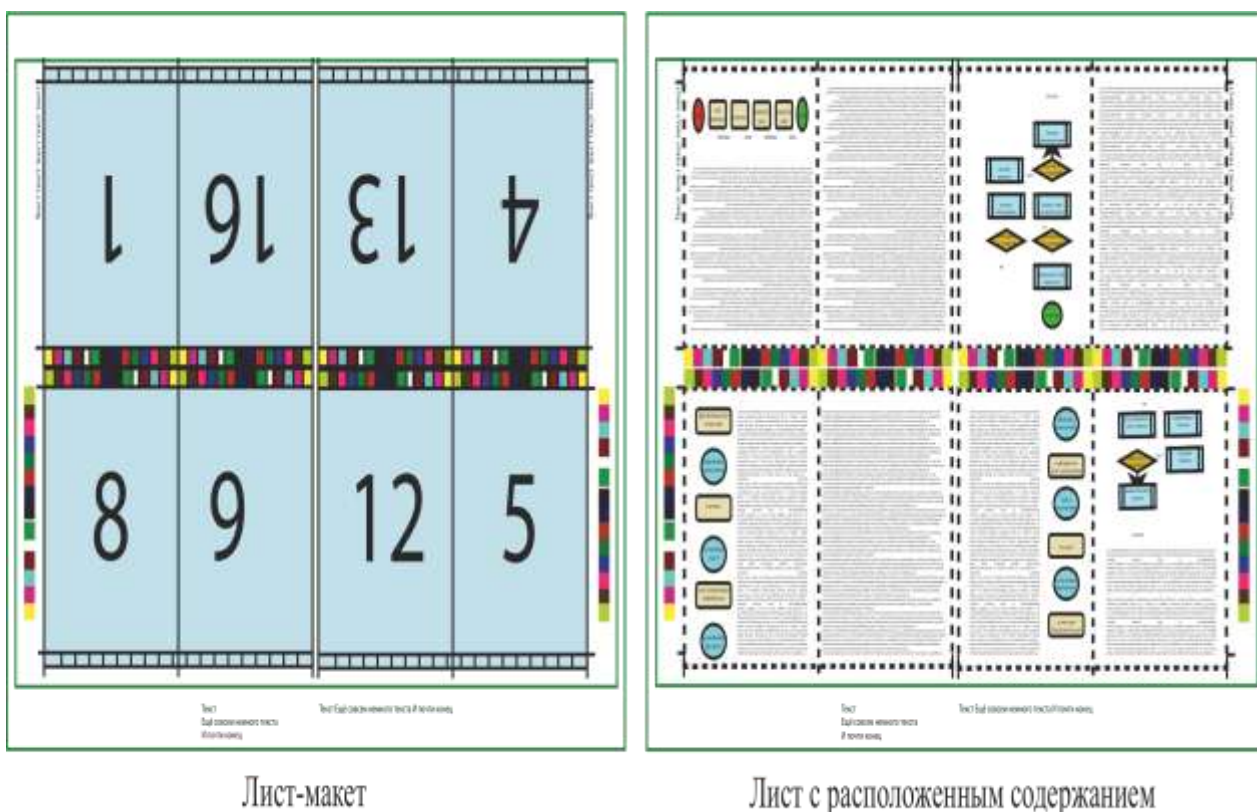


Рис.3.7 Образцы: лист-макет и спуск полос

Раньше оба процесса выполнялись в пределах одной программы (программа спуска полос или программа сборки). Сегодня они в большинстве случаев идут раздельно друг от друга. Разработка листа-макета, которая еще проводится под контролем оператора, часто происходит с применением автономного программного обеспечения настольных издательских систем, в то время как автоматический процесс спуска полос выполняется позднее системой.

Пример диаграммы действий, в которой рассмотрены отмеченные процессы, представлен на рис. 3.8.

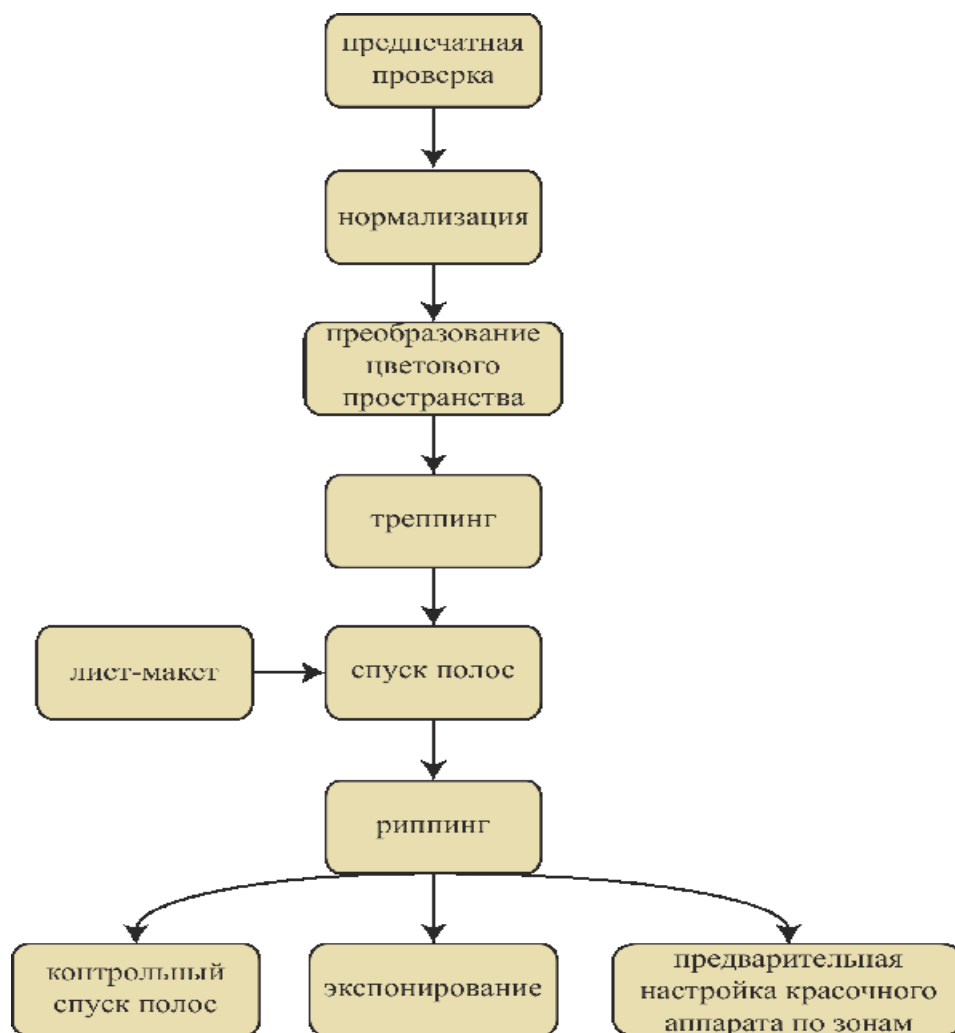


Рис.3.8 Диаграмма действий в допечатных процессах

При углублении в тему «процесс - ресурсы – модель» становится более сложной (рис.3.9). На диаграмме процессы обозначаются прямоугольниками с закругленными углами, а ресурсы окружностями, отличающимися по цвету: R1-R10 – голубые, а R11-R24 – светло - коричневые.

Голубые окружности являются входящими ресурсами для процессов, в то время как светло-коричневые – выходными. Голубые окружности являются параметрами ресурсов, которые предоставляют детальную информацию для соответствующего процесса. Например, риппинг-ресурс R7 содержит информацию о растровом процессе (тип растра, частота растра, угол растра и т.д.), а лист-макет-ресурс R5 содержит сведения о схеме спуска полос, типе переплета, перевороте листа и т.д.

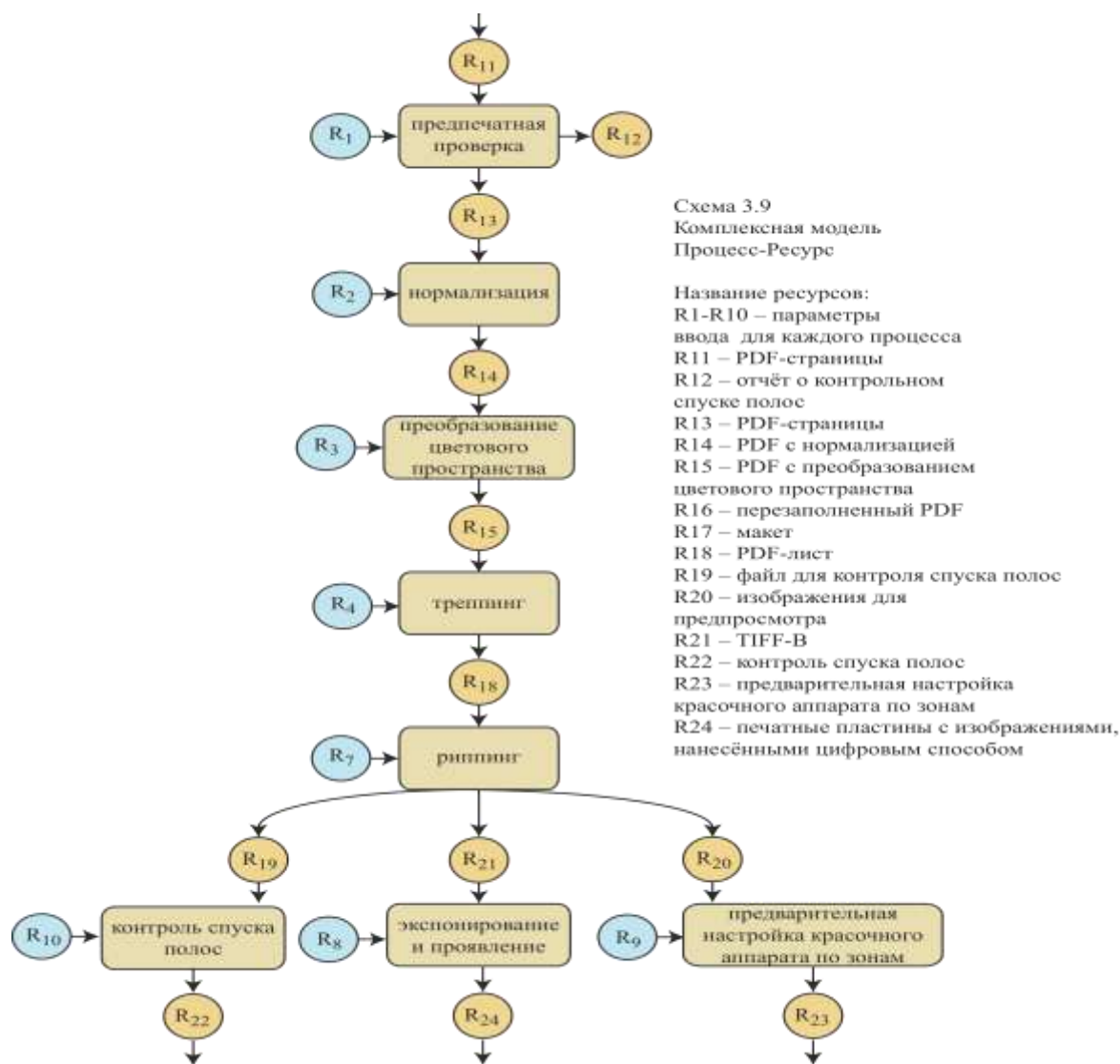


Рис.3.9 Комплексная модель «процесс-ресурс»

От подробного описания всех деталей, которые можно найти в ресурсах, мы должны отказаться, так как это займет много места. На рис. 3.9 в списке кратко представлена расшифровка ресурсов светло-коричневого цвета. Разумеется, необходимо отметить, что

алгоритм на рис. 3.9 описывает только пример автоматизации процесса изготовления печатных форм, который приводится в качестве одного из возможных. Так в главе 12 можно увидеть, что при изготовлении упаковки, а именно складной коробки, используется другой набор работ. А что касается названий ресурсов на рис. 3.9, то мы предположили, что они не всегда служат образцом. Так ресурсы R11 и R12 были названы «PDF-Страницы». Более общим и правильным, и в то же время более трудным была бы речь о «Порядке страниц, сформулированном на языке описания страниц».

Ресурс R11 являлся бы типовым для процесса создания макета страницы. Оба ресурса R22 (контрольный спуск полос) и R24 (готовые печатные формы) являются не только наборами данных, но и вещественными. R22 часто согласовывается с заказчиком или может сразу использоваться в процессе печати. Ресурсы R23 (данные настройки красочного аппарата по зонам) и R24 служат входными ресурсами. На модели (рис. 3.9) хорошо заметно, что процесс-ресурс-модель действительно содержит зависимости между процессами, но они не имеют определенно заданного порядка описания. Так не указывается - нужно ли сначала выполнять контрольный спуск полос, а потом изготавливать печатные формы или наоборот. Последнее является на самом деле абсурдным.

В главе 9 рассматривается метод, который гарантирует точность и пунктуальность выполнения процессов (утверждение-процесс).

Какой точности должен быть процесс? Можно задавать более общие модули, такие, как можно увидеть на рис. 3.4. С другой стороны, можно прописывать также подробно и точно, как и на рис. 3.9. Так можно было бы процесс «Экспонирования и проявки» (при соответствующей пластине и при соответствующем экспонировании) разделить на: экспонирование пластины, нагрев, химическое проявление, промывка, гуммирование, термообработка. Также возможно, что экспонирование могло бы разбиваться на: взятие формной пластины из стопки, подача на барабан экспонирования, крепление, само экспонирование, снятие после экспонирования. И так можно было бы всегда во всех деталях рассматривать каждый процесс, вплоть до нажатия кнопки запуска мотора.

Правильным является выбор каждого события, которое для получения продукции состоит из некоторых необходимых действий и выполняется с помощью одного устройства и программного обеспечения и определяется как один процесс. Так никто не будет позиционировать формную пластину на экспонирующий барабан, если при этом не

проводится закрепление и экспонирование. Также было бы бессмысленным раздробление процессов экспонирования. Процесс экспонирования и процесс обработки, в свою очередь, разумно определять отдельно. В противном случае определение процесса на рис.3.4 также было бы нецелесообразным, так как изготовление печатных форм и их последующее применение требует большого количества независимых друг от друга устройств и приложений.

JDF - модель на деле выглядит аналогично схеме на рис. 3.9. На этой схеме, например, процесс риппинга не рассматривается так подробно, как приведено на рис. 3.10.

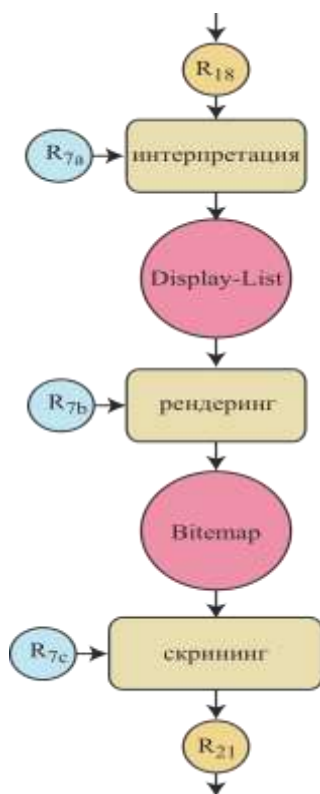


Рис.3.10 Процесс риппинга

Легко заметить, что процесс риппинга разделяется на три этапа: *интерпретация (Interpreting)*, *рендеринг (Rendering)* и *растрирование (Screening)*. Каждому из трех этапов-подпроцессов соответствует свой ресурс: R7a – *параметры для интерпретации (InterpretingParams)*, R7b – *параметры для рендеринга (RenderingParams)*, R7c – *параметры для растрирования (ScreeningParams)*.

Объясним кратко три этих понятия (рис.3.11). Процесс *интерпретации (InterpretingParams)* анализирует язык описания страниц и упрощает структуру данных. В параметрах интерпретации, например, может находиться запись о том, нужно ли адаптировать изображение после изменения его масштабов. Результатом процесса

интерпретации является структура данных (не стандартизированная), которая называется *Display-liste*. Процесс рендеринга (*RenderingProzess*) в качестве главной задачи выполняет конвертацию математических (цифровых) данных в контур изображения. В результате получается структура пикселей, которая называется *Bytemap*. Данная структура пикселей представляет собой записи в параметрах рендеринга (*RenderingParams*). Полученная структура пикселей отражает и насыщенность цвета, которая позволяет передавать различные оттенки красок.

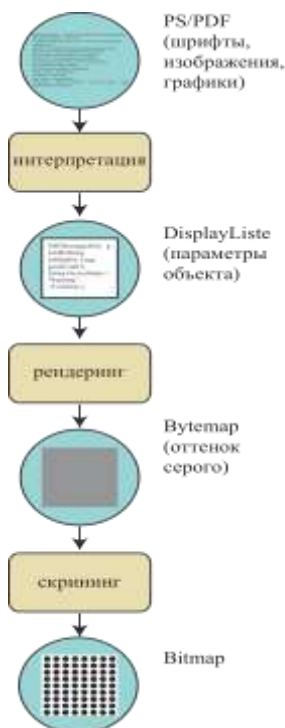


Рис.3.11 Процессы рендеринга и растривания

В *процессе скрининга (Screening-Prozess)* структура пикселей преобразуются в полиграфический растр, а именно в периодический (амплитудно-модулированный) (AM-Raster) или непериодический (частотный) растр (FM - Raster). Информация, которая является важной для растрового процесса (тип, частота и т.д.), передается в *параметрах скрининга (Screening-Params)*. Более подробные объяснения можно почерпнуть, например, в [22] или в [29]. Итак, почему же процесс риппинга рассматривается в качестве трех подпроцессов? Обоснованием может являться то, что процесс скрининга не всегда должен выполняться. Для нанесения изображения на формную пластину или для получения материальной пробы необходимо проводить растривание, что не всегда является необходимым для цифровой печати. Следует еще раз подчеркнуть, что речь идет о модели

рабочего потока, а не о самом рабочем потоке. Выходные ресурсы содержат только описание проэкспонированных формных пластин, а не самих пластин в целом. Несмотря на это, речь идет о *физических ресурсах (physischen Ressource)*. Однако, чтобы не перегружать текст, в этой книге часто не делается различие между моделью и реальностью. Например, мы и дальше будем продолжать использовать термин *ресурс-пластины (PlattenRessource)*.

Сами параметры ресурсов не обязательно должны содержаться непосредственно в модели автоматизации процесса. Например, PDF - файл, который определяет элементы для печати страниц или листов, не встроен в JDF-файл. Вместо этого PDF - файл можно найти только в файловой системе. В JDF -файле находится только лишь ссылка на PDF - файл. Возможно, у Вас возникнет вопрос, кто на самом деле должен составлять модель рабочего потока? Так ли это, что оператор должен для каждого задания на печать подготовить описание этого рабочего потока, чтобы производство более или менее было автоматизировано? И если бы так было на самом деле, не сильно ли это затратно? Сейчас, в действительности, в этом нет необходимости. Из каких источников тогда берется описание автоматизации процесса? Ответ заключается в том, что информация по следующим направлениям автоматически получается из:

- калькуляции / система управления заказами;
- данных печати от заказчика;
- параметров по умолчанию (базовые величины, значения по умолчанию);
- графических масок ввода, которые заполняются пользователями в производстве;
- систем электронной коммерции (eCommerce), в которых заказчик может задать информацию о печатных работах.

Описание рабочего потока протекает, следовательно, без содействия и без эрудиции пользователя. Соответствующий администратор или инженер по производству определяют требования к базовым величинам.

3.3 Процесс листовой офсетной печати

Перечень операций для типичной печатной работы в листовой офсетной печати выглядит следующим образом.

Подготовка:

- чтение папки заказа;
- подготовка бумаги;
- установка подачи бумаги;

- заполнение красочного аппарата краской.

Настройка (наладка) печатной машины:

- смена печатных форм;
- пробная печать, извлечение листа и его визуальный/измерительный контроль;
- настройка регистра;
- настройка красочного аппарата.

Печать тиража:

- печать, извлечение отдельных листов и их визуальный/измерительный контроль;
- регулировка подачи краски;
- дозаправка краски;
- смена стопы бумаги;
- чистка офсетного цилиндра.

Завершение работы:

- смывка офсетного цилиндра;
- смывка красочного аппарата;
- заполнение папки выполнения заказа.

Является ли из этого рациональным моделирование четырех процессов: «подготовка основы», «настройка», «печать тиража» и «завершение задачи»? Или каждый подпункт процесса должен быть отдельным процессом? Последнее, конечно же, не рекомендуется, так как отдельные виды деятельности не являются взаимно независимыми. Однако разделение на четыре процесса также является проблематичным, прежде всего потому, что подпункты обычно идентичны или, по крайней мере, подобны. Действия при настройке или печати тиража являются на самом деле одинаковыми, только они имеют различный результат: в первом случае макулатура, а во втором тиражные листы. Также стоит отметить, что схема списка заданий является неизменной (стабильной). Так на практике работают по принципу шестеренки: в то время как работа по производству какой-либо продукции не закончена, печатник уже читает информацию о следующем заказе. Кроме того не все виды деятельности являются обязательными. Например, когда в машине уже заряжена бумага, то выпадает из подготовки основы пункт «установка подачи бумаги».

Консорциум CIP4 решил создать очень простую модель процессов-ресурсов для офсетной печати, как, например, это представлено на рис. 3.12. Для того чтобы все-таки выявить отличия между отдельными видами деятельности в офсетной печати, говорят о «состоянии», в котором находится процесс печати.

При цифровой офсетной печати (в данном случае печатные формы создаются в печатной машине) и ин-лайн послепечатных процессах, а также при рулонной офсетной печати, в печатно-отделочном комплексе происходит больше процессов, не только печать, но и высечка, фальцовка и т.д.

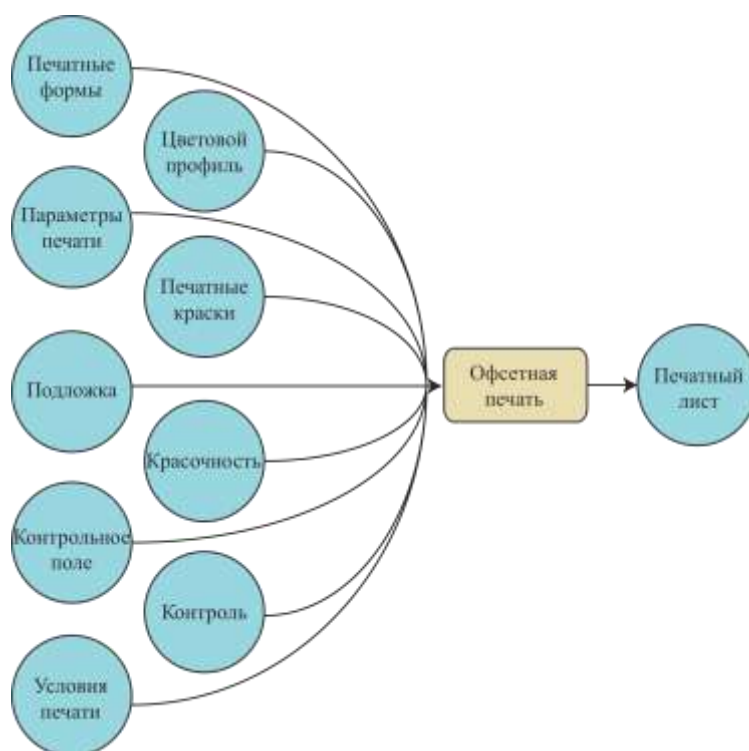


Рис.3.12 JDF - модель для офсетной печати

На рис 3.12 можно заметить, что некоторые входящие ресурсы должны быть доступными, другие, например, контроль (проверка/исправление) являются процессами по требованию. Также видно, что данная модель мало говорит о самом процессе печати. Если Вы хотите ознакомиться с данной темой более подробно, то необходимо использовать диаграмму потока (алгоритмическая схема).

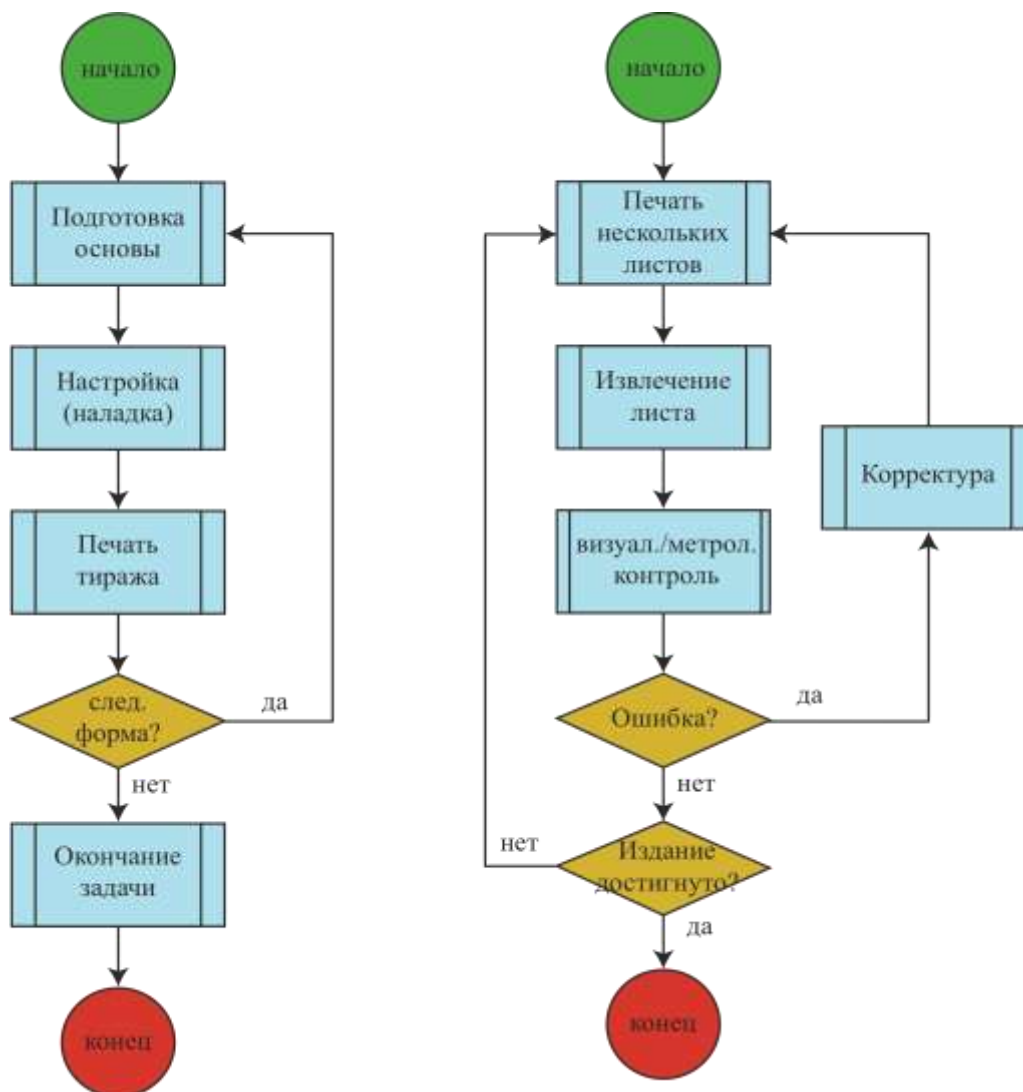


Рис.3.13 Диаграмма потока, которая описывает листовую офсетную печать

На рис. 3.13 слева представлены главные уровни списка активности, которые были определены выше как диаграмма потока, где каждый уровень обращается в подпрограмму. Подпрограмма «печать тиража» представлена на правой части схемы. Здесь можно найти множество других подпрограмм и составить на их базе диаграммы потока для подпрограмм «извлечение листа» и «оптический и/или метрологический контроль». Таким

образом, всегда имеется возможность углубиться и повысить степень детализации описания.

3.4 Пример модели послепечатной обработки

Послепечатная обработка разнообразна, поэтому бывает трудным спроектировать ее целостную модель. Общий список операций и подпроцессов был бы также длинным и всегда не полным, поэтому мы не считаем рациональным составление такого общего списка. Вместо этого мы хотим только рассмотреть модель «Процесс-Ресурс» на примере брошюры скреплением скобками. Ресурс R7 представляет собой входящий ресурс для дальнейшей обработки (например, лист-оттиск), который поступил из печатного цеха. Ресурсы с R8 по R13 представляют собой описание промежуточных продуктов, которые создаются в течение различных процессов послепечатной обработки продукции.

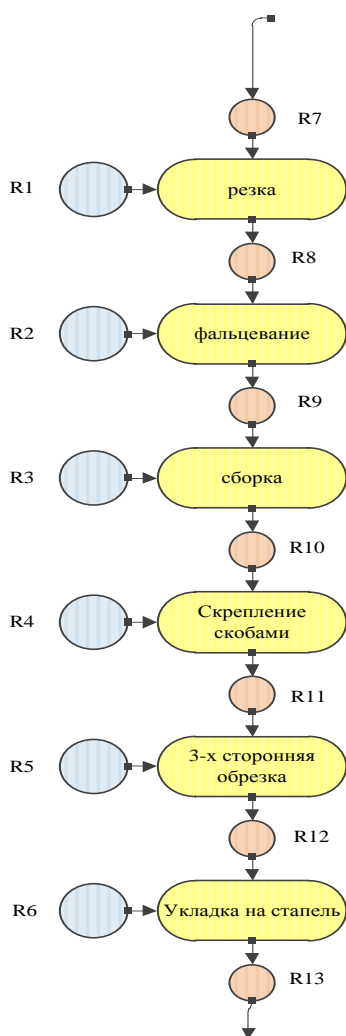


Рис.3.14 Процесс - ресурсная модель производства брошюры скреплением скобами

На рис.3.14 следующие обозначения:

- R7 - лист-оттиск;
- R8 - лист для фальцовки;
- R9 - сфальцованный лист (тетрадь);
- R10 - блок;
- R11 - скрепленные скобами тетради блока;
- R12 - обрезанный блок;
- R13 - подобранный продукт.

Ресурсы от R1 до R6 являются параметрами ресурсов, которые несут в себе информацию для каждого процесса. Мы хотим представить пару типичных записей для каждого ресурса:

- R1 - позиция ножа/меток для резки, размещение;
- R2 - последовательность (порядок) фальца, позиция фальца, размещение;
- R3 - порядок сбора компонентов;
- R4 - число, позиция, ширина, угол и форма скоб;
- R5 - ширина и высота конечного продукта;
- R6 - максимальная высота и вес, число напечатанных продуктов в стопе.

Размер печатных листов содержится в описании JDF в ресурсе R7, а не в ресурсе R1. Свойства бумаги, которые являются особенно важными для фальцовки, содержатся в ресурсе R8, а не в ресурсе R1.

Итак, мы увидели примеры автоматизации процессов допечатной подготовки, печати и послепечатной обработки в виде списков операций, диаграмм активности, диаграмм потока и моделей Процесс-Ресурс. По нашему мнению, модель Процесс-Ресурс является особенно приемлемой, так как она понятно и просто визуализирует последовательность процесса. Диаграмма потока с другой стороны может быть полезной для представления сюжетной линии в процессе. Список операций (активностей) является, возможно, самым простым для составления, он позволяет ознакомиться лишь с наименьшим количеством информации. Кроме того, список активностей склоняет нас к тому, что все записи ведутся в одной группе, что приводит к неструктурированному результату, трудному для понимания.

Задания для самостоятельной работы:

- спланируйте следующую неделю с помощью диаграммы потока. Определите альтернативы для активности (действий), когда не выполняются условия;
- опишите приготовление Вашего любимого блюда в деталях, используя модель Процесс-Ресурс – и не забудьте о первом блюде и десерте!

Глава 4. Классические метаданные и их применение

После того, как в параграфе 2.4 было проведено обсуждение и объяснение разницы между *контент* - данными (данные печати) (*Content-Daten*) и метаданными (*Meta-Daten*), необходимо представить несколько примеров для иллюстрации.

Начнем с *Exif* - формата, который позволяет производить запись технических характеристик и настроек с помощью цифровой камеры. Представим также IPTC - стандарт для фотографий.

Платформа XMP относится к *Extensible (расширяемая) Metadata Platform* и рекомендуется фирмой Adobe для различных форматов изображений и документов. Идея концепции заключается в передаче информации через границы приложения. Так, например, метаданные, которые были записаны в TIFF - формате, помещены в изображение в InDesign - документе, а также в файле в формате PDF, который экспортируется из InDesign. Таким образом, имеется возможность найти информацию в PDF-файле с выходными данными при повторной печати через несколько лет, в течение которых файл не использовался.

В параграфе 4.2 подробно обсуждается *Print Production Format* (формат печатной продукции), после того как он был кратко рассмотрен в Главе 1, как важнейший формат, предшествующий Job Definition Format.

В последнем параграфе данной главы речь идет о предшественнике JDF-формата – PJTF (*Portable Job Ticket Format*) фирмы Adobe. Этот формат тесно связан с не очень актуальным форматом *Extreme-RIP-Architektur* фирмы Adobe. Однако на протяжении долгого времени на RIP - Architektur производителям систем рабочего потока - CtP выдавалась лицензия на его использование и сегодня его можно встретить на рынке.

Содержание этой главы не является важным для дальнейшего понимания глав, связанных с JDF, и может быть пропущено. Однако поскольку многие форматы в настоящее время используются в метаданных, прежде всего JDF / JMF, PPF и PJTF, у нас есть основания считать их основой классических метаданных, поэтому они приводятся в этой книге. Кроме того, вполне возможно, что в будущем XMP и JDF могут совместно использоваться для систем рабочего потока.

4.1 Метаданные для фотографий и документов

Существуют три основные причины, почему фотографии сопровождаются метаданными:

- возможность упрощения идентификации и поиска фотографий в больших архивах;
- предотвращение нарушений авторских прав путем записи данных об авторских правах в фотографиях;
- описание технических деталей фотографий позволяет фотографам через определенное время использовать настройки камеры, при которых была сделана фотография, а также позволяет получить обзор использования фотографии для различного применения.

Важнейшими вопросом при разработке метаданных являются с одной стороны спецификации, которые должны сохранять информацию об изображениях, и, с другой стороны, сохранение этих данных при конвертации.

Exif и IPTC

Exif относится к *Extended Interchange Format* (расширенный формат обмена). Он был спроектирован японской ассоциацией производителей электроники и информационных технологий (*Japan Electronics and Information Technology Industries Association – JEITA*) [27]) специально для цифровых камер. Однако некоторые из этих метаданных используются в программах сканирования. Для каждой фотографии, которая была создана с помощью цифровой камеры, в файле с изображением сохраняется информация о времени, когда была сделана данная фотография, а также о настройках камеры, которые были установлены во время съемки (выдержка, разрешение, апертура и т.д.). С помощью различных программ возможен просмотр, модификация и оценка этих данных. На рис. 4.1 представлены Exif-данные, которые были получены с помощью цифровой камеры автоматически.

Версия спецификации Exif 2.2 [28] определяет гораздо больше полей, чем показано на рисунке, например, такие, как авторские права, комментарии или GPS-информация (координаты съемки). Кроме того она описывает структуру данных Exif как для файлов с изображениями в формате JPEG, так и для файлов с изображениями в формате TIFF. Другие форматы в этом случае не поддерживаются, в том числе и PDF. Тем не менее, Exif-записи переносятся в этот формат, однако информация строится по другой структуре, такой как описано в сведениях про XMP.

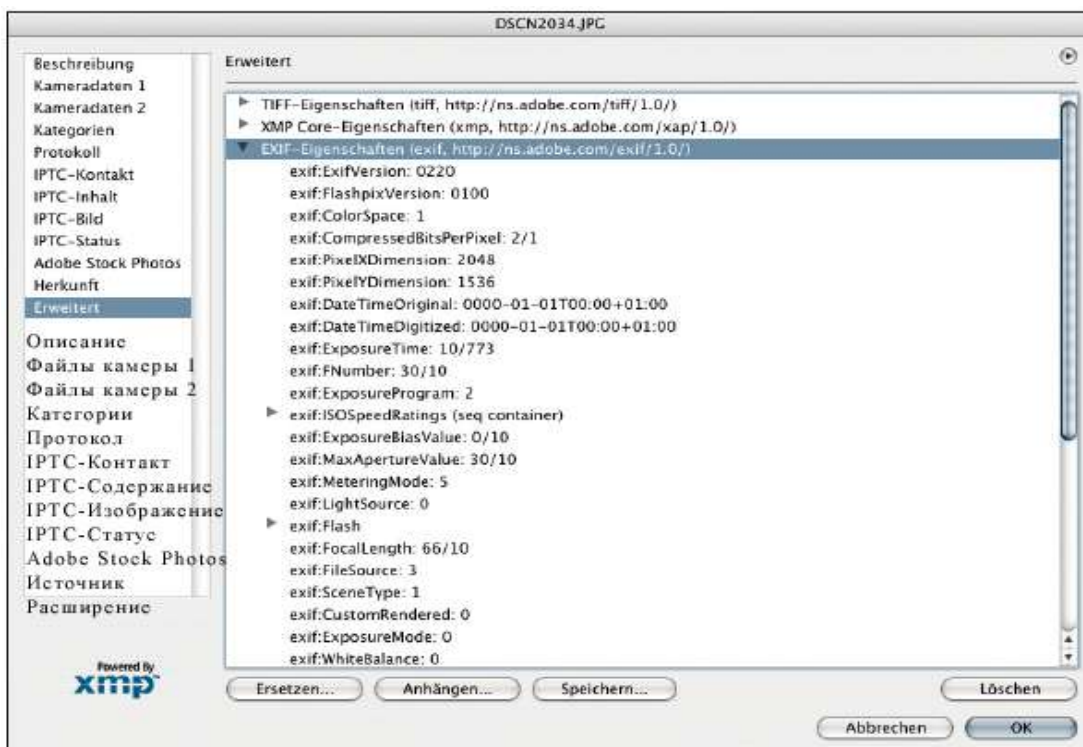


Рис. 4.1 Exif- данные JPG - изображения

Exif -информация может встраиваться в JPEG или TIFF, т.к. оба формата данных являются расширяемыми. Буква «Т» в аббревиатуре TIFF расшифровывается как: «*Tag/Tagged*», что означает «*тег*», «метка» или «помеченный» (также «ярлык»). Фактически это говорит о том, что каждая информационная группа предоставляет сохраненному изображению (его характеристикам) определенное опознавание (идентификатор) в форме положительного целого числа. В общей сложности существует 2^{16} (=65536) различных, возможных тегов. Только небольшое количество из них до сих пор было занято, поэтому имеется возможность использования новых дополнительных тегов [3]. Для изображений JPEG возможность расширения реализуется похожим способом. Сжатые по JPEG-нормам, изображения, как правило, хранятся в формате данных JPEG File Interchange Format (JFIF) (формат обмена файлами JPEG) [20], который, как и TIFF, наделен определенными информационными единицами с идентификаторами, которые в данном случае называются не «тегами» (*Tag*), а «сегментами» (*Segments*). Например, номер 225 является номером для Exif- данных.

Некоторые цифровые камеры сохраняют свои фотографии в формате RAW, который является форматом конкретного производителя. В этом случае Exif - данные хранятся в отдельном текстовом файле.

Уже с начала 90-х годов распространялись метаданные для описания изображений, которые являются похожими на Exif по многим пунктам и известны как *Information Interchange Modell (IIM)*(Модель обмена информацией.). *IPTC (International Press Telecommunication Council)* является международной ассоциацией, в которой представлены информационные агентства и национальные ассоциации газет. Она имеет право собственности на данный продукт, в котором изображения описываются с помощью ключевых слов (рис. 4.2).

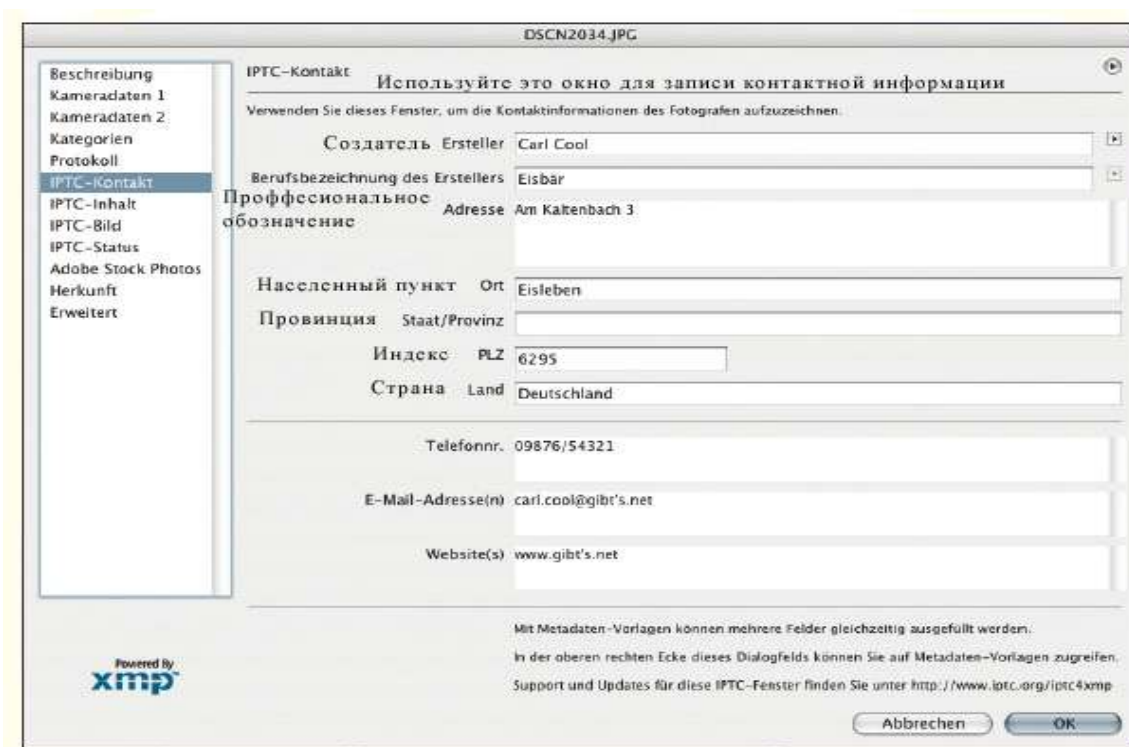


Рис. 4.2 IPTC - данные в JPG - файле

XMP

В 2004 году фирма Adobe выпустила гораздо более широкую версию *Extensible Metadata Platform (XMP)*, которая определяла метаданные не только для изображений, но и также

для различных видов документов [2]. Идея данной спецификации заключается в том, что введенные метаданные остаются после преобразования формата. Имеются предпосылки, что, с одной стороны, является возможным вложение метаданных в специальный формат данных спецификации XMP (как при TIFF, JPEG JPEG 2000, GIF, PNG, HTML, PDF, AI, SVG/XML, PSD, PostScript и EPS), и также, что приложения, которые выполняют трансформацию формата, также используют XMP - данные.

На самом деле XMP - информация может сохраняться отдельно за пределами файлов приложения. Но это все же скорее не типично и применяется максимум в базе записей данных. XMP - данные могут относиться как ко всему документу, так и к отдельным его компонентам. Можно привести пример ситуации, когда графики и изображения (фотографии), которые встроены в один PDF -документ, ассоциируются с собственными метаданными. На рис.4.3 представлен принцип действия, в то время как на рис. 4.4 представлена страница книги, на которой можно увидеть XMP - информацию, выведенную из компонентов.

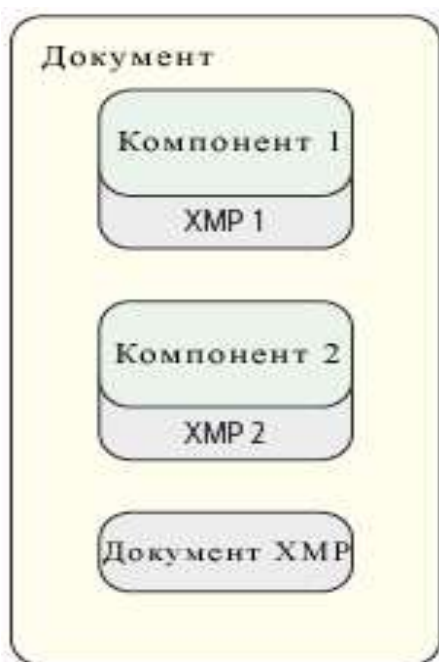


Рис. 4.3 Принцип компоновки документов XMP из компонентов

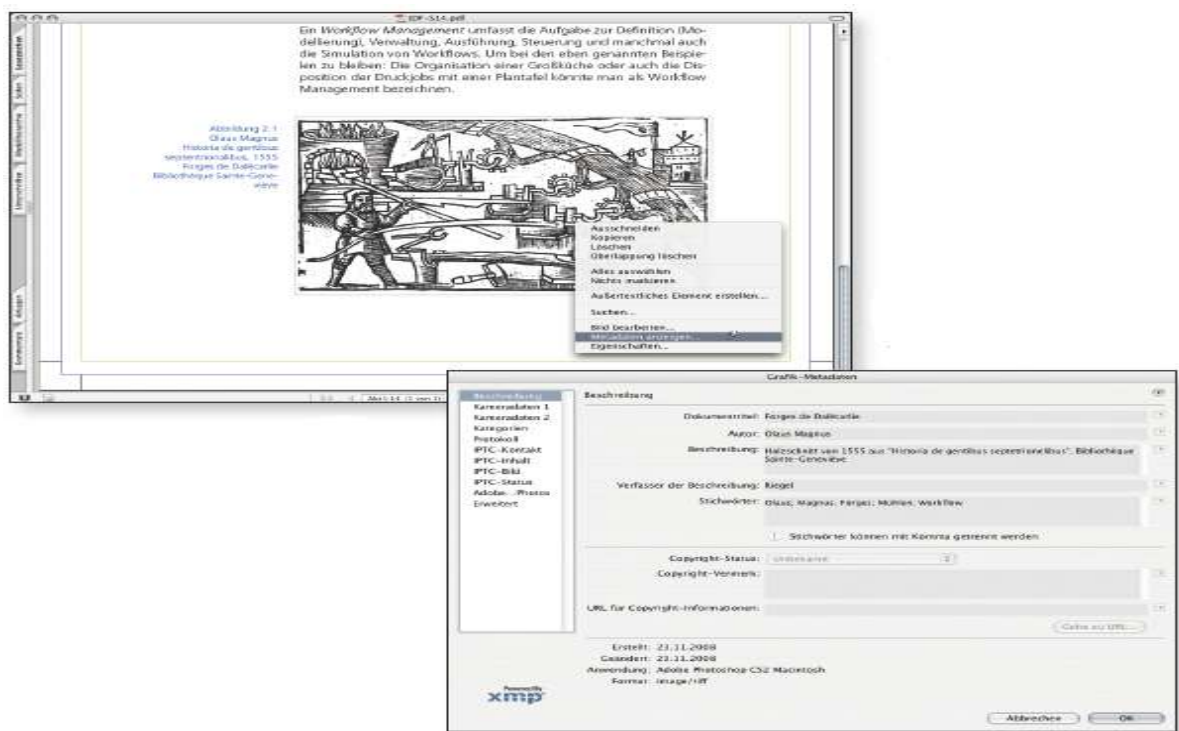


Рис. 4.4 XMP - информация об изображении в пределах PDF - файла

Метаданные XMP могут быть сохранены в XML - синтаксисе. Обсуждение XML проведем в следующей главе, однако мы немного опережим событие. XML - это очень легкое расширение, которое позволяет создавать новые приложения. При этом имеется возможность использовать несколько схем одновременно. Фирма Adobe заранее определила несколько схем, таких как:

- схема Dublin Core;
- базовая схема XMP;
- схема управления правами (Rights Management) XMP;
- схема страничного текста XMP;
- Exif-схема для определенных свойств Exif.

Полный список можно найти в [2]. Здесь перечислено только несколько примеров из множества записей, которые реализуются через эти схемы. *Название (title)* и язык (*language*) ресурса являются свойствами, которые определяются в *схеме Dublin Core*. *Базовая схема XMP (XMP Basic Schema)* предоставляет информацию о *дате создания (CreateDate)* или *имени приложения (CreatorTool)*, которые были созданы вместе с ресурсом. *Схема управления правами (Rights Management) XMP* показывает владельца прав, в то время как *схема страничного текста XMP (Page-Text Schema)*, например, может

запоминать количество страниц (*Npages*) документа. В схеме Exif определяется большинство из свойств Exif, опять же под XMP. Таким образом, метаданные цифровых фотографий могут интегрироваться в структуры XMP. Конечно, не только Adobe, но также и другие фирмы могут определять такие схемы и потом встраивать собственную информацию (со спецификацией определенного производителя) в файлы. Так производитель WMS (Workflow Management System – система управления рабочим потоком) может использовать возможности XMP для передачи метаданных и для контроля производственного процесса. Более подробный пример, который отображает возможности XMP, представлен на рис. 4.5. На данном рисунке показано, как производитель программного продукта может действовать со своим приложением при существующей системе управления рабочим потоком - WMS: на первом шаге приложение вынимает, к примеру, PDF - файлы из рабочих папок системы управления (WMS) и записывает местоположение для каждого файла в записи XMP. На втором шаге приложение выполняет свою работу и при этом преобразовывает PDF - файлы. Последним шагом является копирование данных обратно в оригинальные папки согласно записям XMP.

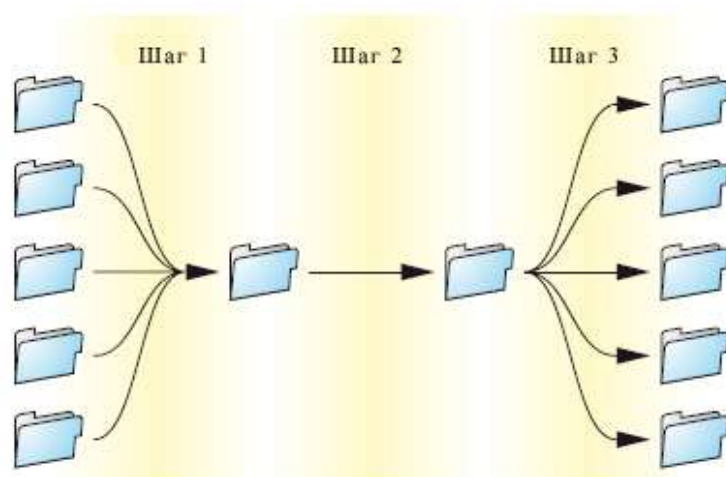


Рис. 4.5 Программное обеспечение сторонних производителей в пределах системы управления рабочим потоком

В итоге можно отметить, что представленные метаданные Exif, IPTC и XMP носят описательный характер, т.е. они описывают свойства цифровых ресурсов. Их основные свойства являются весьма ограниченными. Тем не менее, существуют возможности расширения, по крайней мере, для XMP. Этот вид метаданных не может описывать процессы, необходимые для изменения данных. Также метаданные можно использовать при производстве программного обеспечения для принятия решения о выполнении или

разработке альтернативы. Поскольку метаданные в основном интегрируются в файл вместе с непосредственными данными о ресурсах, они естественно ограничены в области производства, в которой используются цифровые документы (только на допечатном этапе). Оба формата метаданных, которые мы обсудим в следующих разделах, полностью отличаются друг от друга.

4.2 Print Production Format (PPF) – формат печатной продукции

Ранее мы ознакомились с двумя наиважнейшими целями применения PPF: выдача данных из предварительного этапа при расчетах, необходимых для предварительной настройки красочного аппарата по зонам для офсетных печатных машин, а также для расчета программ резки, высечки и фальцовки. В параграфе (2.6) («Расширения») уже упоминалось, что PPF кодируется в PostScript, и там же приводился очень краткий пример.

На рис. 4.6 показано, что при описании печатных листов, PPF - информация по своей логической структуре выглядит в форме дерева.

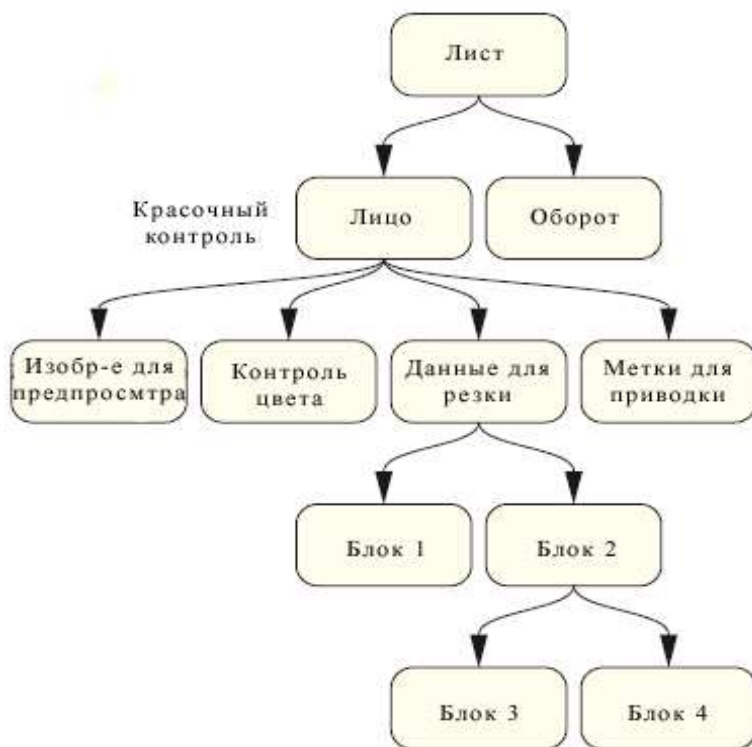


Рис.4.6 Структура PPF - данных

Рис. 4.7 объясняет, почему речь идет о древовидной структуре, если трансформировать наше описание печатных листов в дерево. Он изображается *корневым элементом (root element)*, все разветвления представляют собой ветки, а все информационные группы, которые не имеют дальнейшего продолжения (разветвления), являются листочками.



Рис. 4.7 Древоподобная структура

Обычно вместо «веток» и «листочков» чаще говорится «узел». Каждый узел, до корневого элемента, является ребенком родительского узла; все узлы до листочков также являются родительскими узлами. Так в древоподобной структуре каждый узел имеет одного «родителя», поэтому чаще используется слово «родитель» вместо «родители».

Важным свойством древоподобной структуры PPF (и также многих других древоподобных структур) является наследственность. Дети наследуют все свойства от родителя, однако они могут наделяться новыми значениями. Кроме того, они могут также иметь новые свойства. Например: узлы «лицо» и «оборот» являются детьми «листа». Свойства «листа», такие как размер (*CIP3AdmPaperExtent*) и плотность (*CIP3AdmPaperGrammage*), наследуются от него. Возможно, они также унаследуют характеристику типа растра (*CIP3AdmTypeOfScreen*), но узлы «лицо» и «оборот» могут перезаписать тип растра. В конце концов, также возможно, что тип растра вообще не будет определяться «листочком», а будет определяться только обоими детьми.

Конечно, возможно описание намного большего количества печатных листов в пределах PPF, тогда данные листы являются детьми другого корневого элемента, который называется *Каталогом PPF (PPF Directory)*.

Древоподобная структура реализуется путем охвата PS - кода. С помощью такого способа на рис. 4.8 изображена запись древоподобной структуры, которая была представлена на рис. 4.6.


```
CIP3BeginSheet
  CIP3BeginFront
    CIP3BeginPreviewImage ...
    CIP3EndPreviewImage
    CIP3BeginColorControl ...
    CIP3EndColorControl
    CIP3BeginCutData ...
      CIP3BeginCutBlock ...
      CIP3EndCutBlock
      CIP3BeginCutBlock ...
      CIP3EndCutBlock
      CIP3BeginCutBlock ...
      CIP3EndCutBlock
      CIP3BeginCutBlock ...
      CIP3EndCutBlock
    CIP3EndCutData
    CIP3BeginRegisterMarks ...
    CIP3EndRegisterMarks
  CIP3EndFront
  CIP3BeginBack
  CIP3EndBack
CIP3EndSheet
```

Рис. 4.8 Пример PPF - записей

На рис. 4.9 можно увидеть более подробное извлечение из кода PPF - файла. Первая и последняя строки начинаются со знака комментария в PostScript. Это говорит о том, что данные строки отображаются на PS - интерпретаторе. Последующие строки в большей или меньшей мере описывают себя сами. Формат бумаги был указан в пунктах DTP , ширина - $1984,251968 \cdot 2,54/72 = 70$ см, а высота - $1984,251968 \cdot 2,54/72 = 70$ см. Плотность бумаги как обычно указывается в г/м², а цвет бумаги в CIE-L*a*b*.

```
%!PS-Adobe-3.0
...
CIP3BeginSheet
  /CIP3AdmArtist (Carl Cool) def
  /CIP3AdmJobCode (8) def
  /CIP3AdmJobName (8 Zustaende) def
  /CIP3AdmPaperExtent [ 1984.251968 1417.32283 ] def
  /CIP3AdmSheetName (FB 002) def
  /CIP3AdmTypeOfScreen (amplitude modulated) def
  /CIP3AdmPaperGrammage 100.0 def
  /CIP3AdmPaperThickness 0.035 mm def
  /CIP3AdmPaperColor [ 93.0 0.0 -3.0 ] def
  /CIP3AdmCreationTime (Mon Dec 11 18:29:57 2006) def
  ...
CIP3EndSheet
%%CIP3EndOfFile
```

Рис. 4.9 Детали записей PPF - файла

Самой известной областью применения PPF является предварительная настройка красочного аппарата по зонам для офсетной печатной машины. PPF, на самом деле, играет относительно скромную роль, так как этот формат только сохраняет изображения *предварительного просмотра (Preview)* монтажных листов и некоторую другую информацию, которая представляет собой специальный интерфейс - программное обеспечение, необходимое для расчета значений зон (рис. 1.3). Значения заданных зон также не являются элементами стандартов PPF. На рис.4.10 представлена часть такого изображения для предварительного просмотра, как PPF - структура. Данные об изображении сжаты по длине и закодированы в ASCII85. *Ширина (CIP3PreviewImageWidth)* и *высота (CIP3Preview ImageHeight)* имеют значения в пикселях, а *разрешение (CIP3Preview ImageResolution)* указывается в пикселях на дюйм (ppi). Данное разрешение равняется 50,8 ppi. *CIP3PreviewImageMatrix (матрица)* указывает на последовательность описания пикселей в изображении, которое в данном случае определяется слева направо и сверху вниз. Определение матричного расчета описано в спецификации PS и PDF (также см. задание для самостоятельной работы в конце 9 главы).

```

CIP3BeginPreviewImage
%%Page: 1
%%PlateColor: Cyan
CIP3BeginSeparation
  /CIP3PreviewImageWidth 1490 def
  /CIP3PreviewImageHeight 1210 def
  /CIP3PreviewImageBitsPerComp 8 def
  /CIP3PreviewImageComponents 1 def
  /CIP3PreviewImageMatrix [1490 0 0 -1210 0 1210] def
  /CIP3PreviewImageResolution [ 50.800 50.800 ] def
  /CIP3PreviewImageEncoding /Binary def
  /CIP3PreviewImageCompression /RunLengthDecode def
  /CIP3PreviewImageDataSize 515348 def
CIP3PreviewImage...die Bilddaten
CIP3EndPreviewImage
CIP3EndSeparation
  analog für die Separationen Magenta, Yellow und Black
CIP3EndPreviewImage

```

Рис. 4.10 Определение изображения предварительного просмотра в PPF

Указание только лишь данных изображения предварительного просмотра в программном обеспечении не является достаточным для вычислений, необходимых для

предварительной настройки красочного аппарата по зонам. Причиной этому является то, что расчеты для изображения предварительного просмотра для монтажного листа обычно происходят после преобразования тоновой градации (цветопередачи). И, наконец, на самом деле существуют такие тоновые градации, для которых являются важными расчеты величины для предварительной настройки красочного аппарата по зонам. Так кто или что изменяет тоновые градации? Для производства офсетных печатных форм изображение, которое действительно имеет насыщенность цвета, необходимо растривать (на английском языке - *Screening*). Но при растривании обычно выполняется еще одно изменение (модификация) тоновой градации. Изменение (модификация) тоновой градации должно, к примеру, подогнать тоновые градации к стандартному растискиванию в печати. На языке PS и PDF эти кривые называются *трансферными кривыми (Transferkurven)*, зависящие от кривых линейаризации, кривых калибровки процесса, кривых компенсации тоновой градации или, во флексопечати - от кривых растискивания. На рис. 4.11 представлены две трансферные кривые и их описание в PPF. Всегда имеется набор координат (а именно x и $T(x)$), которые при интерполяции, определяют реальное значение функции. Линия А (красная) является кривой передачи, которая не выполняет никаких изменений тоновой градации, а кривая В (голубая) уменьшает тоновые градации в средних тонах.

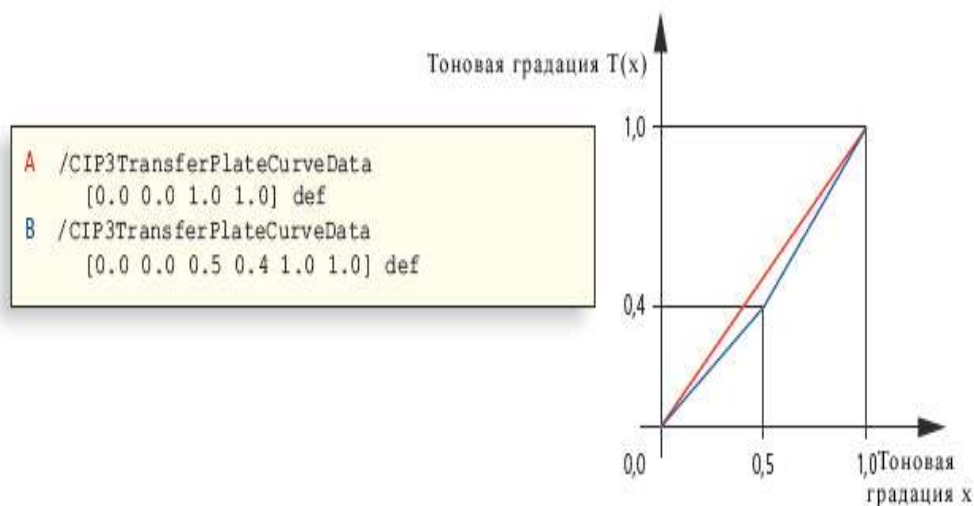


Рис.4.11 Трансферные кривые в записях PPF

Изображение для предварительного просмотра и трансферная кривая могут представляться с помощью RIP, в свою очередь метки резки, приводки, фальцовки и цветные метки не задаются, так как это является бессмысленным. Смысл данных меток знает только программное обеспечение монтажа (Montage Software) и оно может компоновать соответствующую информацию. В PPF-файле указывается позиция цветных меток, меток для приводки и резки на листе. Вместо меток для резки в PPF-файле также могут записываться форматы для резки. В данном случае не определяется никакая последовательность резки, а только указываются определенные размеры, по которым она должна осуществляться (рис. 4.12). Данные о фальцовке могут наоборот содержать информацию о последовательности фальцовки.

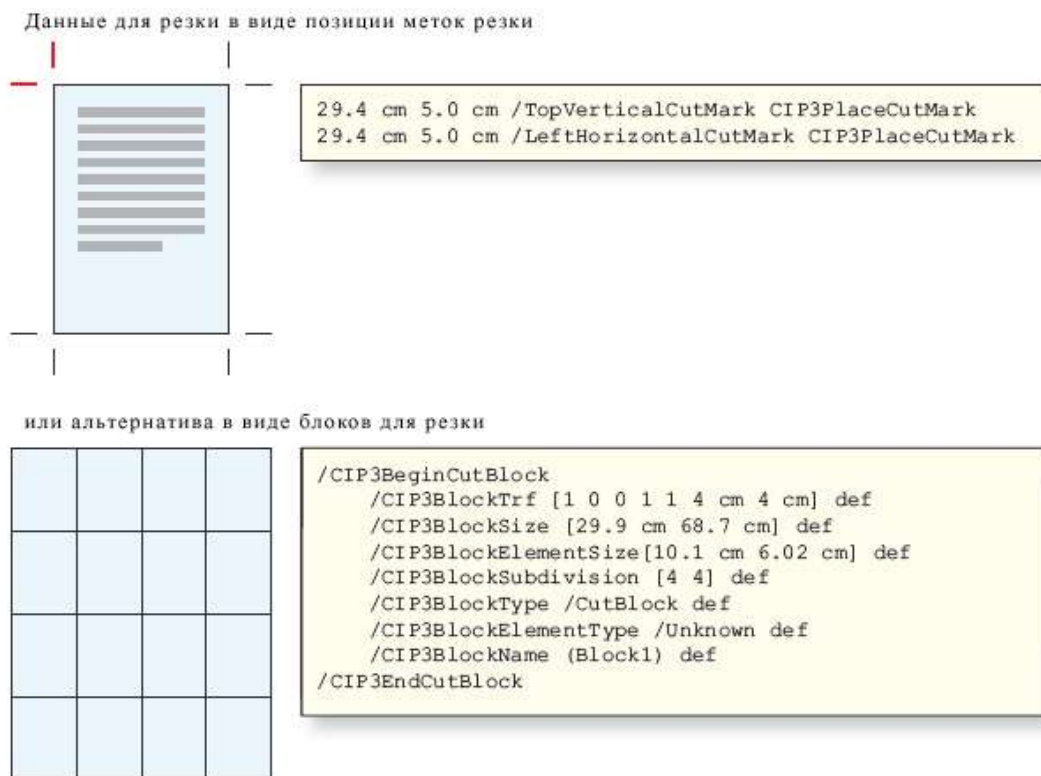


Рис. 4.12 Альтернативные возможности предоставления данных для резки в PPF

Теперь мы имеем все PPF - характеристики для печатного листа. На этом, по сути, версия 2.1 ограничивает действия в области обработки формата. Версия 3.0, которая

вышла в 1998 году и представляет собой последнюю версию этого Jobticket-формата, имеет более широкие возможности определения продукта. Имеется возможность даже описывать в PPF - файле большое количество субпродуктов, которые предшествуют конечному продукту. В целом мы имеем дело с внедрением «продукт-операции» (/CIP3ProductOperation) и «параметр-продукта» (/CIP3ProductParams) в модели процесс-ресурс, которая была рассмотрена в главе 3. Процесс производства, который использует скрепление скобами, шитье или осуществление трехсторонней обрезки, также был определен в версии 3.0.

В приведенных примерах мы ознакомились с двумя PPF - производителями данных для последующих действий, а именно с RIP и программным обеспечением монтажа (Montage Software), а также с тремя PPF - потребителями, а именно с интерфейсными модулями PPF для печатной машины, машины для фальцовки и для резальной машины. Структурная схема данного процесса представлена теоретически на рис. 4.13.

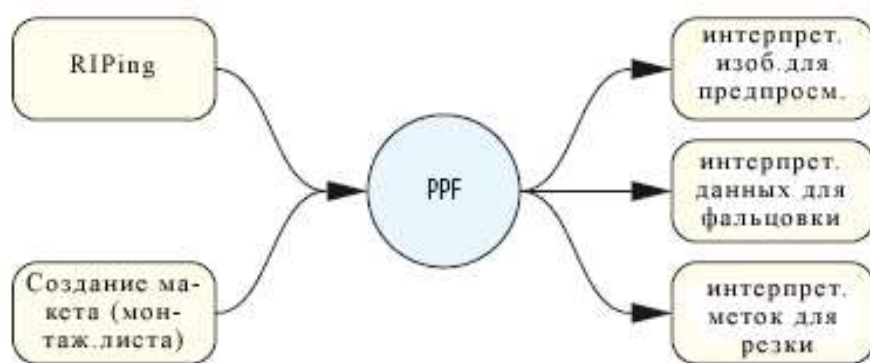


Рис. 4.13 PPF - производитель и PPF - потребитель

К сожалению, на практике создание PPF - файлов не является простым для различных PPF - разработчиков систем. Спецификация PPF, как можно было заметить, не задает никаких строгих требований и это зависит только от отдельного разработчика. Таким образом, на практике при реализации рабочего потока с множеством PPF-файлов необходимо учитывать, что существует множество PPF -разработчиков. Ситуация становится немного запутанной, учитывая то, что PPF -потребитель в различных ситуациях также нуждается в информации от различных PPF - разработчиков систем. Одним из возможных примеров является система PPF – Печатная машина – Интерфейс,

которая, с одной стороны, получает изображения для предварительного просмотра из RIP, а, с другой стороны, метки для приводки – из программного обеспечения для монтажа (Montage Software). Таким образом, от PPF - систем рабочего потока можно ждать возрастающей функциональности в комплексе со сложностью: необходимо создание специальных папок этих разработок и проведение их контроля в случае ошибок. Архивирование же многочисленных PPF - файлов по отдельным решениям является громоздким.

В общем, это не является удивительным, PPF - рабочий поток часто ограничивается только передачей RIP - данных для предварительной настройки красочного аппарата по зонам, т.к. выгода (сокращение времени настройки, макулатурные отходы и др.) по отношению к расходам очень высока.

4.3 Portable Jobticket Format (PJTF)

Portable Jobticket Format, разработанный Adobe Systems Inc [6], является предшественником JDF. Этот формат тесно связан с *Extreme-RIP-архитектурой*, которая, также как и PDF, была создана фирмой Adobe.

Идея *Extreme-RIP-архитектуры* заключается в том, что в RIP соответствующие функции определяются как независимые модули, которые поставляются через PJTF в комплекте с необходимыми метаданными. Модули обрабатывают Jobtickets (рабочая карточка) и поэтому называются «Jobticket-процессоры» (JTP). Примерами использования JTP, с одной стороны, являются классические задачи RIP, такие как *интерпретация (Interpretierung)*, *рендеринг (Rendering)* и *скрининг (Screening)*, или также расширенные функции, такие как нормализация, преобразование цветового пространства, треппинг, спуск полос, вычисление данных для контроля или создание PPF - файла для предварительной настройки красочного аппарата по зонам. JTPs - программы могут работать на отдельном сервере, а также на различных серверах в распределенной системе. Часто бывает, что программа JTP - треппинга распределяется на несколько серверов, так как это очень объемный процесс, требующий большой емкости рабочей памяти процессора. Использование нескольких серверов необходимо для того, чтобы увеличить общую производительность системы. *Extreme-RIP* - архитектура на протяжении многих лет являлась основой для систем управления рабочими потоками в области CtP - устройств и с 2006 года постепенно стала заменяться новой технологией Adobe – *Print Engine (программа двигатель печати)*.

Модуль JTPs контролируется модулем-координатором, который, с одной стороны, получает рабочие карточки (Jobtickets) извне, а, с другой стороны, передает информацию

дальше к следующему модулю JTPs. Как правило, описание монтажных листов (макетов), а также определенные настройки для различных модулей JTP клиентского программного обеспечения передаются через программу - координатор. Поставщик Extreme - RIP - систем может интегрировать несколько JTP - модулей в общую систему или также воспользоваться системами фирмы Adobe. В любом случае координатор должен быть лицензирован фирмой Adobe. Пользователь не может интегрировать никаких других внешних JTP программ (например, других производителей) в существующую систему, так как они не открываются интерфейсом. Установленными в систему могут быть только PJTF -данные (рис.4.14).



Рис. 4.14 Архитектура Extreme - RIPs

Элементами, которые могут быть описаны в PJTF, являются:

- данные для управления заказами (адрес для доставки, срок печати...);
- описание монтажного листа (макета);
- установка значений для процессов рабочего потока (предпечатная проверка, треппинг, растривание...);
- установка значений для процессов обработки печатной продукции.

Основное внимание уделяется описанию макетов и заданий StP в рабочем потоке; другими областями в спецификации чаще пренебрегают.

В предыдущих разделах уже говорилось о том, что XMP - формат кодируется в XML, а PPF в PostScript. PJTF базируется на PDF, а это значит, что PJTF - файл имеет структуру PDF - файла. Специфика PJTF заключается в использовании его ключевых слов. Правда, эта специфика не известна программе Acrobat (фирмы Adobe), поэтому их

значения не могут быть отражены. PDF - файл представляет собой набор пронумерованных объектов и таблицу *перекрестных ссылок* (*Crossreference* - *Tabelle*), в которой точно указано, где в PDF - файле находится определенный объект. На рис. 4.15 показан PJTF - объект под номером 13.

```
13 0 obj
<<
  /BCL 0.95
  /IIO true
  /TW 0.2
  ...
>>
endobj
```

Рис. 4.15 PJTF - объект

Данный объект содержит таблицу, схожую PPF PostScript - кодировкой, в которой каждая строка всегда имеет идентичную структуру: слева ряд символов, а справа – значения данного ряда символов. Эта структура может трактоваться как ключевое слово (или имя переменной) и значение, или также как запись в словаре (что довольно странно). Фирма Adobe действительно называет данную структуру *Словарем* (*Dictionary*). Каждый словарь должен выделяться дополнительными знаками (« и »). Это относится и к PostScript и к PDF.

Объект 13 на рис. 4.15 определяет несколько деталей для треппинга; ключевые слова определяются в PJTF-спецификации. Записано /BLC, означающее *Black Color Limit* (лимит черного цвета), а его значение равняется 0,95. Это говорит о том, что каждый тон черного цвета должен быть заполнен выше, чем 95%. Ключ/IIO и значение *true* означает, что изображение к другим объектам должно быть заполнено или переполнено (*ImageToObjectTrapping*). Зона перекрытия (*TrapWidth*) обозначается как /TW и равняется 1/72 дюйма или один пункт.

Формат PJTF, так же как и XMP и PPF, строится в виде древовидной структуры. Ссылка между узлами реализуются через ссылку от одного объекта к другому. Ссылка определяется как «R», перед которой стоит номер объекта, получающий ссылку. Теперь мы хотим подробно рассмотреть ветвь, которая описывает монтажный лист (макет). На рис. 4.16 каждый прямоугольник символизирует пример объекта PJTF, а каждая стрелка –

ссылку на объект. Идентификаторы объектов расположены над прямоугольниками и выделены красным цветом.

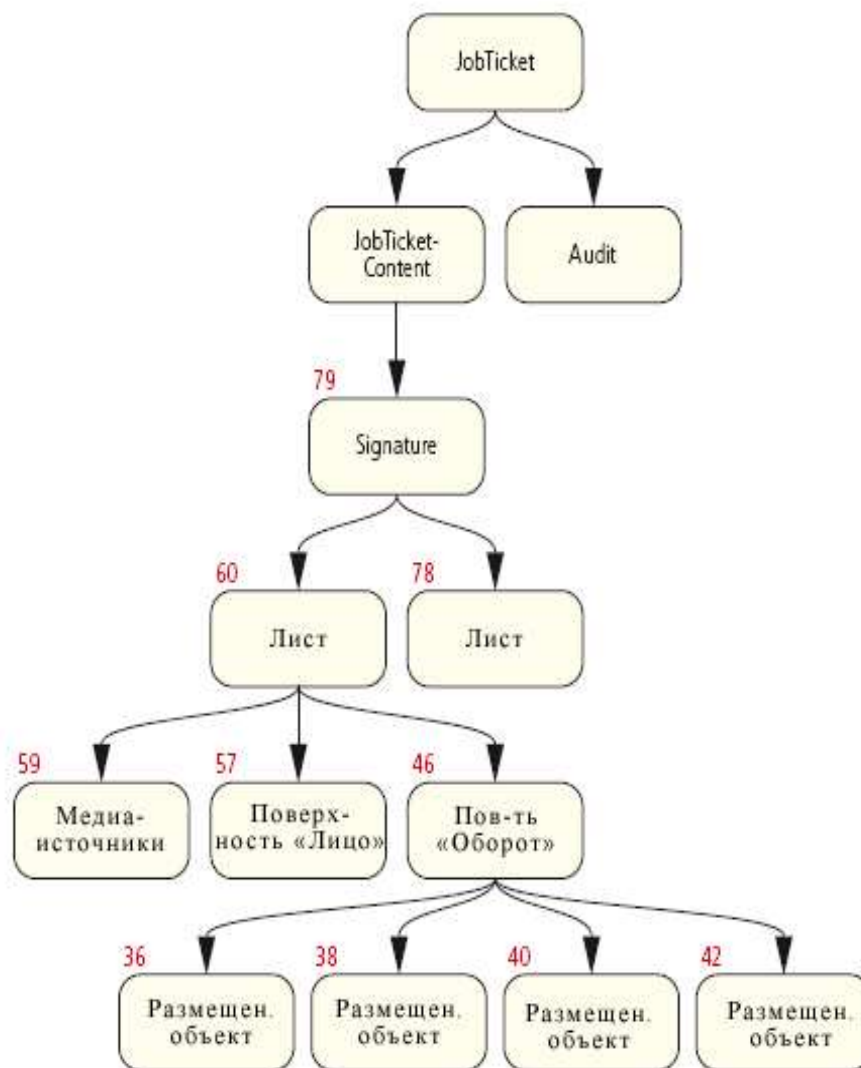


Рис. 4.16 Структура PJTF - документа

Корневым элементом в PJTF является объект с именем *JobTicket* (рабочая карточка). Данный объект ссылается на два других объекта – *содержание рабочей карточки* (*JobTicketContent*) и *проверка* (*Audit*). В объекте *проверка* регистрируются все изменения в PJTF - файле, которые выполняются в процессе производства. Объект *содержание рабочей карточки* ссылается на другой объект – *сигнатура* (*Signature*), в котором листы могут объединяться со схемой спуска полос. На рисунке не представлены другие ссылки объекта *JobTicketContent*, например, на объекты, которые указывают в системе файлов расположение всех PDF - файлов, в которых расположены метки для монтажного листа (макета). Объект *Signature 79* состоит из двух объектов – *лист* (*Sheet*) 60 и 78. Это говорит

о наличии двух страниц. Объект – *лист* под номером 60 по очереди ссылается на два объекта типа *поверхность* (*Surface*), а именно *лицо* (*Front*) и *оборот* (*Back*), которые представлены под номерами 57 и 46. Кроме того, имеется ссылка на объект *Медиа-источники* (*Media-Source*) под номером 59, в котором находится информация о размере листа, цвете листа и т.д. При этом дерево не отображено полностью. Объект 46 указывает на объекты типа *размещенный объект* (*PlacedObject*), которые содержат информацию (размер, позиция, обрезка) о заполнителях страниц. Фактический код объектов 79, 60, 46 и 38 в немного укороченном виде представлен на рис. 4.17, на котором ссылки выделены графически с помощью стрелок.

```

79 0 obj <<
  /Type /Signatur
  /S{
    60 0 R
    78 0 R
  }
>> endobj
...
60 0 obj <<
  /Type /Sheet
  /CP 1
  /Fr 46 0 R
  /B 57 0 R
  /MS 59 0 R
  ...
>> endobj
...
46 0 obj <<
  /Type /Surface
  /PO{
    38 0 R
    42 0 R
    ...
  }
>> endobj
...
38 0 obj <<
  /Type PlacedObject
  /D 0
  /O 0
  /CTM {0.0...0.0}
  /Cl{127.5...506.4}
  ...
>> endobj

```

Рис. 4.17 Ссылки между объектами

В первой строке каждому объекту определяется идентификатор, а также открывается словарь с помощью знака «. Во второй строке указано, о каком типе объекта идет речь (*Signature*, *Sheet*, *Surface* или *PlacedObject*). Все другие записи в каждом словаре касаются определенного объекта и в данном случае не выполняются. Как пример, указана только запись */CTM{0.0...0.0}* в объекте 38, которая определяет Current Transformation Matrix (текущая матрица преобразования) - указывает размер и позицию заполнителей страниц на листе. Каждый словарь закрывается с помощью знака », а каждый объект закрывается с помощью ключевого слова *endobj*.

Подводя итоги можно сказать, что в будущем JDF, безусловно, полностью заменит PJTF (PJTF будет заменен JDF-ом), однако в настоящее время некоторые системы управления рабочим потоком еще импортируют PJTF-монтажный лист (макет).

Задания для самостоятельной работы:

- откройте цифровую фотографию в Photoshop и заполните информацию о файле;
- поместите фотографию в InDesign;
- экспортируйте InDesign-документ в PDF;
- изучите PDF-файл в Acrobat, уделив внимание XMP - данным. Для этого выберите изображение и воспользуйтесь пунктом контекстного меню – Показать свойства изображения;
- изучите PDF - файл с помощью соответствующего текстового редактора, например, WordPad, MFC. Найдите «xmp» и изучите находящиеся в нем записи.

Глава 5 Краткое введение в XML

Extensible Markup Language (XML) – расширяемый язык разметки – является, скорее всего, не языком, а системой для описания языков, т.е. метаязыком. XML является также стандартом для описания типов документов, которыми обмениваются программы. В качестве примера такого XML - документа можно рассматривать Job Definition Format (формат JDF). Это единственная причина, по которой мы занимаемся сейчас XML. Таким образом, мы не ставим задачу досконально изучить тему XML, нам важно представить терминологию, необходимую для понимания формата JDF. Мы также не будем рассматривать преимущества и недостатки XML относительно других структур (PDF или PostScript), его отличия от *HTML* и соотношение с *SGML*. Несколько более подробное, но все еще сжатое введение дано, например, в [42].

Итак, в параграфе 5.1 остановимся на структуре XML - документа и дадим объяснение понятий элемент, атрибут и значение. В параграфе 5.2 речь пойдет о возможностях оформления записей в файлах XML, которые используются для заполнения именного пространства. Наконец в параграфе 5.3 представлен сXML (*commerce eXtensible Markup Language*) - стандарт электронного обмена данными системы на базе XML. В свою очередь сXML составляет основу для частей JDF-спецификации.

5.1. Структура XML- документа

Обычно XML - документ начинается с так называемой декларацией, прологом перед самим XML - документом. Эта декларация размещается в угловых скобках (< >), перед которыми стоят вопросительные знаки (?). На рис. 5.1 в подобных скобках представлен номер версии XML, а именно 1.0. С февраля 2004 г. выпускается версия 1.1, но версия 1.0 по-прежнему используется. Номер версии записывается с помощью атрибута, который состоит из имени атрибута (в примере – *version* - *версия*) и значения в кавычках („1.0“). Значение связано с именем атрибута знаком равенства. Это называется присваиванием значения. Зачастую говорят об атрибуте и называют только его имя – например, об атрибуте *version*. Отдельные атрибуты отделяются друг от друга пробелом.

```
<?xml version="1.0" encoding="UTF-8" ?>
```

Рис.5.1 Декларация XML - документа.

Атрибут *version* является обязательным, т.е. должен всегда быть в декларации XML - документа, другие атрибуты опциональны. В рассмотренном примере на рис. 5.1 есть опциональный атрибут *кодирование (encoding)*, представляющий информацию о способе кодирования XML - документа. UTF-8 – это международная система кодирования на основе стандарта ISO/IEC-10646 с длиной символа минимум 8 бит. UTF – это аббревиатура для «UCS Transformation Format», а UCS, в свою очередь, для «Universal Multiple-Octet Coded Character Set». Набор символов UCS соотносится с более известным набором символов Юникод.

За декларацией обычно следует элемент XML, корневой элемент (*root element*) структуры данных XML. У формата JDF корневой элемент всегда называется JDF, как будет показано в следующей главе. Сейчас пока не будем затрагивать формат JDF, а остановимся в примере на рис. 5.2 на другом корневом элементе, элементе *контакт (contact)*. Элементы, в некоторой степени, являются кирпичиками структуры XML. У каждого элемента есть имя, например, *контакт (Contact)*, *физическое лицо - персона (Person)* или *канал данных (ComChannel)*. Элементы ограничиваются с обеих сторон начальными и конечными тегами, представленными угловыми скобками. Так, начальный тег элемента *контакт (Contact)* представлен `<Contact >`, *конечный тег* - `</Contact >`. Признаком конечного тега является косая черта `/`. Элементы могут содержать субэлементы, так, элемент *персона (Person)* является субэлементом (еще его называют дочерним) элемента-родителя-*контакт (Contact)*. Три элемента *ComChannel* являются дочерними для элемента *Person* и «сестринскими» друг для друга, также как *персона (Person)*, *организация (Company)* и *адрес (Address)* являются сестринскими элементами. Древоподобная структура примера на рис.5.2 наглядно представлена на рис. 5.3.

```
<Contact>
  <Person FirstName="Carl" FamilyName="Cool" NamePrefix="Herr">
    <ComChannel Locator="03475/101010" ChannelType="Phone"
      ChannelUsage="Private" ChannelTypeDetails="Landline" />
    <ComChannel Locator="03475/101011" ChannelType="Phone"
      ChannelUsage="Business" ChannelTypeDetails="Landline" />
    <ComChannel Locator="carl.cool@frisch.de"
      ChannelType="E-Mail" ChannelUsage="BusinessPrivate" />
  </Person>
  <Company OrganizationName="Frisch GmbH"/>
  <Address City="Eisleben" Street="Am Kaltenbach 3" Country="Deutschland"
    PostalCode="06295" />
  <!--Das ist eine einfache XML-Struktur, die zufälligerweise sogar JDF-Code
    ist -->
</Contact>
```

Рис5.2 XML - элементы

Не все элементы имеют явные начальные и конечные теги. Элемент *организация* (*Company*) не содержит субэлементов и поэтому может быть записан без *конечного тега*. Однако перед последней угловой скобкой должна стоять косая черта /. Элементы, не имеющие субэлементов, называют пустыми элементами. То есть только пустые элементы можно записывать в форме <Имя элемента.../ >. Как показано в примере, пустые элементы непременно должны иметь собственные атрибуты.

Многие элементы, но не все, имеют атрибуты, как показано на примере элемента *Contact*. Какие свойства элемента пишутся в виде атрибута, а какие в виде субэлемента, определяется тем, кто создает спецификацию. Так, в нашем примере можно было бы атрибуты элемента *персона* (*Person*) записать атрибутами элемента *контакт* (*Contact*). Тогда элемент *персона* (*Person*) стал бы лишним, и можно было бы обойтись без него.

При записи атрибутов и имен элементов различают строчные и прописные буквы. Кроме того, не все знаки можно использовать, например, не используются пробелы, но могут использоваться специфические знаки, такие как угловые скобки или апостроф.

В XML - документы может быть внесен комментарий, предназначенный для прочтения человеком, и не обрабатываемый программным обеспечением XML. Комментарий начинается <! - -, а заканчивается - ->, и может занимать несколько строчек.



Рис.5.3 Структура XML - элементов.

5.2 Именное пространство XML

Обмен данными на базе документов XML находит широкое применение в разных областях, начиная от описания молекул в химии, языка описания сайтов XPS компании Microsoft (*XML Paper Specification*), до интерфейса для обмена бизнес-данными (заказы, счета, каталоги продукции) *Business-to-Business* (B2B). Кто и как задает необходимые для

XML - документа структуры, типы элементов и их атрибуты? Подобный «словарь разметки» и правила, как их применять, не являются едиными для всех, каждый может использовать свою модель. Однако, если для обмена данными используются XML - документы разных модулей программного обеспечения, то конечно необходимо единство словаря разметки. Такое единство достигается с помощью декларации «именного пространства» (англ. *name space*) в XML - документе. Конечно, если XML - документ предназначен для работы в пределах одного предприятия или даже для личного использования, то конфликтов, вызванных совпадением названий или атрибутов, не может возникнуть, для него не нужно декларировать именное пространство. Для документов JDF, в основном, декларируются несколько именных пространств: одно для названия и структуры всех элементов и их атрибутов в соответствии со спецификацией CIP4, а другие для внесения личных дополнений, сделанных производителем рабочего потока (Workflow).

Именное пространство XML задается атрибутом *xmlns*. Значение данного атрибута служит лишь идентификатором, универсальным идентификатором ресурса (*Universal Resource Identifier (URI)*). Этот идентификатор, как показано на рис. 5.4, выглядит как адрес в адресной строке Интернета, но обозначает лишь имя именного пространства. За URI может стоять ресурс Интернета, но не обязательно. Иными словами: URI может быть также URL (*Uniform Resource Locator*), но на практике, в большинстве случаев, это не используется. На примере рис. 5.4 в этом легко можно убедиться. Первое именное пространство является URL, а второе – только URI (такого адреса в Интернете не существует).



```
<?xml version="1.0" encoding="UTF-8" ?>
<JDF xmlns="http://www.cip4.org/JDFSchema_1_1"
xmlns:HdM="www.hdm-stuttgart.de.com/schema/HdM"...>
  <ResourcePool...>
    ...
  </ResourcePool >
  <HdM:PrivateElement... />
</JDF>
```

Рис.5.4 Именные пространства в XML - документах

Во второй декларации за именем атрибута *xmlns* расположен отделенный двоеточием префикс *HdM* – это лишь аббревиатура для „Hochschule der Medien“ (Высшая школа медиа в г. Штуттгарте). С помощью подобного префикса можно упорядочить имена элементов

или атрибутов разных именных пространств. Если у адресного пространства нет префикса, то оно задано по умолчанию. В примере на рис.5.4 элемент *ResourcePool* является субэлементом элемента *JDF*. Оба имени относятся к адресному пространству по умолчанию, заданному без префикса (http://www.CIP4.org/JDFSchema_1_1), в то время как элемент *HdM:PrivateElement* связан с адресным пространством префиксом *HdM*.

Структура XML - документа может быть описана с помощью DTD (*Document Type Definition*), или схемы XML, причем схема XML – это более современный вариант и предоставляет больше возможностей, т.к. нам не нужно заботиться о различных мелочах, как в DTD. Но для целей нашей книги это не важно. Схемы XML иногда называются XSD (*XML Schema Definition – определение схемы XML*). Соответственно, файл, содержащий такую схему, называется “*JDF.xsd*” и может быть загружен из веб-страницы консорциума CIP4. Поскольку схема, в свою очередь, описывается с помощью XML и кодируется в виде текста, её можно прочитать и изменить с помощью обычной программы обработки текстового документа или браузера.

В схеме обычно задается:

- какие элементы допускаются в документе;
- какие атрибуты допускаются в элементе;
- какие атрибуты обязательны для элемента, а какие опциональны;
- иерархические отношения между документами (дочерние – родительские);
- тип данных для элементов и атрибутов;
- частотность элементов;
- ссылки между элементами.

XML - документы должны быть *хорошо сформированы* и *достоверны*. Под «хорошей сформированностью» понимается соблюдение общего для XML синтаксиса, а *достоверность* подразумевает соблюдение правил, заданных в схеме (в схемах). В XML - документе должно быть указано, по какой схеме (по каким схемам) он структурирован. Программа, которая может читать XML, называется *Parser*, что можно перевести как «Анализатор». Её задачей является лишь анализ текста XML и отслеживание его сформированности. Если программа проверяет также достоверность XML-документа, то она называется «анализатор достоверности». Преимущество такой программы состоит в том, что XML - документы, составленные не по заданным схемам, сразу распознаются и при необходимости отфильтровываются. Тем самым сокращается риск неправильной

обработки XML - документа. Таким образом, анализатор достоверности для языка XML аналогичен программе предварительной проверки файлов для формата PDF.

5.3 Resource Description Framework RDF и XMP

В предыдущей главе был представлен формат XMP, и отмечалось, что его можно записывать языком разметки XML. Этот язык называют *Resource Description Framework* (RDF) (44), он представляет собой, прежде всего, систему для описания ресурсов Интернета. В основном, речь идет о сохранении метаданных для ресурсов сети Интернет, которые должны обеспечивать эффективный поиск Web - страниц. Кроме того, речь идет также о концепции *семантических сетей* (*Semantischen Web*), когда метаданные могут быть найдены и распознаны не только поисковыми машинами, но и другими программами, так называемыми Вэб-агентами.

Язык RDF не очень важен для формата JDF, однако кратко рассмотрим его, чтобы описать представление данных в документах XMP. Остановимся подробнее на примере рис.5.5, в котором показан объект RDF/XMP в формате PDF: элемент `<x:xmpmeta >` - это корневой элемент, его дочерним элементом является `<rdf:RDF >`, который, в свою очередь, имеет три дочерних элемента `<rdf:Description >`. В каждом элементе `rdf:Description` задано свое именное пространство: в первом, с префиксом *pdf* занесены метаданные файла в формате PDF, во втором, с префиксом *xap* сохранена дата создания (изменения) файла, в третьем, с префиксом *dc* сохранены данные об авторе и заголовке файла. “Dc” в нашем примере означает “Dublin-Core” (см. главу 4).

Несколько строк из примера, очевидно, пока непонятны: элемент `<rdf:Seq>` обозначает упорядоченный список, каждый пункт которого представлен элементом `<rdf:li >`. В нашем примере список состоит только из одной записи. Элемент `<rdf:Alt>` позволяет задавать альтернативные значения. Немного странными могут показаться на первый взгляд атрибуты `rdf:about=""`. RDF, как сообщалось, был создан, чтобы описывать преимущественно ресурсы Интернета. Значение атрибута *about* – это URI соответствующего ресурса в Интернете. В нашем примере осталась пустая строка, поскольку речь идет не об Интернет-ресурсах, а о метаданных файла в формате PDF в локальной системе файлов, в котором даже информация XMP сохранена внутри самого PDF - файла.

```

20 0 obj
...
<:xmpmeta xmlns:="adobe:meta/" x:xmpk="XMP toolkit 2.0-9, framework
1.6">
  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:="http://na.adobe.com/1X/1.0/">
    <rdf:Description rdf:about="" xmlns:pdf="http://na.adobe.com/pdf/1.3/"
    pdf:Trapped="False" pdf:Producer="Acrobat Distiller 7.0 (Windows)"
    pdf:OTS_PDFXConformance="PDF/X-1a:2001"
    pdf:OTS_PDFXVersion="PDF/X-1:2001">
    </rdf:Description>
    <rdf:Description rdf:about="" xmlns:xap="http://na.adobe.com/xap/1.0/"
    xap:CreateDate="2007-08-24T12:00:39+02:00"
    xap:CreatorTool="FScript5.dll Version 5.2"
    xap:ModifyDate="2007-08-24T12:00:39+02:00">
    </rdf:Description>
    <rdf:Description rdf:about=""
    xmlns:dc="http://purl.org/dc/elements/1.1/">
    <dc:creator>
    <rdf:Seq>
    <rdf:li>Carl Cool</rdf:li>
    </rdf:Seq>
    </dc:creator>
    <dc:title>
    <rdf:Alt>
    <rdf:li xml:lang="x-default">
    test.indd
    </rdf:li>
    </rdf:Alt>
    </dc:title>
    </rdf:Description>
  </rdf:RDF>
</:xmpmeta>
...
endobj

```

Рис.5.5 Документ XMP, кодированный с помощью XML.

5.4 cXML (Commerce Extensible Markup Language)

В деловом мире предприниматели постоянно обмениваются каталогами услуг и продукции, запросами предложений, коммерческими предложениями, заказами, подтверждают заказы или счета для оплаты. Все это можно делать не только с помощью традиционных писем или электронной почты, но и на Вэб-порталах, или посредством других систем электронной коммерции. В полиграфическом производстве деловыми партнерами выступают (в самом простом случае) покупатель печатной продукции и типография. Существуют различные способы описания таких деловых контактов формальными средствами. Консорциум CIP4, отвечающий за формат JDF, использует для этих целей язык разметки формата XML *PrintTalk*. Последняя версия – PrintTalk 1.3. Таким образом, если детали печатной продукции описываются в формате JDF, то описание деловых контактов происходит с помощью PrintTalk. Точнее говоря: в транзакцию на языке PrintTalk можно интегрировать описание печатного изделия в формате JDF. Оба языка тесно взаимодействуют, но имеют свои специфические задачи. Это позволяет реализовывать модели электронной коммерции, например, Web2Print. Клиенты могут в

одном браузере составлять спецификацию печатной продукции в определенных вариантах, передавать данные и делать заказы.

В разделе 8.4 подробнее рассмотрим PrintTalk. А поскольку он, в свою очередь, базируется на универсальном языке *Commerce Extensible Markup Language* (сXML). Кратко опишем в этом параграфе основы этого языка.

сXML - документы содержат коммерческие транзакции и имеют структуру, представленную на рис. 5.6. Корневой элемент – всегда элемент сXML, который содержит элемент-заголовок (*Header*) и элемент-запрос (*Request*). Элемент *Header* включает адреса отправителя и получателя.

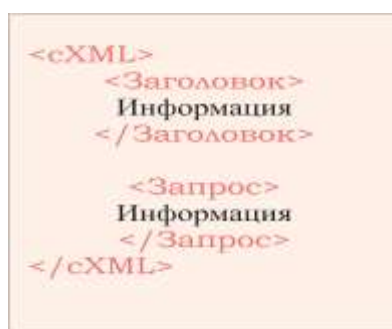


Рис.5.6 Структура сXML-документа.

Помимо этого в нем могут передаваться данные для аутентификации шифрования (= легитимность, идентификация) отправителя – пароль. Каркас заголовка *Header* представлен на рис. 5.7. Элементы *From* (*от кого*) и *To* (*кому*) идентифицируют создателя и получателя сXML - документа. Объекты *From* (*от кого*) и *отправитель* (*Sender*) могут, не должны совпадать. Так как элемент *From* определяет создателя документа, а элемент *Sender* – ту структуру, которая осуществляет на базе языка *http* соединение с получателем. Это могут быть разные структуры, когда сXML - документы проходят через большие сети электронной коммерции. Для элемента *отправитель* (*Sender*) также может для аутентификации вводиться пароль. В рассматриваемом примере элемент *запрос* (*Request*) содержит заказ некоторого товара. Заказ также содержит, прежде всего, общий заголовок, *OrderRequestHeader*, в котором заданы адрес поставки, адрес счета, информация о налоге и об оплате. Элемент *позиции* (*Request*) содержит детальную информацию о заказываемой продукции.

```
<?xml>
<Header>
  <From>
    ...
  </From>
  <To>
    ...
  </To>
  <Sender>
    ...
  </Sender>
</Header>
<Request>
  <OrderRequest>
    <OrderRequestHeader ...>
      ...
      <ShipTo>
        ...
      </ShipTo>
      <BillTo>
        ...
      </BillTo>
      <Tax>
        ...
      </Tax>
      <Payment>
        ...
      </Payment>
    </OrderRequestHeader>
    <ItemOut ...>
      <ItemID>
        ...
      </ItemID>
    </ItemOut>
  </OrderRequest>
</Request>
</xml>
```

Рис. 5.7 cXML - документ

В языке PrintTalk заголовок идентичен заголовку в cXML. Элемент *позиция (Request)* находим и в PrintTalk. Но у этого элемента уже нет дочернего элемента *OrderRequest*, а есть элемент *BusinessObject*. Такой элемент задает одну из названных выше транзакций, напр., запрос, коммерческое предложение и т.д. У некоторых элементов *BusinessObject*, например, у запроса, есть дочерний JDF -элемент, в котором содержится информация о требуемом печатном изделии.

Глава 6. Введение в JDF

В предыдущих главах читатели уже кое-что узнали о формате JDF. Перечислим еще раз основные моменты:

- JDF – это формат рабочей карточки JobTicket, в которой может сохраняться техническая информация и которая также поддерживает функцию управления заказами;
- JDF охватывает функциональность форматов PPF и PJTF;
- JDF основывается на модели процессов и ресурсов;
- JDF базируется на языке XML.

В данной главе подробнее представим в первых шести разделах код формата JDF. Это будет базой для всех последующих глав. В последнем материале попытаемся снова сосредоточиться на последовательности этапов рабочего потока (Workflow). Отдельно остановимся на упомянутой во введении к книге спецификации ICS (*Interoperability Conformance Specification*).

Данная глава должна дать представление об описании продукции и настройках производственного процесса, которые обеспечиваются форматом JDF. К концу изучения материала данной главы читатель должен научиться читать и понимать стандартные документы в формате JDF. Данная глава не может заменить спецификацию JDF (13), которая содержит намного больше деталей.

6.1 Структура JDF-документа

Для каждого задания на печать в системе рабочий поток (Workflow) есть как минимум один документ в формате JDF, который содержит метаданные о задании на печать. Как любой XML- документ, он имеет древовидную структуру. Корневой документ дерева, а также каждая ветвь и каждый листочек называются *узлом JDF (JDF node)*. Обычно задание состоит из нескольких узлов, причем каждый узел описывает его определенную часть. Каждый узел представляет собой элемент в формате XML и имеет, в свою очередь, субэлементы.

Конечно, не всегда есть три уровня иерархии, как это представлено на рис. 6.1. Может быть больше или меньше уровней. На одном уровне может располагаться любое количество JDF - узлов.

Не все JDF- узлы, а только отдельные, изображают процесс. Как правило, это листья дерева, поскольку структура данных в формате JDF основывается лишь отчасти на модели процессов и ресурсов. Имеется четыре различных JDF –узла:

- узел продукта (*product node / product intent node*);
- узел группы процессов (*process group node*);
- комбинированные процессы (*combined processes*);
- узел процесса (*process node*)

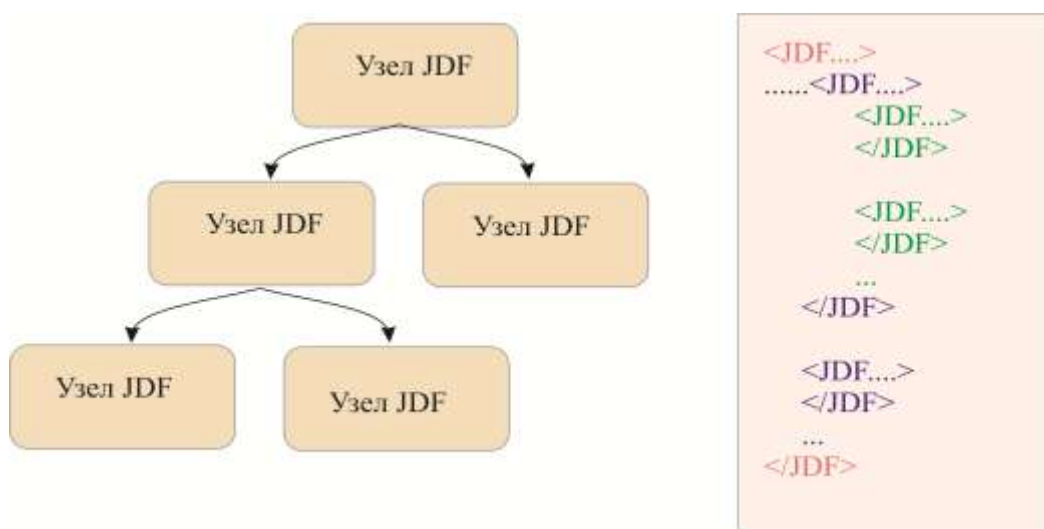


Рис. 6.1 Иерархия JDF-узлов

В узле продукта описывается производимый печатный продукт с его составными частями. Например, брошюра состоит из двух частей - обложки (переплета) и блока (книжного блока). Следовательно, будет три узла продукта: книжное издание, обложка и блок.

Узлы процесса задают отдельные рабочие шаги по изготовлению печатного продукта или одной из его частей. В разделах 3.2 – 3.4 мы познакомились с процессами: *интерпретация (Interpreting)*, *рендеринг (Rendering)*, *скрининг (Screening)*, *резка (Cutting)*, *фальцовка (Folding)*, *сборка (Gathering)*, *брошюровка (Stitching)*, *трехсторонняя обрезка (Trimming)*, *укладка (Stacking)*. Но, конечно, существуют и другие процессы, в следующих главах мы познакомимся с некоторыми из них.

По различным причинам процессы могут комбинироваться и собираться в группы. Этот вид узлов называют узлами группы процессов, или комбинированными процессами.

Например, *RIPing* (рис. 3.8) может быть узлом группы процессов, а также комбинированным процессом, объединяющим три процесса – *интерпретация*, *рендеринг*, *скрининг*. Чтобы показать точнее различия между узлом группы процессов и комбинированным процессом, обратимся к разделу 6.4.

Изготовление печатных форм (PlateMaking) на рис. 3.4 – это следующий пример для узла группы процессов, состоящего из процессов *спуск полос (Imposition)* и *экспонирование (ImageSetting)*. Кроме того, этот узел содержит еще один узел группы процессов, а именно *риппинг (RIPing)*. Таким образом, узел группы процессов может состоять из отдельных процессов, а также из других групп процессов (которые, в свою очередь, содержат процессы и группы процессов). Это напоминает повторную группировку отдельных графических объектов с уже сгруппированными графическими объектами в программе создания графики.



Рис. 6.2 Взаимосвязь JDF - узлов: «узел продукта» (красный), «узел группы процессов» (серый) и «узел процессов» (желтый)

Диаграмма на рис.6.2 показывает взаимосвязь JDF - узлов, причем это не стандартный вариант, а лишь пример. Красным цветом помечены узлы продукта, серым – узлы групп процессов, желтым – узлы процессов. Комбинированного процесса на рисунке нет.

Следует заметить, что не в каждом JDF - документе должны быть все три типа узлов. Бывает, что JDF - документ состоит только из узлов продукта, или только из узлов продукта и узлов группы процессов. Однако JDF - документ полного заказа на печать не может состоять только из узлов процесса, так как процессы всегда относятся к продукту или его частям. Иными словами, корневой элемент в этом случае является JDF - узлом продукта, представляющим все задание целиком.

По мере детализации рабочего потока производства печатного продукта увеличивается соответствующий ему JDF - документ. Если в начале узлов процесса пока нет (потому что процессы еще не определены), то постепенно они добавляются. Новые элементы отчасти генерируются в системе рабочего потока, как показано в разделе 3.2, предварительно заданными по умолчанию значениями, или вносятся оператором.

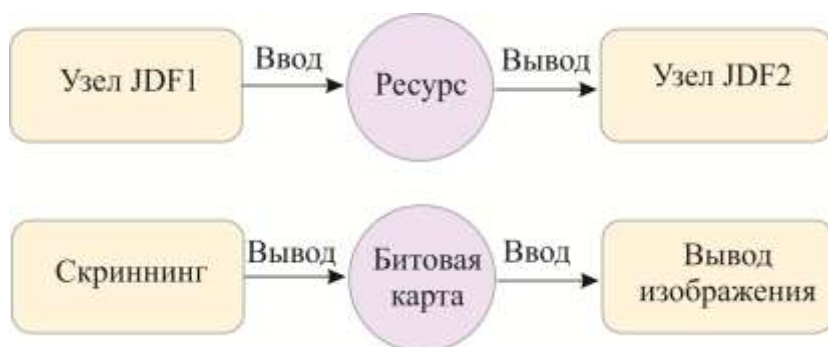


Рис.6.3 Взаимосвязь между JDF –узлами и ресурсами

JDF - узлы существуют не сами по себе, они связаны с ресурсами. Ресурс, как было показано в разделе 3.2, - это *физический объект* (бумага, печатная форма и т.д.), или *электронный контент* (набор параметров, файл и т.д.), который используется (ресурсы *ввода, Input*) или создается (ресурсы *вывода, Output*) определенным узлом. Рис. 6.3 еще раз представляет это положение на очень простом примере. В главе 3 мы видели, что обычно для описания рабочего потока создается более сложная сеть ресурсов и узлов (рис. 3.8). К тому же не только узлы процессов, но и узлы продукта используют ресурсы. Например, они задают детальную информацию о бумаге, необходимой для определенной части продукта. Здесь нужно отвлечься от наглядной идеи о взаимосвязи в производственном процессе производителя и потребителя, которая была очевидна при описании процессов. Можно представить, как, например, в процессе печати изнашиваются печатные формы, но не описать то, как узел продукта, например, обложка, потребляет описание бумаги.



Рис. 6.4 JDF - узел с субэлементами

Как же структурно связаны ресурсы с JDF - узлами? Каждый узел имеет не только атрибуты и их значения, но и субэлементы. Два субэлемента особенно важны, *ResourcePool* (пул ресурсов - набор используемых объектов) и *ResourceLinkPool* (ссылки на пул ресурсов). *ResourcePool* содержит ресурсы для узла. *ResourceLinkPool* указывает, какие ресурсы использовать для узла, являются ли они ресурсами ввода или ресурсами вывода (рис. 6.4). Не каждый ресурс, который используется JDF - узлом, должен находиться в элементе *ResourcePool* данного узла. Могут использоваться также ресурсы, расположенные в древовидной структуре ближе к корневому элементу, и сохраненные в субэлементах *ResourcePool* других узлов. Таким образом, процессы могут делиться ресурсами, их не нужно дублировать (рис. 6.5).

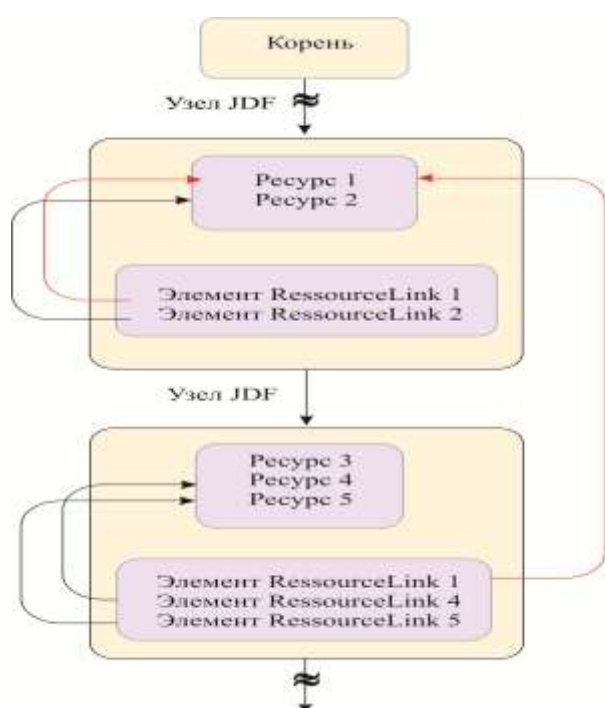


Рис. 6.5 JDF - узлы используют один и тот же ресурс

Итак, не каждый узел должен иметь элемент *пул ресурсов (ResourcePool)*. Но в каждом узле должен быть элемент-ссылка *ResourceLinkPool*, поскольку *ResourceLinks* всегда указывает на узел, где находится *ResourceLinkPool*. Это значит, что если бы у узла не было субэлемента *ResourceLinkPool*, то у него не было бы связи с ресурсами.

Структура JDF - документа с одним корневым элементом и одним JDF - узлом выглядит обычно так, как изображено на рис. 6.6.



Рис. 6.6. Структура JDF - документа с одним корневым элементом и одним JDF -узлом

6.2. Примеры JDF - узлов

Рассмотрим подробнее JDF - документ. Чтобы не перегружать данный раздел, мы остановимся лишь на самых важных элементах и атрибутах.

На рис. 6.7 в первой строчке представлен XML - пролог в том виде, как было описано в предыдущей главе. За ним следует корневой элемент с именем JDF (красный шрифт). Имена атрибутов в примерах всегда записываются зеленым шрифтом.

```

<?xml version="1.0" encoding="UTF-8" ?>
<JDF ID="_4711" DescriptiveName="Frisch-Werbung" JobID="_42"
  Status="InProgress" Type="Product" Version="1.3"
  xmlns="http://www.CIP4.org/JDFSchema_1_1">
  <!-- Generated by the CIP4 C++ open source JDF Library version
  CIP4 JDF Writer Java 1.3 -->
  ...
</JDF>

```

Рис. 6.7 Типичные атрибуты для JDF - корневого элемента.

JDF - узел содержит несколько атрибутов, кратко остановимся на них:

- каждый узел в документе должен иметь однозначный идентификатор - ID;
- значение опционального атрибута *DescriptiveName* – это понятное для человека название узла. В нашем примере узел называется “Frisch Werbung” (реклама компании Фриш), т.к. JDF - документ относится к печатному заданию – рекламной брошюре фирмы Фриш ГмБХ;

- атрибут *JobID* – это название задания или идентификатора, как оно выдается системой управления заказами;

- в атрибуте *статус (Status)* заложено текущее состояние узла. Есть несколько состояний: в нашем примере – *выполняется (InProgress)*, есть также *готово (Ready)*, *остановлено (Stopped)*, *завершено (Completed)*, *прервано (Aborted)* и др.;

- рассматриваемый узел относится к типу *продукт (Product)*, речь идет об узле продукта, а не об узле процесса, узле группы процессов или комбинированном процессе. Узел группы процессов относится к типу *группы процессов (ProcessGroup)*, *комбинированный процесс* – к типу *Combined*. Узел процесса описывает процесс более точно, у него бывают статусы *Interpreting (интерпретация)*, *Cutting (обрезка)*, *Folding (фальцовка)* и т.д.;

- атрибут *версия (Version)* задает версию JDF - спецификации, по которой структурирован узел;

- атрибут *xmlns* определяет адресное пространство XML (см. раздел 5.2).

После атрибутов и значений следует комментарий, который определяет, как и на каком языке программирования (C++) и на базе, какой библиотеки программ (CIP4 JDF Writer Java1.3) был создан документ.

Проследим дальше за процессом создания рекламной брошюры фирмы Фриш ГмБХ. Мы располагаем элементами *ResourcePool* и *ResourceLinkPool* корневого элемента (на рисунке 6.8 представлен лишь фрагмент записей).

```
<ResourcePool>
  <CustomerInfo ID="_4712" CustomerID="0815" Class="Parameter"
  CustomerJobName="Frisch-Werbung" Status="Available" />
  <Contact ContactTypes="Customer">
    <Person FirstName="Carl" FamilyName="Cool" NamePrefix="Herr">
      <ComChannel Locator="03475/101010" ChannelType="Phone"
      ChannelUsage="Private" ChannelTypeDetails="Landline" />
      ...
    </Person>
    <Company OrganizationName="Frisch GmbH" />
    <Address City="Eisleben" Street="Am Kaltenbach, 3"
    Country="Deutschland" PostalCode="06295" />
  </Contact>
  ...
</CustomerInfo>
...
</ResourcePool>
```

Рис.6.8 *ResourcePool* с ресурсом *CUSTOMERINFO*

Первым ресурсом является *Customerinfo*, в котором приведены данные о заказчике. Каждому ресурсу требуется – как и каждому JDF - узлу – однозначный идентификатор ID.

Из атрибута *Class* узнаем, что речь идет о параметрах (*Parameter*). Наконец, статус ресурса (*Status*) – *Available* (доступный). В субэлементе узнаем контактные данные уже знакомого нам из предыдущей главы Карла Кула (*Carl Cool*) (рис. 5.2). Конечно, здесь можно записать и контактную информацию других лиц, например, если не совпадают адрес, на который выписан счет, и адрес поставки.

Элемент *ResourcePool* содержит и другие ресурсы. На рис. 6.9 представлены ресурсы *изделие (Component)* и *поставка (DeliveryIntent)*. Ресурс *Component* является особенно важным и обязательно задается каждому корневому элементу в формате JDF. Этот ресурс представляет конечный продукт, который должен быть изготовлен. В атрибуте *класс (Class)* задано *количество (Quantity)*, и речь идет о материальных ресурсах. Конечные продукты представляют собой счетное множество, в отличие от материальных ресурсов, например, ресурса *краска (Ink)*.

```
<Component ID="_4713" Class="Quantity" ComponentType="FinalProduct"
  DescriptiveName="Frisch-Werbung" Status="Unavailable" />
...
<DeliveryIntent Class="Intent" ID="_4714" Status="Available">
  <DropIntent>
    <DropItemIntent Amount="2000">
      <ComponentRef rRef="_4713" />
    </DropItemIntent>
  </DropIntent>
</DeliveryIntent>
```

Рис. 6.9 Фрагмент записи ресурсов корневого элемента

Ресурс *DeliveryIntent* содержит информацию о том, *как и когда* конечный продукт должен быть поставлен клиенту, *в отдельных случаях также адрес поставки и транспорт для перевозки*. В нашем примере задано не очень много, только то, что будет поставлен некоторый продукт в количестве 2000 экземпляров (внесена только одна позиция *DropItemIntent* - количество *Amount*).

Самая интересная строка – ссылка на *Component* - это ссылка на конечный продукт. Атрибут *rRef* - это ссылка на ресурс, в нашем примере – ресурс 4713, *ID* конечного продукта.

На рис. 6.10 представлен элемент *ResourceLinkPool*, это информация о том, какие ресурсы включены в JDF-узел, идет ли речь о ресурсах ввода (*Input*) или о ресурсах вывода (*Output*).

```

<ResourceLinkPool>
  <DeliveryIntentLink Usage="Input" rRef="_4714" />
  <ComponentLink Amount="2000.0" Usage="Output" rRef="_4713" />
  <ComponentLink Usage="Input" rRef="_4715" />
  <ComponentLink Usage="Input" rRef="_4716" />
</ResourceLinkPool>

```

Рис.6.10 Фрагмент ResourceLinkPool.

Связь между отдельными ссылками на ресурсы (*ResourceLink*) и самими ресурсами очень простая. Если ссылка *ResourceLink* относится к некоторому ресурсу, имеющему название *XYZ* с идентификатором *ID 333*, то для элемента *ResourceLink* выбирается имя *XYZLink* и задается атрибут *rRef = "333"*. В рассматриваемом примере узел содержит ресурс ввода *DeliveryIntent* и ресурс вывода *Component*.

В примере также есть две другие ссылки на ресурсы ввода (ресурсы 44715 и 4716), которые мы пока не рассматривали. Корневой элемент имеет два JDF-узла второго порядка, первый – обложка, второй – блок. Оба продукта, в свою очередь описываются ресурсом компонент (*Component*). Поскольку и обложка, и книжный блок необходимы, чтобы получить конечный продукт, оба ресурса *Components* для конечного продукта будут ресурсами ввода.

JDF - узлы второго порядка имеют собственные элементы *ResourcePool* и *ResourceLinkPool*. В них представлена более подробная информация о составляющих продукта - какая бумага должна использоваться, какая цветность, сколько страниц должно быть в конечном продукте. Ресурс *LayoutIntent* (*компоновка*) задает информацию о формате конечного продукта и о количестве страниц, ресурс *MediaIntent* (*материал*) задает качество бумаги (Рис. 6.11). Если оператор вносит эту информацию в систему управления, то часто некоторые детали опускаются. Например, может оказаться, что нет точных данных о конечном количестве страниц, или вес бумаги задан лишь приблизительно. Поэтому эти ресурсы допускают для ввода определенную длину описания данных. *Intent* по-английски – это «намерение», в ресурсах *намерение (Intent)* задается информация о конечном продукте. Поэтому такие ресурсы бывают только в узлах продукта, по-английски эти узлы называются *Product Intent Nodes (Node-узел)*.

В нашем примере оба ресурса относятся не к конечному продукту, а к его составляющим – обложке и блоку. Конечно, для них используют разную бумагу, краску и т.д., и задавать это все нужно по отдельности.

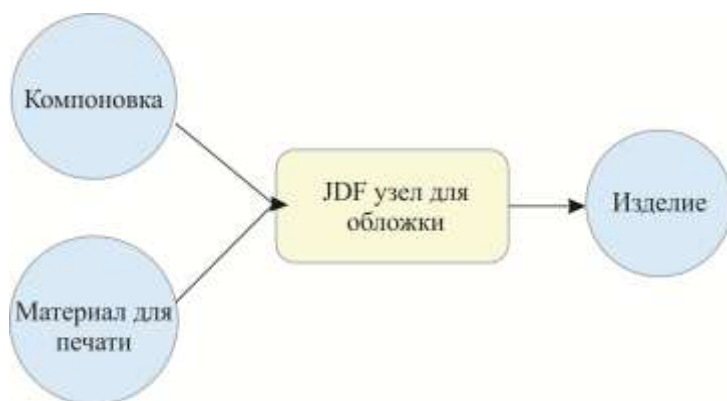


Рис. 6.11 Отдельный JDF - узел «обложка» с собственными элементами ResourceLinks

Рис. 6.12 представляет ресурс *MediaIntent* для обложки. Этот ресурс из класса *Intent*. Обращают на себя внимание субэлементы – размеры (*Dimension*) и вес (*Weight*).

```

<JDF DescriptiveName="Umschlag" ID="_4715" Type="Product" >
  <ResourcePool>
    <MediaIntent ID="_4717" Class="Intent" Status="Available"
      DescriptiveName="Papier fBr Cover">
      <Dimensions Actual="2437,7952795905512 1729,1338582677165"
        DataType="XYPairSpan"
        Preferred="2437,7952795905512 1729,1338582677165" />
      <Weight DataType="NumberSpan" Range="115 - 125" Preferred="120" />
    </MediaIntent>
  </ResourcePool>
</JDF>
  
```

Рис.6.12 Ресурс *MediaIntent* с информацией о материале для печати

Значения атрибутов: *действительный (Actual)* и *желаемый (Preferred)* в элементе *размеры (Dimension)* совпадают. Это размер бумаги, заданный в системе DTP. Если разделить значение на 72 и умножить на 2,54, то можно прийти к привычному виду значений в сантиметрах (в примере – 86x61). В атрибуте *Preferred* содержатся данные о параметрах, указанных клиентом, в атрибуте *Actual* – значения, заданные типографией. В примере они совпадают.

В атрибуте *вес (Weight)* указан вес бумаги, заданный клиентом в граммах на кв. метр. Видно, что приведен диапазон 115-125г., а в поле *Preferred* стоит точное значение 120.

Для системы необходимо, конечно, указать, что в значении «115-125» речь идет об интервале значений. Для этого в типе данных задается *интервал (NumberSpan)*. У элемента *Dimension (размеры)* тип данных – *XYPairSpan*, он может определять интервал

пары чисел: размер бумаги от «ширина 1 x высота 1» до «ширина 2 x высота 2». В примере на рис. 6.12 эта возможность не используется, указан фиксированный размер бумаги.

Помимо элементов *ResourceLinkPool* и *ResourcePool* каждый узел содержит еще одну группу, которую мы пока не рассмотрели, элемент *пул аудита (AuditPool)*. В него вносятся протокольные записи процесса, и после его окончания можно ознакомиться со всеми этапами производства. С одной стороны, записываются модификации самого JDF - документа, с другой стороны – выполнение всех заданных процессов. Сохраняются, например, все действия, произведенные с печатной машиной (замена печатных форм, смывка офсетного цилиндра, и т.д.), если печатная машина фиксирует эти значения. Однако не всегда есть подобный список (*Audit*), который сохраняется в элементе *AuditPool*. При ведении списка в нем отражается:

- создание, изменение или удаление узлов;
- время выполнения процессов (начало, окончание и т.д.);
- статус на момент завершения работы (*Completed (выполнено)*, *Aborted (прервано)*, *Stopped (остановлено)*...);
- ошибки (*Warning (предупреждение)*, *Fatal (сбой)*, *Error (ошибка)*);
- израсходованные или недостающие ресурсы.

С помощью этих записей в *AuditPool* системы может рассчитать, например, окончательную стоимость задания после его выполнения.

На рис. 6.13 представлена запись простого элемента *AuditPool*, в котором есть только информация о том, когда и кем был создан узел. Мы видим:

- *AgentName* – имя программы, создавшей узел;
- *AgentVersion* – версию этой программы;
- *Author* - лицо (пользователь ПК), которое создало узел;
- *TimeStamp* - время генерации узла.

Таким образом, узел был создан 22 октября 2008 г. в 17:09. Число 47 дает значение секунд, а +01:00 – это часовой пояс, т.е. отклонение от среднего времени по Гринвичу.

```
<AuditPool>
  <Created AgentName="SuperJDF" AgentVersion="1.0" Author=" Administrator"
  ID=" 4717" TimeStamp="2008-10-22T17:09:47+01:00" />
</AuditPool>
```

Рис.6.13 Элемент AuditPool

6.3 Разделяемые ресурсы

С некоторыми ресурсами возникает проблема. Например, все печатные формы определенной работы – это один ресурс, или каждая форма представляет собой самостоятельный ресурс? Еще сложнее вопрос с оттисками. С одной стороны, имеет смысл рассматривать все оттиски некоторого продукта (например, тиража) как единое целое. Это необходимо тогда, когда речь идет о спецификации бумаги. С другой стороны, необходимо учитывать выход готовой продукции с учетом количества макулатурных листов.

Если создавать отдельный ресурс для каждого оттиска, то получился бы документ гигантского размера, а его создание было бы по большей части бесполезной работой. Все-таки лучше создавать один ресурс для всех тиражных оттисков, но такой ресурс, который при необходимости можно было бы разделить на несколько частей. Итак, нужны разделяемые ресурсы.

У таких ресурсов, конечно, необходимо указывать, как они структурированы. В последнем примере рассматривался ресурс *изделие (Component)* (см. рис. 6.9). Рассмотрим его до описания разделяемого ресурса, даже если этот пример покажется некорректным (рис.6.14).

```
<Component ID="_4713" Class="Quantity" ComponentType="FinalProduct"
  DescriptiveName="Frisch-Werbung" Status="Unavailable" PartIDKeys="Condition">
  <Component Condition="Good" IsWaste="false" />
  <Component Condition="Waste" IsWaste="true" />
</Component>
```

Рис. 6.14 Разделяемый ресурс

К ресурсу *Component* с идентификатором ID 4713 добавляется атрибут *PartIDKey*. Его значение задает команду на разделение, иными словами, он является ключом к разделению. В нашем примере печатные листы разделяются на две категории: на «хорошие» и «плохие» (как герои голливудских фильмов). Оба дочерних подресурса – субэлементы родительского ресурса *Component*, получают все его атрибуты.

В последнем примере было разделение ресурсов по одному уровню. Но во многих случаях необходимо разделение по нескольким уровням. В качестве примера рассмотрим экспонируемые формные пластины. Совокупность всех печатных форм одного задания на печать, как показано на рис. 6.15, разделяется на 4 уровня. Печатные формы

(*ExposedMedia*) изготавливаются для четырех красок. Каждая из них включает несколько сигнатур(*Signature*). Страница (*Sheets*)- часть оттиска, она имеет - лицевую (*Front*) и - обратную (*Back*) стороны, а каждая страница (*Side*) один или несколько составляющих блоков (текст, графика и др.)..

Рис 6.16 показывает такой разделенный ресурс. В атрибуте *PartIDKey* задан список имен на разделение "*SignatureName SheetName SideName SideSeparation*", в соответствии с неким абстрактным примером, представленным на рис. 6.15. Последовательность записей в списке нельзя изменять, они должны всегда выстраиваться «сверху вниз». При этом действует правило: чем короче цепочка родственных элементов от корневого элемента, тем дальше слева должен элемент располагаться в списке.

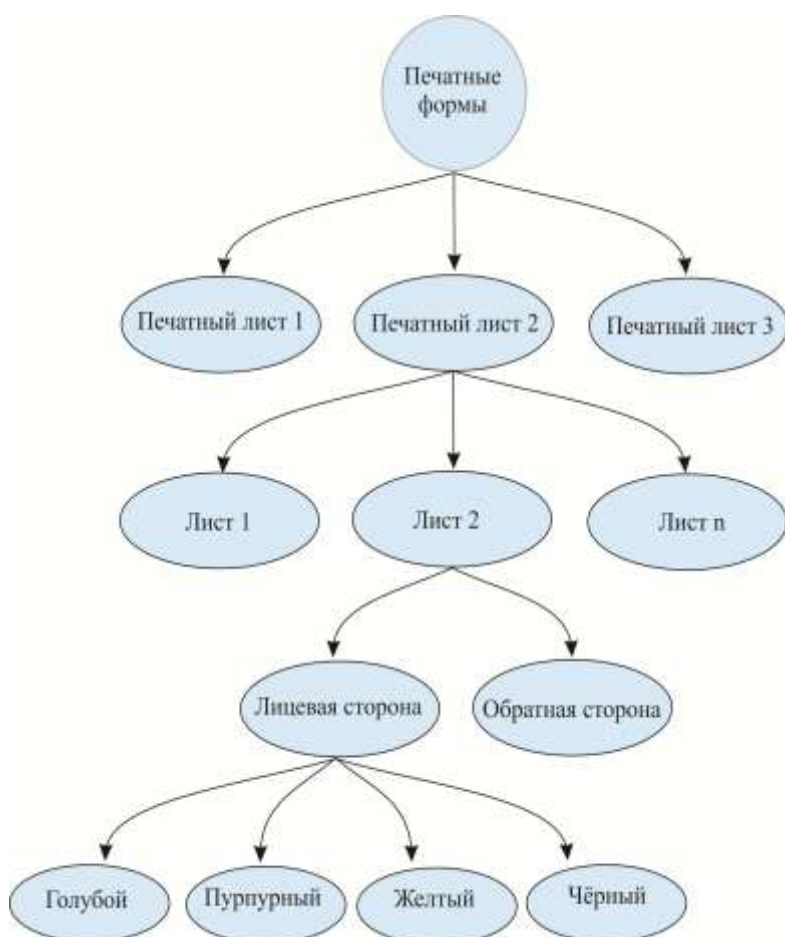


Рис. 6.15 Ресурсы *Component* на примере печатных форм

Такая же иерархия должна соблюдаться и при записи субэлементов разделяемого ресурса.

```
<ExposedMedia Class="Handling" ID="_4719" PartIDKeys="SignatureName.SheetName
Side Separation" Status="Unavailable">
  <ExposedMedia SignatureName="Signature1">
    <ExposedMedia SheetName="Sheet1" Status="Unavailable">
      <ExposedMedia Side="Front">
        <ExposedMedia Separation="Cyan" />
        <ExposedMedia Separation="Magenta" />
        <ExposedMedia Separation="Yellow" />
        <ExposedMedia Separation="Black" />
      </ExposedMedia>
      <ExposedMedia Side="Back">
        <ExposedMedia Separation="Black" />
      </ExposedMedia>
    </ExposedMedia>
  </ExposedMedia>
  ...
</ExposedMedia>
  ...
</ExposedMedia>
```

Рис. 6.16 Разделяемый ресурс

В примере 6.16 лицевая страница 1 задана в цветовом пространстве СМУК, оборотная сторона страницы не запечатывается.

6.4 «Черные ящики» и комбинированные процессы

Есть различные причины, почему желательно объединять несколько процессов в одну новую структуру:

- в начале производства некоторого продукта отдельные процессы пока имеют недостаточную спецификацию. Поэтому удобно рассматривать сначала большие этапы производства в совокупности, не задавая промежуточные результаты процессов внутри этапа. Промежуточные уточнения вносятся в ходе производства;

- если машина или программа последовательно выполняют несколько процессов, и никто не изменяет промежуточные результаты, нет необходимости точно описывать все эти процессы. Это встречается, например, в цифровой печати, где в линейке оборудования выполняются все процессы: интерпретация, рендеринг, скрининг, печать, брошюрование, трехсторонняя обрезка, которые следуют друг за другом. Но и в офсетной печати на допечатном этапе можно объединить два процесса – *создания макета и расчет настроек красочных зон* - в одну комбинацию.

Первый случай приводит нас к понятию «Черный ящик» (*GrayBox*), второй – к *комбинированный процесс (Combined Process)*.

«Черный ящик» – это узел группы процессов, в котором детали еще не имеют спецификации. Они вводятся лишь в процессе производства. Когда добавляются все процессы и ресурсы, «черный ящик» разделяется, т.к. он сам не может управлять отдельными процессами. Обычно система управления создает один или несколько «черных ящиков», т.к. не располагает всеми необходимыми параметрами. Например, система может определить, сколько форм какого формата нужно сделать, но не может задать отдельные этапы их создания. Система обычно не содержит точной информации о треппинге и растрах, т.к. она не является необходимой для произведения расчетов/составления смет и калькуляций. Недостающие установки добавляются из заданных значений по умолчанию в модулях рабочего потока, значений печатного задания или вводятся оператором.

На схеме 6.17 видно, что «черный ящик» (*GrayBox*) может содержать ресурсы ввода (*Input*) и вывода (*Output*). Ресурсы вывода должны иметь спецификацию. Далее будем пользоваться только термином *GrayBox*.

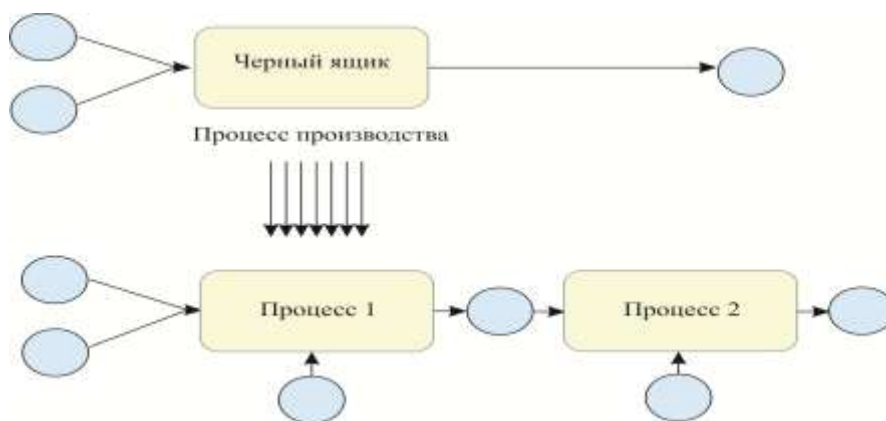


Рис. 6.17 «Черный ящик» (*GrayBox*) в процессе производства разделяется

Рисунок 6.18 представляет пример *GrayBox*, который задает создание офсетной печатной формы (*PlateMaking*). Он имеет 5 (или больше) ресурсов ввода, и как минимум один ресурс вывода. Поскольку в следующих главах нам понадобятся приведенные ниже ресурсы. Кратко остановимся на них:

- ресурс *RunList* (список, последовательность описания страниц) – упорядоченный массив описаний страниц. Здесь приводятся контент-данные, а также маркировочные данные, необходимые для монтажного листа - макета (в примере 6.18 с добавлением “Marken-маркировка”);
- ресурс *Layout* (компоновка) описывает монтажный лист/макет;
- ресурс *ColorantControl* (контроль цвета) задает цветовые параметры;

- ресурс *Media* (материал, носитель информации) содержит характеристику формных пластин (размер, название продукта и т.д.);
- ресурс *ExposedMedia* (экспонированный материал), в нашем случае – печатные формы.

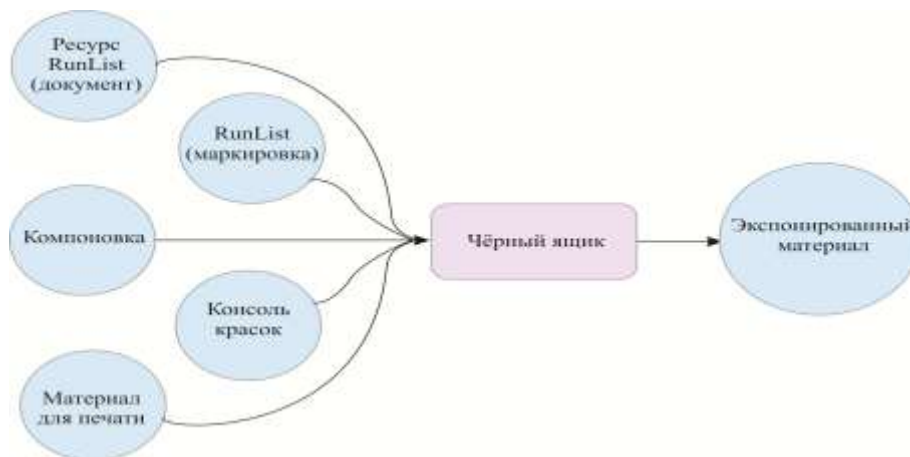


Рис.6.18 «Чёрный ящик» (*GrayBox*) при производстве офсетной печатной формы

Код для создания офсетной печатной формы (ресурс *PlateMaking*) представлен на рис. 6.19. Узлы групп процессов типа *GrayBox* всегда можно узнать по комбинации атрибутов *Type* (*тип*) и *Types* (*типы*). Значение атрибута *Type* – это *ProcessGroup* (группа процессов), значение *Types* – список процессов, включенных в *GrayBox*. Следует подчеркнуть, что элемент *RIPing* не является процессом, а представляет собой узел группы процессов.

```

<JDF Type="ProcessGroup" Types="Imposition RIPing ImageSetting"
  DescriptiveName="GB PlateMaking" ...>
...
  <ResourceLinkPool>
    <RunListLink ProcessUsage="Document" Usage="Input" ... />
    <RunListLink ProcessUsage="Marks" Usage="Input" ... />
    <LayoutLink Usage="Input" ... />
    <ColorantControlLink Usage="Input" ... />
    <MediaLink Usage="Input" ... />
    <ExposedMediaLink Usage="Output" ... />
  </JDF>

```

Рис 6.19 Записи *GrayBox* при производстве офсетной печатной формы

Существуют также узлы групп процессов, которые не выполняют функций *GrayBox*, в них отсутствует атрибут *Types*, а атрибут *Type* имеет значение *ProcessGroup* (*группа процессов*). Такой тип получают процессы, записанные в виде JDF-узлов. Процессы могут

быть не только сгруппированны, но и скомбинированны. Результатом такой комбинации является *комбинированный процесс (Combined Process)*. Он содержит список имен процессов и необходимые ресурсы. На рис. 6.20 можно увидеть, что в таком случае значение атрибута *Type* – „*Combined*“ - *комбинированный*, а в атрибуте *Types* перечислены процессы, как и в GrayBox.

```
<JDF Type="Combined" Types="Imposition Interpreting Rendering"...>
  <ResourcePool>
    <RunList ID="_100"... />
    <Layout ID="_200"... />
    <InterpretingParams ID="_300"... />
    <RenderingParams ID="_400"... />
    ...
  </ResourcePool>
  <ResourceLinkPool>
    <RunListLink CombinedProcessIndex="0" Usage="Input" rRef="_100" />
    <LayoutLink CombinedProcessIndex="0" Usage="Input" rRef="_200" />
    <InterpretingParamsLink CombinedProcessIndex="1" Usage="Input"
      rRef="_300" />
    <RenderingParamsLink CombinedProcessIndex="2" Usage="Input"
      rRef="_400" />
    ...
  </ResourceLinkPool>
</JDF>
```

Рис. 6.20 Пример комбинированного процесса изготовления офсетной печатной формы.

С точки зрения структуры, комбинированный процесс похож на группу процессов. Разница состоит в том, что комбинированный процесс обрабатывается одним устройством, а группа процессов – в основном несколькими устройствами. Поэтому в комбинированном процессе не нужно перечислять ресурсы, которые будут промежуточными результатами процессов.

Новым на рисунке 6.20 является лишь атрибут *индекс комбинированного процесса (CombinedProcessIndex)*. Его значения указывают, к каким заданным в атрибуте *Types* процессам относятся ресурсы. Индексация начинается с нуля, т.е. первый процесс в списке, а именно *спуск полос (Imposition)*, получает индекс 0, второй (*интерпретация*) – индекс 1, а третий (*рендеринг*) – индекс 2. Следовательно, ресурсы ввода *список (RunList)* и *компоновка (Layout)* относятся к процессу *Imposition (спуск полос)*, ресурсы ввода *InterpretingParams* – к процессу *Interpreting (интерпретация)*, а ресурсы ввода *RenderingParams* к процессу *Rendering (рендеринг)*.

6.5 JDF рабочий поток - архитектура

После того как мы рассмотрели, каков принцип построения документа в формате JDF, выясним, как попадают эти документы в модули рабочего потока.

Самый простой способ – последовательная передача информации, как показано на рис.6.21.



Рис. 6.21 Последовательная передача данных

Эта примитивная модель в действительности не встречается, в крайнем случае – только в простейших условиях полиграфического производства, когда информацией обмениваются два или три соседних процесса формата JDF. Например, программа *процесса сборки (Stripping)* создает JDF - документ для спуска полос (*Imposition*).

Более реальная модель – центральный накопитель данных в формате JDF, к которому имеют доступ различные модули рабочего потока, как представлено на рис. 6.22. Таким накопителем может быть база данных или просто отдельная директория на некотором сервере. Конечно, отдельные инструменты рабочего потока не имеют свободного доступа к данным в формате JDF, он осуществляется только под управлением специальной программы более высокого уровня, обеспечивающей согласованность данных. В отличие от ситуации последовательной передачи данных, данные находятся на одном центральном сервере, что предпочтительнее с точки зрения обеспечения их безопасности. На рис. 6.22 показано, что управление информацией в формате JDF является частью программного обеспечения, отвечающего за производственный процесс.



Рис. 6.22 Центральное управление

Существуют также системы управления, где управление данными осуществляется по схеме, представленной на рис. 6.23. Конечно, не всегда для одного заказа создается только один документ в формате JDF. Каждый модуль рабочего потока может делать новые версии JDF - документа. Тогда в случае возможного сбоя (как минимум теоретически) можно вернуться к прежнему варианту.

Еще раз отметим, чтобы избежать возможного непонимания: Конечно, нет такого построения систем, что все данные для создания печатного продукта в формате JDF сохраняются в одном определенном месте. В действительности система может обращаться к своим собственным базам данных со своими таблицами и выражениями, в которых также сохраняются данные, не отображаемые в формате JDF (например, стоимость машин и выполнения рабочих операций). JDF/JMF позволяет реализовать лишь интерфейс. То же самое происходит и в допечатной подготовке, процессе печати и послепечатной обработке.

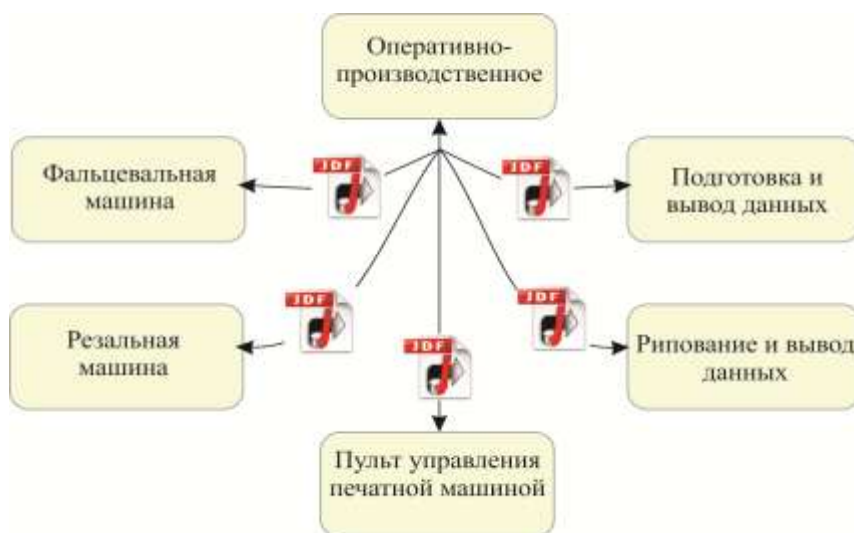


Рис. 6.23 Один из вариантов построения системы рабочего потока на основе формата JDF

Консорциум CIP4 предлагает следующую, скорее иерархическую модель построения рабочего потока, пример которой изображен на рис. 6.24. В этой модели речь идет скорее о функциях, которые могут выполнять программные средства в формате JDF. Здесь действуют следующие правила языкового оформления:

-*агент (Agent)* может генерировать документ в формате JDF, создавать и модифицировать узлы. Обычно эту функцию выполняет система управления заданиями;

-*контроллер (Controller)* должен передавать на оборудование JDF-документы и JMF-сообщения. И наоборот, он должен обеспечивать обратную связь, передавать сообщения от оборудования к агенту;

-*устройство (Device)* интерпретирует JDF - узлы и инициирует соответствующие процессы в подключенном оборудовании. Коммуникация между *устройством (Device)* и оборудованием осуществляется не на базе JDF/JMF, а задается производителями оборудования. На «компьютерном» языке *Device* можно было бы назвать драйвером внешнего устройства;

-*машина (Machine)* выполняет один или несколько процессов, не понимая формат JDF/JMF и не создавая никаких документов.

Такое разделение не всегда четко прослеживается. Так, многие *контроллеры* и *устройства* могут создавать и модифицировать узлы в формате JDF, т.е. выполнять функции агента. *Агент* зачастую может также выполнять функции *контроллера*, как это часто бывает в системах управления заданиями. Тогда описанная структура становится трехступенчатой, агент (система управления) одновременно является *универсальным*

контроллером. Разделение на три контроллера на рис. 6.24 также является примером. В действительности может быть больше или меньше контроллеров. И, наконец, контроллеры могут работать рекурсивно, один контроллер может обслуживать одного или нескольких контроллеров следующей ступени.

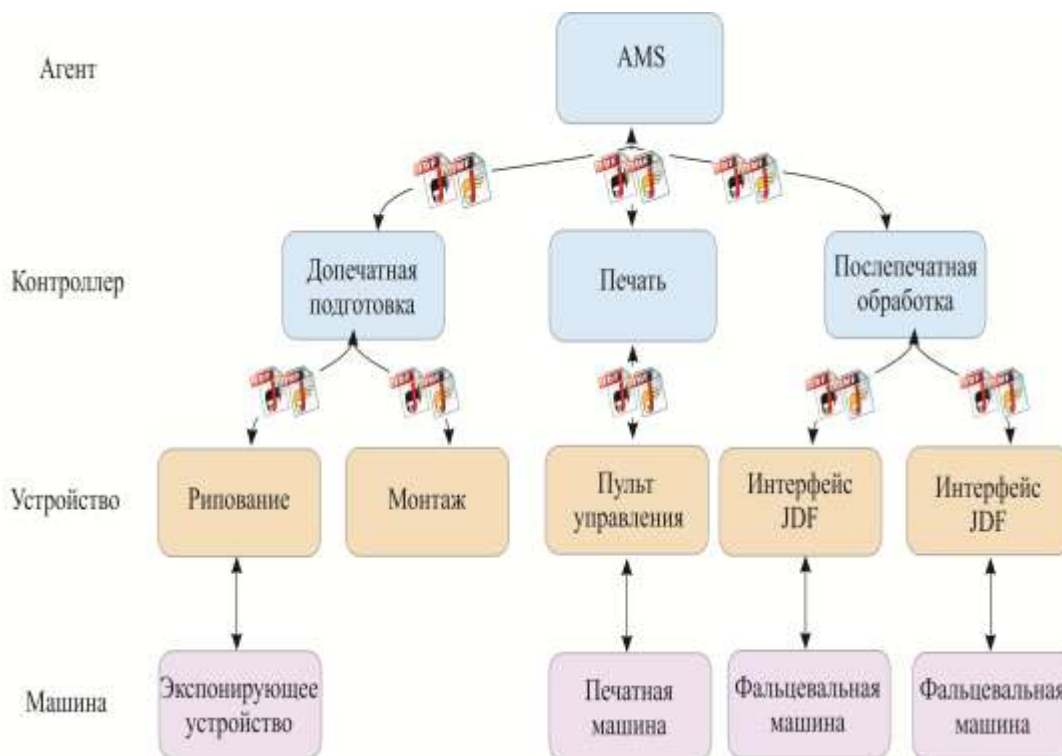


Рис. 6.24 Иерархическая структура JDF/JMF - коммуникации

Контроллер передает данные следующими способами:

- всё задание последовательно всем устройствам;
- *контроллер* выясняет через JMF, какое устройство может обрабатывать определенный процесс и отправляет ему соответствующие части заданий;
- как и в предыдущем пункте, *контроллер* передает устройствам определенные части задания, но параметры устройств в контроллере четко заданы.

В последних двух пунктах должны быть выделены конкретные части узла JDF. Как это делается, рассмотрим в следующем разделе.

Комбинации устройств и машин в системе рабочего потока рассматривались в предыдущих параграфах, отмечалась роль рабочей карточки - процессора (JobTicket-Prozessorent) или инструмента – двигателя рабочего потока (Workflow-Engines).

Представленные далее модели могут сначала показаться очень сложными и достаточно статичными, и возникает вопрос, как процессы могут выполняться автоматически, чтобы применение формата JDF была выгодным на производстве. В действительности программные средства в формате JDF могут автоматически запускать процессы и оборудование. Для этого должны соблюдаться следующие условия:

- оборудование должно соответствовать по техническим данным тому, чтобы выполнять процесс, заданный в формате JDF;
- статус процесса – *Ready (готов)* или *Waiting (ожидание)*;
- все ресурсы ввода должны иметь статус *Available (доступны)*;
- реальное время должно соответствовать предусмотренному программой временному интервалу.

Предусмотренный программой *временной интервал* для определенного узла процесса содержится в отдельном ресурсе с именем *NodeInfo*. Обычно в этом ресурсе задаются данные о выполнении, ответственность за процесс и последствия превышения заданного времени. В примере 6.25 процесс должен начаться не раньше 1.8.2008, в 0:00, и не позднее 2.8.2008, 8:00. Процесс должен быть завершен в тот же день, в 15:00.

```
<ResourcePool>
  <NodeInfo FirstStart="2008-08-01T00:00:00+01:00"
    LastStart="2008-08-02T08:08:00+01:00"
    LastEnd="2008-08-02T15:00:00+01:00"... />
  ...
</ResourcePool>
```

Рис. 6.25 Ресурс *NodeInfo*

6.6 Разделение и объединение

В системе управления рабочего потока некоторые операции могут протекать параллельно. Соответственно, данные в формате JDF должны быть доступны одновременно нескольким устройствам. Кроме того, из соображений сохранения данных целесообразнее задавать некоторому устройству только те данные, которые ему необходимы, а не весь документ. В обоих случаях необходимо дублировать части JDF - кода и передавать их на другие устройства. Поскольку эти устройства могут добавлять к скопированным данным другую информацию, а к моменту завершения производственного процесса его протокол должен быть снова в одном JDF - документе, должна быть возможность вернуть скопированные и измененные части в исходный документ. Чтобы этот процесс функционировал, нужно знать некоторые детали. Прежде всего, не все

устройства должны изменять JDF - код, нужно предварительно задать, какие устройства могут только *считывать* (*read*), а какие и *считывать*, и *записывать* (*read-write*). Иначе говоря, должна обеспечиваться согласованность данных. Этот метод называется на профессиональном жаргоне *Spawn* и *Merge*, «размножение» и «слияние». Вместо «размножения» мы будем использовать термин «разделение» - хотя это, скорее всего, процесс копирования.

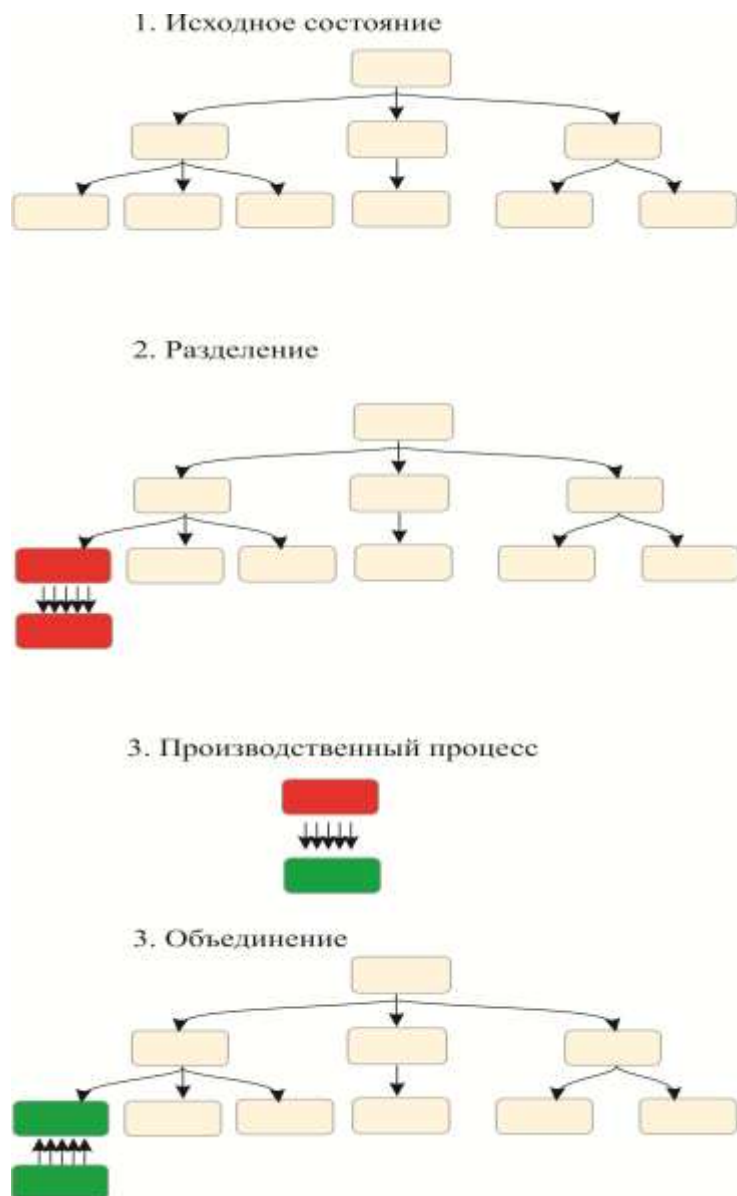


Рис. 6.26 Процессы разделения и объединения

На рис. 6.26 описанная ситуация представлена графически. Вверху (рис. 1.) представлен исходный документ, от которого (рис. 2.) отделяется JDF - узел и создается новый документ. На рис. 3. отчетливо видно, как отделенный узел изменяется независимо от исходного документа (становится ярко-зеленым), и на рис. 4. он опять присоединяется к исходному дереву. Отделенный узел может, в свою очередь, создавать дочерние узлы.

Ресурсы также могут отделяться, это будут разделяемые ресурсы. Какие ресурсы разделяются, нужно заранее задать в исходном документе. Процессы разделения и объединения протоколируются в элементе *Auditpool*.

Дальше рассмотрим пример записи разделения и объединения. На рис. 6.27 представлено JDF – задание на печать перед разделением. Шагом к разделению является отделение части «обложка» (например, чтобы выпустить ее на другой производственной площадке).

```
<JDF DescriptiveName="Gesamtprodukt" JobID="_001"...>
  <JDF ...DescriptiveName="Umschlag" ID="_002" JobPartID="_003"
    Status="Waiting" Type="Product">
    ...
  </JDF>
</JDF>
```

Рис.6.27 Исходный массив данных до разделения

На рис. 6.28 видно, как изменяется главный файл после разделения: В элемент *Auditpool* вносится новый идентификатор - *NewSpanID*, который оказывается в отделенном файле. Кроме того, записывается также *URL* отделенного файла, в атрибуте *jRef* указывается *идентификатор* отделенного узла, и, наконец, в атрибуте *rRefsROCopied* – отделенные ресурсы (R1–R9). Буквы *RO* в имени атрибута означают, что ресурсы предназначены только для чтения (*Read-Only*), т.е. процессы, обрабатывающие отделенную часть, не могут вносить в них изменения. Следует еще раз напомнить, что отделенные части дерева JDF, в частности здесь узел «обложка», в дальнейшем снова окажутся в исходном документе.

```

<JDF DescriptiveName="Gesamtprodukt" JobID="_001"...>
  <AuditPool>
    <Spawned NewSpawnID="_004" Status="Waiting"
      TimeStamp="2008-10-29T09:38:09+02:00"
      URL=file:///file:/C:/GetrennteDatei.jdf
      JRef="002" rRefsR0Copies="R1 R2 R3 R4 R5 R6 R7 R8 R9" />
    </AuditPool>
    <JDF DescriptiveName="Umschlag" ID="_002" JobPartID="_003"
      Status="Spawned" Type="Product">
      ...
    </JDF>
  </JDF>

```

```

<JDF ... DescriptiveName="Umschlag" ID="_002" JobID="_001" JobPartID="_003"
  SpawnID="_004" Status="Waiting" Type="Product">
  </JDF>
  ...
  <AncestorPool>
    <Ancestor FileName="file:/C:/Originaldatei.jdf" JobID="_001"... />
    ...
  </Ancestor>
</AncestorPool>
</JDF>

```

Рис. 6.28 (верхний) Исходный файл после отделения

Рис. 6.29 (нижний) Отделенный файл в формате JDF

Рис. 6.29 показывает отрывок файла с отделенными данными в формате JDF. В этом случае идентификаторы (*JobID*, *JobPartID*) не изменяются, к ним лишь добавляется атрибут *SpawnID* (тот же номер, как раньше у атрибута *NewSpawnID*). К тому же в элементе *Ancestor* (исходное задание) появляется информация об исходном задании, в нашем примере – это имя файла, место его хранения, а также идентификатор задания. Эта информация необходима, чтобы в дальнейшем корректно собрать все данные. Теперь главный файл и отделенный файл могут развиваться независимо друг от друга, особенно если отделенный файл открыт для записи. В завершении оба дерева должны объединиться. Результат представлен на рис. 6.30, он идентичен исходной форме до отделения. Только в элементе *Auditpool* есть запись о том, что в процессе работы были отделены и позднее снова присоединены определенные части. Весь процесс рекурсивен, т.е. отделенные раньше элементы можно снова отделить.

```

<JDF DescriptiveName="Gesamtprodukt" JobID="_001"...>
  <AuditPool>
  ...
  <Spawned NewSpawnID="004" Status="Waiting"
    Timestamp="2008-10-25T09:38:09+02:00" URL="file://file:/C:/Test.jdf"
    JRef="002" rRefARCOopies="R1 R2 R3 R4 R5 R6 R7 R8 R9 />
  <Merged ... MergeID="004" Timestamp="2008-10-25T09:41:01+02:00"
    URL="file://file:/C:/GetrennteDatei.jdf" JRef="002" />
  </AuditPool>
  ...
  <JDF DescriptiveName="Umachlag" ID="002" JobPartID="_003"
    Status="Waiting" Type="Product"...>
  ...
</JDF>
</JDF>

```

Рис. 6.30 Файл в формате JDF после объединения.

6.7 ICS (Interoperability Conformance Specification)

При взаимодействии двух разных сторон часто возникают проблемы. Когда, например, рекламное агентство делает заказ на издание некоторого печатного продукта, его специалисты не всегда знают, как правильно оформить необходимые контент-данные. Сотрудники типографии, в свою очередь, не знают, что в деталях необходимо специалистам рекламного агентства, из-за этого возникает недопонимание и разочарование. Хоть коммуникация и происходит на одном языке, её результаты часто не удовлетворяют ни одну из сторон. Выходом из такой ситуации служит договоренность, иными словами, некоторая стандартизация оформления данных на основе PDF-стандартов, PDF/X [23] или PDF/X-Plus [19], которые лимитируют возможности формата *Portable Dokument Formats*.

Подобные проблемы возникают и при организации рабочего потока на основе формата JDF. Когда агент (*Agent*), контроллер (*Controller*) или устройство (*Device*) оперируют определенными данными в формате JDF, предполагается, что получатель данных их понимает. И наоборот, получателю, возможно, нужны некоторые данные, которых ему не хватает. Здесь на помощь приходит спецификация ICS (*Interoperability Conformance Specification*). Она ограничивает функциональность некоторых компонентов программы, которую предлагает сетевое обеспечение JDF/JMF.

В подобных проблемных ситуациях всегда участвуют две стороны: одна генерирует информацию, а другая ее принимает, по крайней мере, пытается это сделать. И здесь уже не хватает терминов *Agent*, *Controller* и *Device*, необходимо рассмотреть два новых понятия, используемых в системах рабочего потока (это модули):

- *менеджер (Manager)* отправляет JDF - документы в первую очередь *исполнителю (Worker)-контроллеру (Controller)* или *устройству (Device)*). Он может пересылать сообщения в формате JMF. И лишь после пересылки он может считывать документы и сообщения, которые присылают ему *Controller* или *Device*

- *исполнитель (Worker)* получает от *менеджера (агента или контроллера)* документы, а также возможные JMF - сообщения. И затем он также может обратно отправить информация в формате JDF/JMF.

Можно поспорить о корректности выбора слов *менеджер* и *исполнитель*, но мы не будем этого делать, и продолжим пользоваться данными терминами. Следует заметить, что как *менеджер*, так и *исполнитель* могут и считывать, и записывать информацию – JDF - данные и JMF - сообщения. Например, менеджер отправляет исполнителю некоторое задание на JDF - процесс, исполнитель выполняет его и после завершения пересылает данные об израсходованных и созданных ресурсах. Если использовать иную терминологию и учитывать при этом возможность записи и считывания информации, можно говорить о *производителе (producer)*, когда устройство записывает информацию, и о *потребителе (consumer)*, когда оно ее считывает. *Менеджер* и *исполнитель* выступают то в роли производителя, то в роли потребителя.



Рис. 6.31 MIS - интерфейсы, записанные в ICS - документах

В ICS - документах описываются различные интерфейсы, которые частично базируются друг на друге. Разные MIS - интерфейсы, возможные в ICS -документах, основанные на версии JDF 1.3, представлены на рис. 6.31. Весь комплект ICS-документов в их иерархии изображен на рис. 6.32. Например, *MIS – послепечатная обработка (MIS to Finishing ICS)* основана на интерфейсах *Base ICS*, *JMF ICS* и *MIS ICS*, а *офисная цифровая печать (Office Digital Printing ICS)* – только на интерфейсе *Base ICS*.

Кроме того, отдельные интерфейсы, в свою очередь, подразделяются на разные уровни, которые также имеют иерархическую структуру.

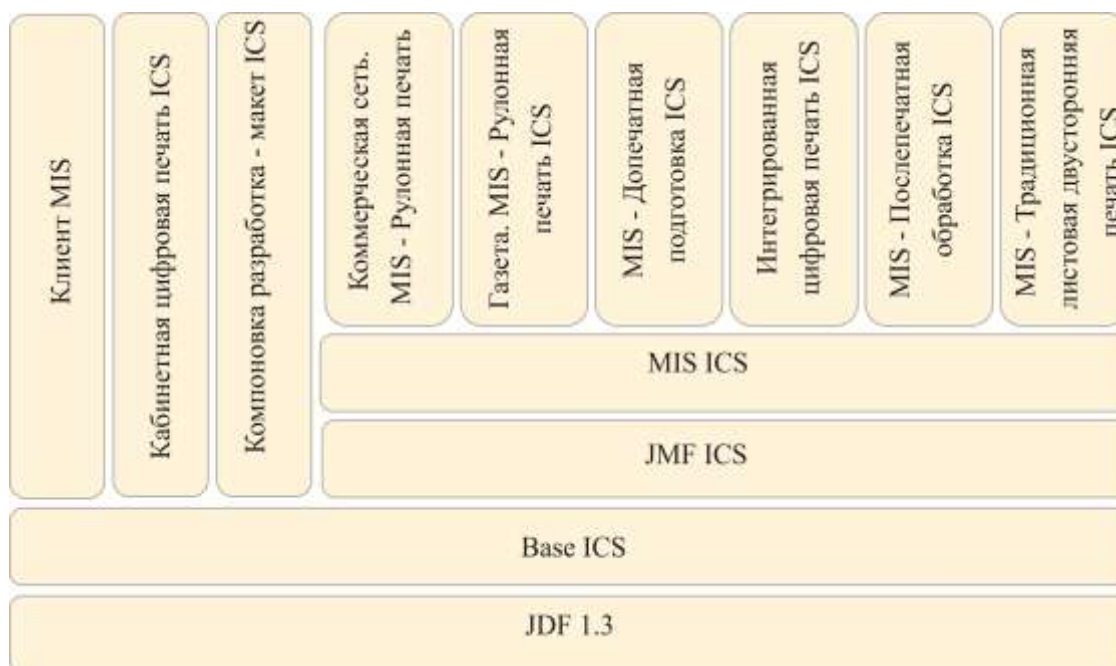


Рис. 6.32 ICS - документы частично базируются друг на друге

Некоторые ICS - документы подробнее рассмотрим в следующих главах. А сейчас несколько подробнее представим *Base ICS*. В нем различают три уровня, уровень 0, уровень 1 и уровень 2, причем уровень 1 охватывает функциональность уровня 0, а уровень 2, соответственно, функциональность уровня 1. Во всех остальных ICS-документах задается необходимый уровень *Base ICS*. Уровня 0 достаточно только интерфейсу, когда *менеджер* передает *исполнителю* JDF - данные по «горячей линии», а тот лишь считывает их. Уровень 1 позволяет обеспечить обратную связь, JDF - документы передаются также от *исполнителя менеджеру*, и он может их считывать. Уровень 2 должен отвечать следующим требованиям:

- при реализации системы управления рабочим потоком может понадобиться перенос большого количества данных одновременно: JDF - документы, JMF -сообщения, данные в формате PDF, предварительный профиль Preflight-Profile, цветовой профиль ICC, и предварительное изображение. Они могут быть упакованы в один пакет, например, MIME (*Multipurpose Mail Internet Extensions*, по аналогии с электронной почтой E-Mail, где

можно послать вместе много разных данных). В уровне 2 менеджер формирует MIME - пакет, а исполнитель интерпретирует его (рис. 6.33);

- как правило, внутри JDF - документа могут быть выделены, например, PDF - файлы. В этом случае имеется ссылка на сервер в локальной сети, или путь к базе данных, в которой находится файл. На рис. 6.34 имеется список (*RunList*), состоящий из одного PDF - документа (*LayoutElement*), его имя и место хранения заданы в информации о доступе (*спецификация файла - FileSpec*). Атрибут "0~ 1" означает, что нужно использовать все страницы этого файла. Уровень 2 требует, чтобы менеджер сделал эти файлы доступными на языке HTTP, а исполнитель смог их на этом языке считать.

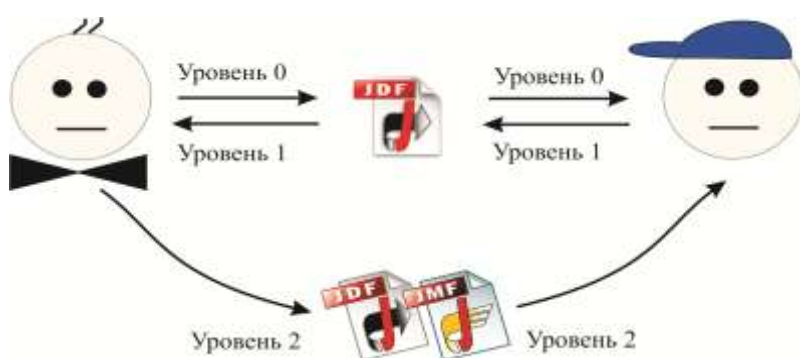


Рис. 6.33 Различные уровни в Base ICS

Помимо данных уровней база (*Base*) *ICS* проясняет много технических деталей, например, из какого количества знаков может состоять идентификатор. Интересны ограничения, которые ставятся узлам JDF и их атрибутам и субэлементам. Например, задается, что менеджер корневого элемента записывает идентификатор задания (*JobID*), а исполнитель должен его считать, или задается, кто должен вносить опциональные аудит - записи в элемент *AuditPool* и т.д.

```
<ResourcePool>
  <RunList ID="_4719" Pages="0~1" Class="Parameter" Status="Available">
    <LayoutElement>
      <FileSpec URL=file://Workflowserver/Pagefiles/Frisch-Werbung.pdf />
    </LayoutElement>
    ...
  </RunList>
  ...
</ResourcePool>
```

Рис. 6.34 Ссылка на данные через URL

В заключение стоит сказать, что ICS - документы являются не только инструментом для создания программы в формате JDF, они представляют собой основу для интеграции и проверки взаимодействия между компонентами в формате JDF. В случае возникновения сбоев в работе следует сначала проверить, все ли участники процесса соблюдают правила системы ICS.

Задания для самостоятельной работы:

- скачайте с сайта CIP4.org редактор *JDFEditor*, и некоторые примеры файлов в формате JDF. Откройте их и изучите структуру JDF - узлов и ресурсов;
- переименуйте файлы с расширением .jdf в файлы с расширением .xml, откройте их с помощью браузера. Проанализируйте код XML на конкретных примерах;
- с помощью редактора отделите какой-нибудь JDF - узел, а затем присоедините его к исходному документу. С помощью браузера проанализируйте промежуточные состояния.

Глава 7. Формат рабочих сообщений (JMF)

В предыдущей главе мы рассмотрели, как посредством JDF может происходить обмен данными между различными агентами, руководством и JDF-устройствами. В простейшем случае JDF-данные пишутся JDF - производителем и размещаются в *горячей папке (Hotfolder)* JDF - потребителя, который в определенные промежутки времени пересматривает их после поступления. Это нереверсивный интерфейс пересылки файлов. В основном, процесс статичный, динамичное взаимодействие происходит с участником рабочего потока, который, например, является очень важным для сбора производственных данных, учета последовательности работ, изменений в процессе, учета ошибок, подтверждения использования материалов, отображения коэффициента использования машин и для управления системой, что особенно важно при ее запуске. Для этого вместе с *JDF форматом данных* введен *JMF формат передачи сообщений*. Не в каждом JDF - рабочем потоке реализованы JMF - каналы сообщений, мы постараемся показать пользу, которую они могут приносить. Для MIS, к примеру, пишутся сначала определенные JMF сообщения MIS ICS уровня 2.

Общее понятие технологии JMF очень важно для понимания того, как функционирует весь JDF - JMF рабочий поток. В этом случае нужно четко представлять, как оптимально задействовать пользователя JMF сообщений. Так как данные JDF преимущественно существуют как доступная информация для анализа, нахождения возможных ошибок, JMF информация использует HTTP протокол, который не имеет возможности сохранения. Естественно, можно применить *HTTP-Sniffer* программы, которые перехватывают HTTP-пакеты и делают их доступными. Однако при большом количестве сообщений, которые обычно посылаются по сетевым каналам, этот способ не всегда является удачным. Поскольку в этой книге мы будем больше акцентировать внимание на JDF - формат, поэтому некоторые тонкости применения JMF сообщений будут опущены.

7.1 Модели коммуникаций

Прежде, чем мы рассмотрим структуру JMF - протоколов, необходимо обсудить, что общего в различных моделях коммуникаций, которые реализуются между модулями программного обеспечения при работе с графическими изображениями.

Отметим следующее:

- пересылка файла через *горячую папку (Hotfolder)* (или также ручную);
- пересылку данных через протоколы печати;
- интерфейсы банка данных;
- международные средства коммуникации (Интернет и др.);
- сервисы сети;
- протоколы сообщений.

Пересылка файлов через Hotfolder имеет выгодное преимущество за счет простоты использования и реализации связи преимущественно системными администраторами. Имеются и некоторые недостатки:

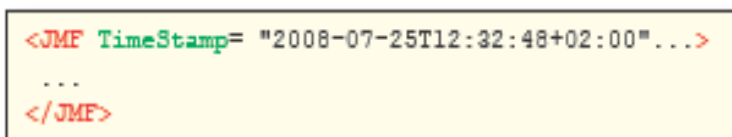
- когда функционирование Hotfolder прерывается (например, не работает компьютер) и нет сообщений о сбое;
- производитель данных не получает никакого уведомления о том, что его информация успешно доставлена, прочитана и правильно интерпретирована;
- в случае ошибки при выполнении работы отсутствует возможность у контролирующей инстанции определить её причины;
- коммуникации осуществляются медленно и не соответствуют оптимальному производственному обмену данных, подтверждению связи при одновременной работе двух или нескольких модулей.

Hotfolder широко используется и будет в дальнейшем применяться при RIP - обработке, что связано с размещением данных. Как правило, эти папки распределяются по видам, которые ассоциируются с установками RIP (разрешение, частота полиграфического растра...), что, однако, достаточно запутано. При использовании JDF этой сложности нет, так как метаданные, а также установочные данные для RIP могут вноситься в соответствии с требованиями технологии, а Hotfolder используется только как коммуникационный канал. В этом случае вместо пересылки файлов, данные могут поступать в RIP через протоколы печати, позволяющие осуществлять специальные регулировки для работы. Также распространена передача данных в формате JDF в базы данных, но в этом случае процесс является недостаточно рациональным, к тому же не существует стандартизированных методов доступа. Это же относится к дополнительным процессам взаимодействия, которые предоставляются отдельными поставщиками. В них

адреса пересылки определяются, например, при PJTF - коммуникации между процессорами рабочей карточки.

Сервис сети определяет обмен информацией между приложениями на базе XML [45]. Таким образом, может функционировать, к примеру, система сервисов сети для бюро путешествий по резервированию гостиниц, учету и удовлетворению всех запросов. По аналогии отдельные модули системы управления рабочего потока могут обмениваться сообщениями внутри системы.

Сервис сети нуждается в протоколах сообщений, например SOAP (первоначально *Simple Object Access Protocol*) от W3C. Этот протокол применяется, в действительности, для систем рабочего потока. Он служит обмену XML – сообщениями и может использоваться для JMF. Применяется преимущественно протокол HTTP. Сообщения JMF – это короткие XML документы, которые имеют общий коренной элемент с обозначением JMF, что можно увидеть на рис. 7.1. В подэлементах JMF - корня кодируется содержание сообщения. Сообщение отсылается, как правило, с помощью протоколов HTTP или HTTPS, т.е. протоколами, используемыми в Интернете. При этом может быть выстроено быстрое двунаправленное общение. Альтернативно могут быть созданы некоторые JMF - сообщения, которые, также как и файлы в Hotfolder, получатель должен снова прочитать.



```
<JMF TimeStamp= "2008-07-25T12:32:48+02:00"...>  
...  
</JMF>
```

Рис. 7.1 JMF-сообщение

В подэлементах JMF - корня кодируется содержание сообщения. Сообщение отсылается, как правило, с помощью протоколов HTTP или HTTPS, т.е. протоколами, используемыми в Интернете. При этом может быть выстроено быстрое двунаправленное общение. Альтернативно могут быть созданы некоторые JMF - сообщения, которые, также как и файлы в Hotfolder, получатель должен снова прочитать.

Вместе с тем, этот способ подобен тому, как и при широко распространенном JDF - формате (это представлено на рис. 7.2).

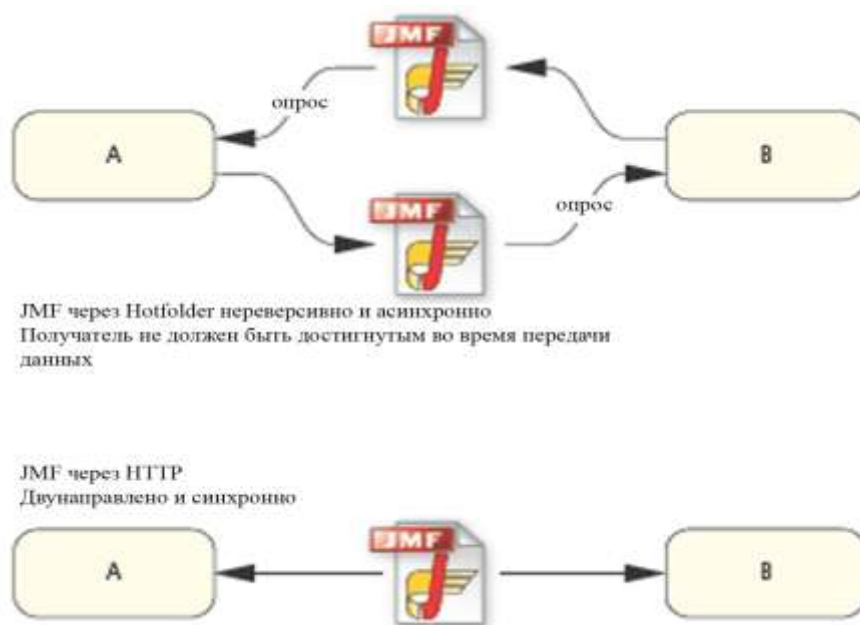


Рис. 7.2 Различные виды JMF - коммуникаций

7.2 JMF - семейства

Какие сообщения являются важными и могут передаваться с применением JMF? Вот несколько примеров:

- инициализация устройств;
- состояние устройств и их статус работы;
- контроль ожидания очередей и их регистрация;
- исполнение этапов (заданий).

Прежде всего, выделим из перечня передачу данных о состоянии устройств и выполнении особо важных заказов на печать, так как от этого зависит коммерческая выгода. *Отслеживание работ (job-tracking)* возможно вместе с оформлением отчетной калькуляции. Наблюдение за работой устройств легко реализовать на основе JDF - сообщений. Проблема возникает скорее в фильтрации больших массивов данных с различных точек зрения. Отчетная калькуляция часто готовится не только на базе JMF - сообщений, но и с помощью *аудиторов (Audit-Pools)*, которые рассматривались в параграфе 6.2. *Audit* - записи регистрируются типичным способом в определенном JDF - файле, который после завершения процессов системой или центральным сервером JDF -

рабочего потока возвращается обратно. Фильтр JDF - сообщений должен быть определен менеджером, чтобы послание аудитора передавалось исполнителю.

JMF - сообщения делятся на шесть категорий, так называемые JMF - семейства:

- JMF - *запрос (Query)*;
- JMF - *команда (Command)*;
- JMF - *ответ (Response)*;
- JMF - *подтверждение (Acknowledge)*;
- JMF - *сигнал (Signal)*;
- JMF - *регистрация (Registration)*.

Для получения определенной информации JMF - запрос направляется на JMF - контроллер или на JMF - устройство. Он изменяется в противоположность команде, а не по состоянию адресата. Он реагирует на запрос, как правило, с *ответом (Response)*. На рис.7.3 представлен пример запроса и ответа в графическом виде. Каждый контроллер и каждое устройство имеет свой идентификатор, также кодируются сигналы запроса и ответа. Таким образом, на последовавший запрос ожидается конкретный ответ. В этом запросе контроллер с идентификатором ID=4711 хотел бы получить сведения об устройствах, которыми управляет контроллер с ID=42. Такой запрос очень важен, к примеру, в системах Plug & Play для модулей JDF (причем объект управления может находиться на значительном удалении). Устройство отвечает на запрос только тогда, когда управляет RIP и СтР.

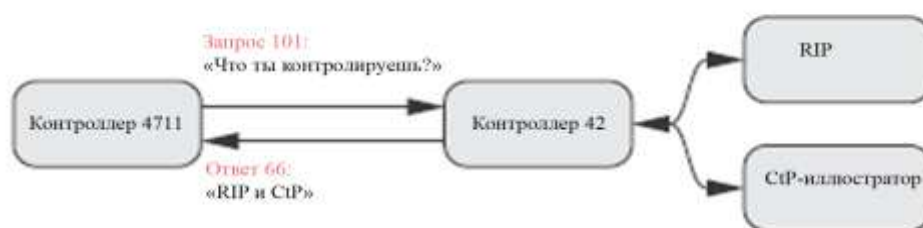


Рис. 7.3 JMF - запрос и – ответ

На рис. 7.4 представлен соответствующий JMF - код. Может быть, бросается в глаза, что ID - получателя ни в запросе, ни в ответе не содержит XML - кода.

```

<JMF TimeStamp="..." SenderID="4711">
  <Query Type="KnownDevices" ID="101"/>
</JMF>

<JMF TimeStamp="..." SenderID="42">
  <Response Type="KnownJDFServices" ID="66" refID="101"/>
  <DeviceList>
    <DeviceInfo DeviceStatus="Idle">
      <Device DeviceID="Rip" />
    </DeviceInfo>
    <DeviceInfo DeviceStatus="Running">
      <Device DeviceID="CeP"/>
    </DeviceInfo>
  </DeviceList>
</Response>
</JMF>

```

Рис.7.4 JMF - запрос на ответ

Так идентификатор ID-42 не содержится в запросе, а ID=4711 отсутствует в ответе. Тогда, каким образом сообщения вообще могут достичь правильного адресата? Решение этой загадки в том, что сообщение передается получателю через HTTP - протокол (или также через *Hotfolder*). Для наглядности можно представить, как сообщение JMF вкладывается в конверт HTTP с соответствующими почтовыми наклейками и затем отсылается адресату. Узнать об однозначной отправке сообщения (на ранее отправленный запрос) можно также по атрибуту *ref ID*.

Команды могут изменять получателя. Это относится и к JMF - командам. Например, посредством команды можно оставлять или исключать получателей из очереди только по одному статусу. И прежде, чем мы продолжим, определимся, что такое ожидание очереди? Здесь имеется в виду, конечно, не очередь перед кассой в супермаркете, а общая структура данных, которая действует в качестве накопительного буфера, который работает по единому принципу «Первый В – Первый Из» (*First in – First Out*) (кто первый пришел, тот первый и начинает). Это может быть, например, буфер данных, в котором временно хранятся созданные для RIP задания, их можно также разместить в рабочую память JDF - работа перед контроллером или устройством.

Относительно очередей имеется два вида JMF - команд. Либо они касаются индивидуальных получателей в пределах одной очереди, либо для набора очередей. Так, если команда *Hold Queue* останавливает всю очередь, и никакая работа далее не исполняется, в то время как команда *Hold Queue Entry* приостанавливает только отдельную работу из очереди. Даже если команды относительно ожидания очередей или их регистрации, на первый взгляд, кажутся неважными или даже бесполезными, тогда должно быть ясно, что без них рабочий поток функционировал бы с ограничениями. К примеру, если устройство может сообщить контроллеру, что работа больше не должна ждать и активно запускается, при этом информация актуализируется и становится доступной для пользователей, обновляется и график состояния производства.

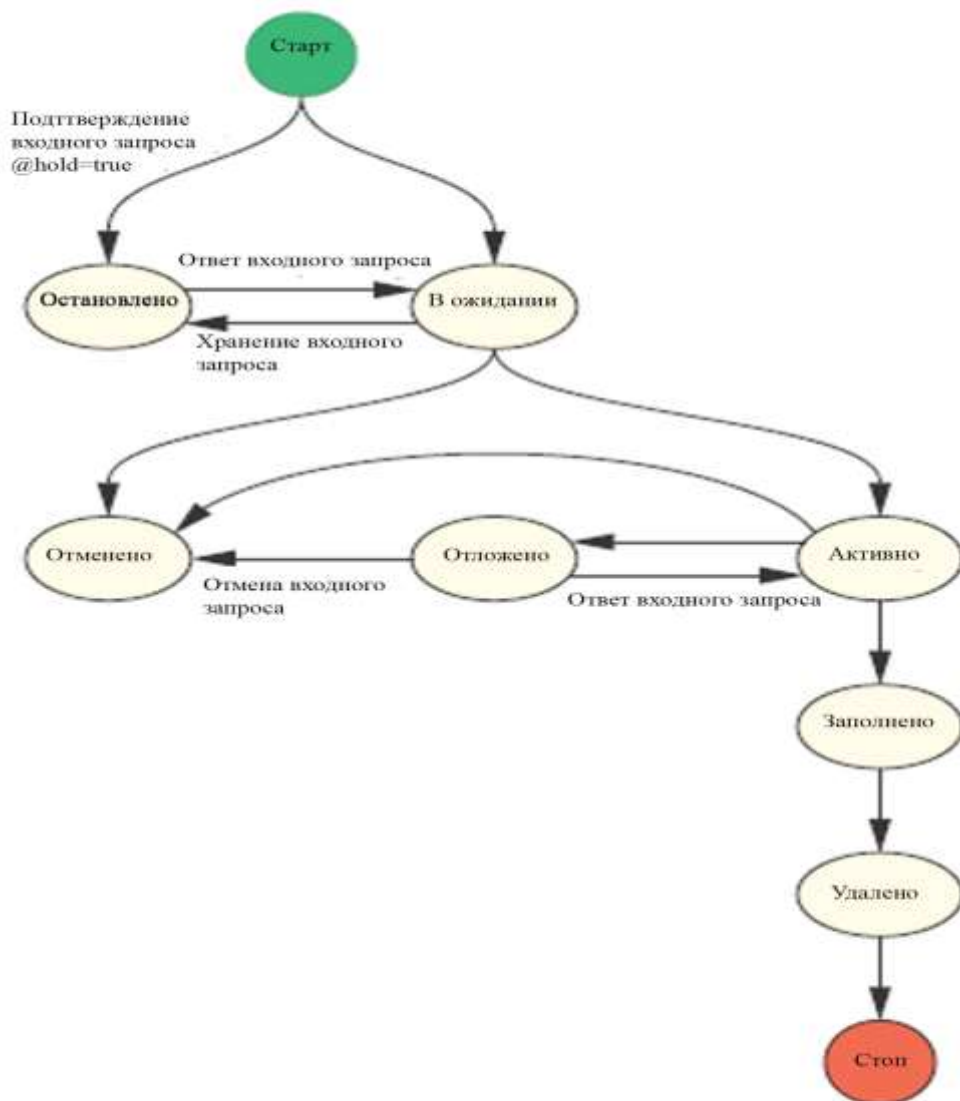


Рис. 7.5 Состояния и переходы состояний в элементах очереди

У элементов очереди существуют разные состояния, например:

- в ожидании (*Waiting*);
- остановлено (*Held*);
- активно (*Running*);
- отложено (*Suspended*);
- отменено (*Aborted*);
- удалено (*Removed*).

Эти состояния могут касаться не только очередей ожидания, но и самих устройств, что наглядно отражено на рис. 7.4 (на котором можно распознать *статус устройств - DeviceStatus*).

На рис. 7.5 показаны состояния и некоторые переходы (чтобы не перегружать рисунок, некоторые элементы опущены). Кроме того, в некоторых случаях JMF -команды вызывают соответствующие переходы состояний. Обращает на себя внимание различия между состояниями «остановленный» (*Held*) и «отодвинутый» (*Suspended*). В обоих случаях работы приостанавливаются, дальнейшая обработка осуществляется один раз из состояния «ожидание» (*Waiting*) и в другой раз - из состояния «активно» (*Running*).

```
<JMF SenderID="Controller-1" TimeStamp="2008-08-13T10:05:32+01:00"
Version="1.3"...>
  <Command ID="C1" Type="AbortQueueEntry">
    <QueueEntryDef QueueEntryID="job-4711" />
  </Command>
</JMF>

<JMF SenderID="Device-1" TimeStamp="2008-08-13T10:05:33+01:00"
Version="1.3"...>
  <Response ID="R1" Type="AbortQueueEntry" refID="C1">
    <Queue DeviceID="Device-1" Status="Running">
      <QueueEntry JobID="job-4711" QueueEntryID="job-4711"
        Status="Aborted" />
    </Queue>
  </Response>
</JMF>
```

Рис.7.6 Команда, которая направлена на «ожидание очереди» и соответствующий ответ

Команда для отмены работы, а также ответ на эту команду представлены на рис. 7.6 . Контроллер-1 посылает команду *AbortQueueEntry* – команду (C1) относительно элемента очереди *job-4711*. Получателем этой команды является устройство (*Device-1*), его ответ поступает на команду C1. Отмененная работа снова регистрируется в очереди, разумеется, со *статусом отменено (Aborted)*.

Файлы JDF - заданий, как мы уже знаем, могут передаваться через Hotfolder. Существует и другая возможность сообщать контроллеру или устройству, как и где эти JDF - файлы могут быть забраны, при этом имеется два варианта: или стандартная передача данных, или через HTTP. Пример кода на рис.7.7 определяет способ передачи.

```
<Command ID="C2" Type="SubmitQueueEntry">
  <QueueSubmissionParams URL="File://HdM/Server1/Jobs/job1.jdf" />
</Command>
```

Рис. 7.7 Индивидуальный элемент очереди

Другой пример для команд имеется при структуре данных *Pipe* (канал, а также способ передачи вывода одной команды на вход другой), что кратко было рассмотрено в параграфе 2.6. В производстве имеются процессы, выполняемые последовательно с задержкой по времени. Вывод команд процесса P1 является ресурсом в *Pipe* для второго - процесса P2. Особенность этой *Pipe*-структуры состоит в том, что команды процессов вводятся не комплектно, запуск одного вырабатывает только по его окончании команды на запуск следующего. Так в параграфе 2.6 процессом P1 являлся процесс подготовки форм, а P2 – печать.

Действия при этом следующие: из Процесса P1 данные по команде *PipePush* пересылаются и по команде *PipePull* извлекаются в Процесс P2, что и продемонстрировано на рис. 7.8.

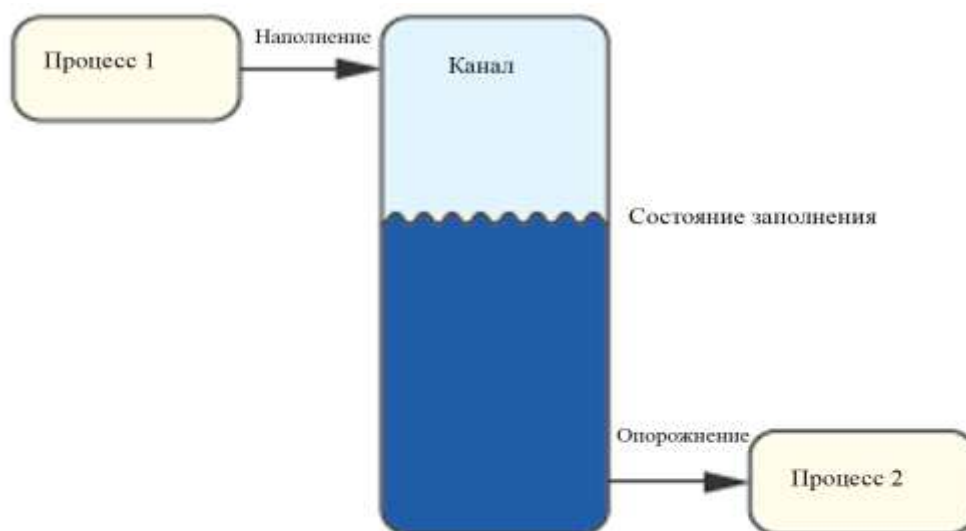


Рис.7.8 Pipe - система и JMF - команды к наполнению и опорожнению

Наконец, мы хотели бы упомянуть еще *ресурс сообщений (Resource Messages)*. Это *запросы* или *команды* относительно JDF - ресурсов. Таким образом, ресурсы запросов по материалам, устройствам или работам (*ResourceQuery*) могут модифицироваться в ресурсы команд (*ResourceCommand*). С помощью команд может посылаться на устройство либо новая версия JDF-ресурсов, либо обновляться существующие данные. Если, например, процесс экспонирования формной пластины завершился, соответствующий контроллер посылает команду, чтобы установить статус JDF-ресурса *готов к работе*. Под подтверждением JMF понимается переданный через некоторое время (асинхронно) ответ

на запрос или команду. Команды и запросы могут довольно долго находиться в ожидании, до тех пор, пока они не будут востребованы. Получатель сначала посылает ответ в получении запроса или команды, а только потом подтверждение, что началось выполнение задания. Однако в нормальной жизни терминология противоположна: первоначально подтверждается выполнение задания, а потом дается ответ.

Сигналы – это нереверсивные сообщения, которые, чаще всего, касаются статуса машин или работ. Контроллеры могут работать с различными видами и способами получения таких сигналов-сообщений, это означает полную осведомленность о том, какое известное сообщение необходимо отправить или получить. На рис. 7.9 отображена структура сообщения, которое информирует о статусе работы печатной машины.

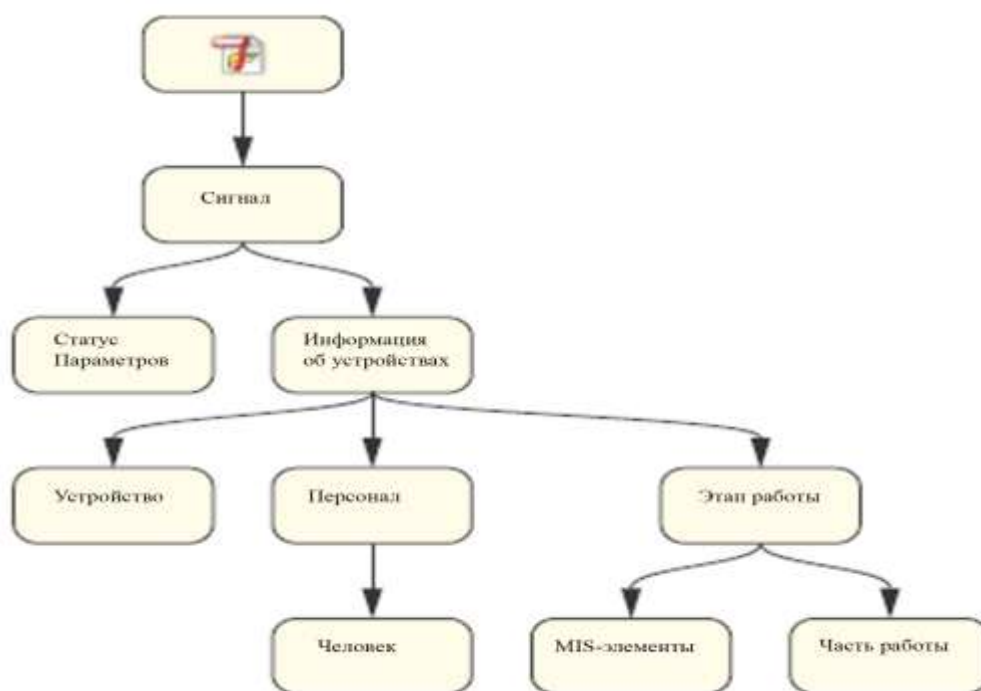


Рис.7.9 Структура JMF - сообщения

JMF – сообщение представляет собой запись, которая содержит два элемента. Элемент *статус параметров* (*StatusQuParams*) определяет работу, на которую получен заказ. А элемент *DeviceInfo* дает *информацию об устройстве* с указанным статусом и количественные данные о продукции и т.д. В подэлементах *Device*, *Employee* и *JobPhasa* находится более подробная информация об устройстве, операторе, обслуживающем устройстве (если он зарегистрировался) и о работе. На рис.7.10 в кратком виде представлен код этого сигнала. В элементе *JobPhasa* содержится информация о том, что работа завершилась (*статус завершено*), при этом напечатано 1002 листа из них 71 лист -

макулатурный. В элементе *MISDetails* отмечено, что клиент должен платить за размещенный заказ (атрибут *CostType* установлен на *Chargeable*), и что заказ напечатан без изменений и соответствует оригиналу (*WorkType*). Атрибут *Device-Status* в элементе *Device-Info* стоит на *Idle*, что означает *бездействует*; другие возможные состояния: *вниз* (*Down*), *установка* (*Setup*), *выполнение* (*Running*), *очищено* (*Cleanup*), *остановлено* (*Stopped*) или *неизвестное* (*Unknown*).

Следующий атрибут в том же самом элементе (на рис. 7.10 его нет) называется *статусом деталей* (*StatusDetails*). В нем содержатся более точные данные состояния машины или специальных печатных машин.

```
<JMF SenderID="_4004"...>
  <Signal ID="81" Type="Status">
    <StatusQuParams JobID="_0001" JobPartID="_002"... />
    <DeviceInfo DeviceStatus="Idle" TotalProductionCounter="13862.0">
      <Device DeviceID="_4004" Status="Available"... />
      <Employee ID="111" PersonalID="_013" Roles="Operator"
        Status="Available">
        <Person FamilyName="Cool" FirstName="Carl"... />
      </Employee>
      <JobPhase JobID="_001" JobPartID="_002"
        StartTime="2008-07-22T13:16:12+02:00" Status="Completed"
        TotalAmount="1002.0" Waste="71.0">
        <MISDetails CostType="Chargeable" WorkType="Original" />
        <Part SheetName="Blaetter" SignatureName="SIG1" />
      </JobPhase>
    </DeviceInfo>
  </Signal>
</JMF>
```

Рис. 7.10 Пример JMF - сигнала

Список 7.11 «Значения атрибута *StatusDetails*» содержит перечень основных возможностей. Этот список исчерпывающе поясняет специализацию форматов JDF и JMF.

Список 7.11 Значения атрибута *StatusDetails*

Общее изменение данных, Полная смывка, Поломка, Очистка красочной системы, Задержанное управление, Открытая поверхность, Смывка цилиндров, Смывка охлаждающих валов, Открытый кожух, Отказ, Смена формы, Хороший, Смывка красочных валиков, Обслуживание, Полная область вывода, Смывка пластин, Восстановление, Закрыть, Изменение размера, Смена сливса, Ожидайте одобрения, Нагревание, Промывка UV, Расход (отход)

Последняя составляющая JMF - семейства – регистрация. При этом речь идет о возможности поручить отправление определенных команд третьей стороне. К примеру, MIS

смог бы вызывать первоначально WMS и отправлять SIP-3 данные на определенную печатную машину. Эта иерархия команд нуждается в прочных *коммуникационных каналах (persistent channels)*, которые необходимо в начале создать. Так как эти каналы важны для сигналов, мы хотели бы на них подробно остановиться в следующем разделе. Иначе, дальше специальная информация не будет поступать. Напоследок нужно еще упомянуть, что JMF - протокол может быть расширен. Посредством подчинения именных пространств могут создаваться новые элементы и атрибуты.

7.3 JMF- ICS (интегральная коммуникационная подсистема)

Интегральная коммуникационная спецификация (Interoperability Conformance Specification) для JMF [12] специфицирует условия для JDF-/JMF - модулей в два уровня (*levels*).

** Уровень 1*

- a) Данные JDF передаются в *Hotfolder (горячую папку)*.
- b) Менеджеры могут установить прочные коммуникационные каналы, что сделает возможным размещение соответствующих запросов и регистрации в JDF -документах.

** Уровень 2*

- a) Менеджеры могут установить прочные коммуникационные каналы JMF -сообщений непосредственно через HTTP.
- b) Данные JDF могут передаваться через JMF.
- c) Осуществляется двунаправленный обмен информацией JMF между *менеджером (Manager)* и *исполнителем (Worker)*. В частности могут реализовываться условия ожидания очереди через JMF.

Элемент *NodeInfo* представлялся уже в параграфе 6.5, как пример назначения срока для выполнения процесса печати, который должен быть реализован. С помощью этого ресурса могут создаваться *прочные коммуникационные каналы (persistend channels)*, как это изложено в пункте b) уровня 1. Эти прочные коммуникационные каналы требуются для того, чтобы контроллер смог принять сигнал от специальных JMF - устройств. Таким образом, MIS регулярно может получать информацию о состоянии печатной машины и изготовленной продукции.

Понятие «*abonnieren*» (подписывать) является волшебным словом, так как оно, в действительности, будет элементом *подписка, абонемент (Subscription)*, встроенным в ресурс *NodeInfo*. Реализацию отношений «родитель-ребенок» можно увидеть на рис. 7.12. В пределах запроса *Query* могут создаваться надежные коммуникационные каналы,

которые функционируют достаточно долго, пока не прервутся или пока не обработаются соответствующими JMF - узлами.



Рис. 7.12 Опрос с помощью *NodeInfo* абонемента

Конечно, запросы по этим коммуникационным каналам могут быть вложены JMF - узлами в HTTP и потом отправлены. Это соответствует: требованиям а), которые должны быть выполнены на Уровне 2.; требованиям б) и с), которые ранее уже представлялись. В частности, мы видели на рис. 7.7 , как благодаря команде от *SubmitQueueEntry* JDF - задание может передаваться в устройство через протокол HTTP. Наконец, мы хотели бы еще раз независимо от ICS - уровня обобщить различные возможности передачи файлов в JDF/JMF - окружении, проследив сложность дополнений по возрастанию сверху вниз:

- отсутствие JMF - сообщений;
- сигналы JMF;
- запросы JMF / ответы JMF;
- команды JMF;
- JDF - передача через HTTP.

Задания для самостоятельной работы:

- загрузите HTTP-Sniffer как бесплатное программное обеспечение из Интернета и установите эту программу. Попробуйте перехватить в Sniffer и прочесть с помощью просмотрной программы данные, генерируемые HTTPakete.

Глава 8. Системы управления заказами

Системы управления заказами (AMS) в полиграфии имеют много названий: это и информационно - управляющие системы (MIS), отраслевые решения, инициативные ресурсы планирования (ERP), системы производственного планирования (PPS) или просто программа вычислений. Конечно, между отдельными понятиями имеются отличия, но на практике часто используются синонимы, и мы в дальнейшем будем применять только понятия AMS и MIS.

Займемся не только определением понятий, но и рассмотрим существенные возможности программного обеспечения систем, как:

- предварительная и итоговая калькуляции;
- коммерческое выполнение заказов (от производства до выписки счетов-фактур);
- управление клиентскими данными;
- материально-техническое снабжение;
- интерфейс производства:
 - * передача информации о производстве;
 - * планирование производства / размещение (срок, ресурсы);
 - * учет производительности / производственный сбор данных (BDE);
 - * отслеживание хода выполнения заказов (*Job Tracking*);
- интерфейс финансов (бухгалтерии) и расчета зарплат;
- интерфейс клиентов (электронный бизнес, Интернет - порталы);
- интерфейс поставщиков (поставщики бумаги).

Типичные системы управления заказами (AMS) инсталлированы в типографиях, но не все перечисленные функции являются основными для систем управления заказами, в большинстве случаев применяются только опциональные модули, например, планирование производства. Не все модули имеют решающее значение для рабочего потока, в частности, финансовый интерфейс бухгалтерского учета и расчет заработной платы. Предварительная калькуляция, выполнение заказов, работа с клиентами, управление материальными ресурсами, напротив, являются самыми важными функциями AMS. Поэтому в следующих параграфах мы будем более подробно рассматривать некоторые из этих модулей и анализировать связи между ними. Для JDF - рабочего потока интерфейс от системы управления к производству имеет особое значение, исследовать

этот вопрос будем в следующем разделе. Интерфейс может быть также реализован и в противоположном направлении - от производства к системам AMS, причем тогда производственные данные (BDE) будут использоваться для окончательной калькуляции, а также передаваться ответы на запросы о статусе производства и сроках исполнения. Коммуникации между заказчиками печатной продукции и производителями осуществляются посредством переписки, это обсудим в параграфе 8.4.

8.1 Основные функции AMS

Задача этого параграфа не в том, чтобы описать, как создается отчетная калькуляция и как она используется. Речь пойдет только о том, какие данные необходимо ввести для калькуляции, какие расчеты проводятся для определения стоимости печатной продукции. Способ ввода данных для калькуляции естественно важен, так как они затем должны использоваться в производстве, основанном на базе JDF.

Для калькуляции печатной продукции, как правило, необходима информация из следующих областей:

- данные (сведения) заказов;
- оборудование для выпуска продукции;
- производительность машин;
- данные материального снабжения;
- производственные процессы.

Используемые производственные процессы определяются, во многих случаях, технологиями производства и производственными машинами, допечатной подготовкой и исходными данными, которые типография получает от заказчиков. В калькуляционных программах будут, по праву, существовать четкие различия между использованием приложений специалистами-пользователями и администраторами системы. В то время как специалисты-пользователи контактируют с ответственными исполнителями калькуляции, системные администраторы заботятся об основных данных для введения в систему. К основным данным, в частности, относятся:

- настройки издержек производства;
- ввод стоимости и продолжительности отдельных производственных процессов;
- описание типов продукта;
- ввод данных для машин типографии;
- ввод клиентских и материальных данных на складе;
- данные управления;
- ведения бланка заказа.

Пользователь системы может просто и быстро составить счет, если основные данные для калькуляции первоначально были установлены правильно. Зачастую, в базе данных имеется контактная информация партнера, адрес поставки, расчетные счета, кроме того, проверяется его платежеспособность (кредитоспособность).

При описании продукции определяется вид продукта (книга, брошюра, визитная карточка...), из списка выбираются и основные характеристики продукта, такие как тип переплета, количество страниц, цветность, конечный и обрезной форматы и т.д. Так, например, необходимо отдельно указывать стоимость изготовления переплета для книжного блока и полноцветной (СМΥК-печать) обложки, чтобы, таким образом, установить размер печатных форм, размер бумаги, конечный формат издания, красочность или ширину полотна при рулонной печати и др. В сложных и составных изданиях должна указываться стоимость отдельных их частей. Печатный продукт специфицируется по виду фальцовки, типу бумаги, ее весу в граммах на кв.м. и т.д. Наконец, в зависимости от продукции выбирается одна или несколько схем прохождения заказа в производстве из общего каталога схем. При выборе материалов можно сразу задать их цену из соответствующей базы данных. Для используемых в производственном процессе машин имеются данные об их характеристиках и параметрах. Естественно, задания машинам для исполнения заказов выбираются или оператором, или автоматической системой управления.

Отдельные операции производства должны определяться не системным администратором, а по умолчанию при изготовлении конкретного типа продукции. Информация о заказах клиентов, повторно размещающих заказы, (параметры настройки машин и др.) сохраняется для дальнейшей работы. Из всей этой информации AMS рассчитывает производственные затраты, как предложение клиенту цены печатной продукции. После этого может быть отпечатан прайс-лист для оправки по почте, факсу или посредством электронной почты. После окончательного согласования заказа, клиенту отправляется подтверждение в его получении, заказ поступает в портфель заказов и вносятся установочные данные для оптимального формирования JDF - рабочего потока.

Отчетная калькуляция служит для фиксации отклонений в стоимости произведенной продукции. На базе этого, с одной стороны, оптимизируется предварительная калькуляция, а, с другой стороны, повышается эффективность производства. В большинстве случаев, отчетная калькуляция используется ограниченно, в основном, для выяснения фактически затраченного рабочего времени эксплуатации оборудования и материальных затрат. Основой калькуляции являются нормы временных затрат и издержки

отдельных технологических операций. Отчетная калькуляция предусматривает сбор данных со всех участков производства. Эти данные могут представляться сотрудниками через таблицу учета рабочего времени, который заполняется вручную, а потом вводится с терминала в программное обеспечение бухгалтерского учета. Вместо оформления таблицы рабочего времени на бумаге, сотрудники могут вводить эти данные через специальные BDE - терминалы (сбора производственных данных). Для этого уже давно имеются специально поставляемые системы, в большинстве случаев это дополнительные модули систем управления заказами. Однако настоящая цель JDF / JMF - систем генерация BDE - данных полностью автоматически. JMF имеет возможность посылать данные от многих устройств различных производителей с использованием стандартизированного протокола централизованного обмена данными в MIS. Такая система, работающая с использованием динамических данных, не только сокращает затраты на подготовку отчетных калькуляций, но и может более точно и успешно контролировать сбои, состояние производства и отслеживать ход выполнения работ.

JDF - формат не позволяет генерировать все данные конкретного производства автоматически. Часто используется в этом процессе компьютер со специальным программным обеспечением для узких мест, не имеющих JDF/JMF - интерфейса для приема и ввода производственных параметров в систему. Эти данные затем вводятся посредством JDF/JMF - интерфейса в MIS или другую систему обработки информации.

8.2 Использование JDF для процессов резки

JDF – интерфейс между системой управления заказами и производством является самым важным интерфейсом JDF - рабочего потока. Реализация этого интерфейса успешно решена, и он открыт в том смысле, что AMS и программное обеспечение оборудования различных производителей через JDF/JMF могут взаимодействовать. Это, однако, не реализуется всеми производителями оборудования и, в частности, оборудования для резки. Использование интерфейсов на основе формата JDF дополнительно рассмотрим в главе 9. Сейчас отметим, что в примерах типографий Y и Z, описанных в параграфах 2.2 и 2.3, уже были эскизно представлены связи на базе формата JDF между MIS, допечатной ступенью и заданиями для процессов резки.

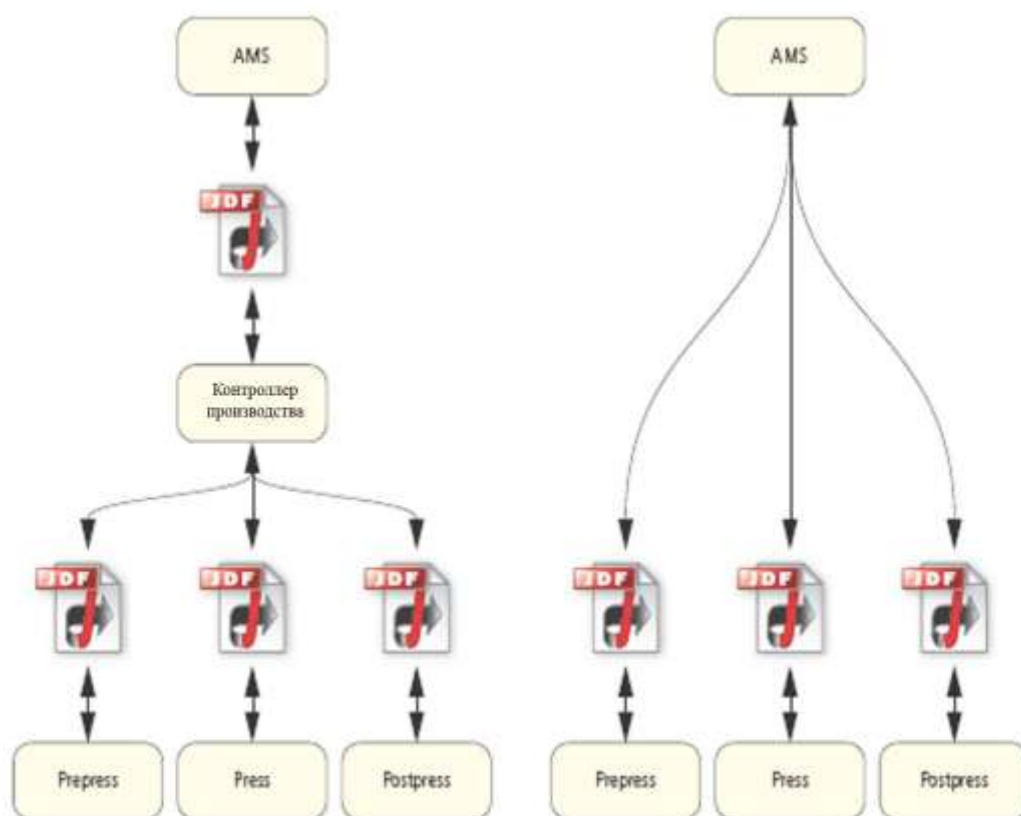


Рис.8.1 JDF между MIS и производством

Принципиально имеется две модели передачи производственных JDF - данных. Либо AMS передает JDF - данные центральному контроллеру JDF - производства, который со своей стороны, передает информацию в контроллеры и устройства в отделениях допечатной, печатной и послепечатной обработки, имея с ними двухстороннюю связь, либо данные передаются непосредственно из системы на устройства без использования промежуточного контроллера (рис. 8.1). Эти две схемы имеют много приложений, размывающих между ними границы. Так как в конфигурации слева расположено промежуточное устройство управления рабочим потоком в производстве, то оно осуществляет взаимодействие с подразделениями. В правой схеме MIS берет на себя выдачу заданий для устройств. В этом случае система управления должна оперировать, в том числе, техническими данными производственного процесса, например, параметры установки красочных зон, позиции ножа резальной машины или последовательность фальцев в соответствующие устройства - контроллеры оборудования печатного и других цехов и участков согласно требованиям производственной загрузки. Обе конфигурации, представленные на рис. 8.1, как уже говорилось, реализованы в полиграфическом производстве.

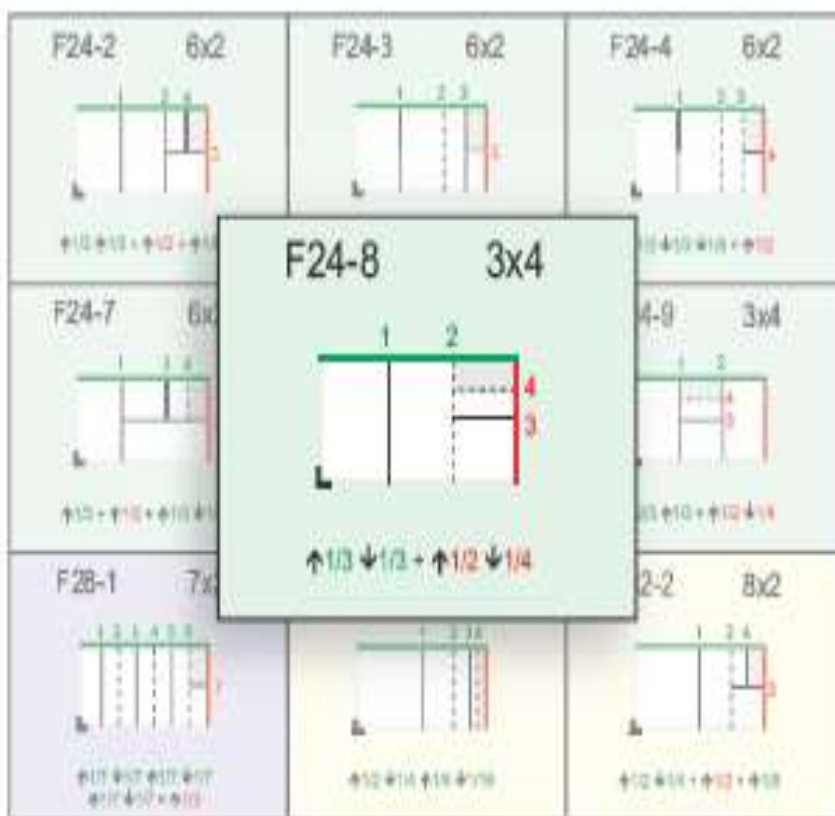


Рис.8.2 F 24-8 JDF - каталога видов фальцев

Далее на примерах более детально рассмотрим систему с центральным контроллером в JDF - рабочем потоке. Передача JDF - данных для машин отделочных процессов до сих пор находит малое применение, поэтому будем дополнительно обсуждать этот интерфейс для резальных машин в главе 11. Рассмотренный пример не будет абстрактным.

Возьмем 96-страничную, 4-красочную брошюру формата 12x12 см. Как обложка, так и внутренний блок, запечатываются с двух сторон на листе форматом 43x61 см. Тип фальца для внутреннего блока F24-8 в соответствии с JDF - каталогом видов фальцев с матрицей 3x4 выбран так, чтобы в совокупности получилось 4 тетради (4x24=96 страниц), как это показано на рис. 8.2. Обложка печатается на шести листах.

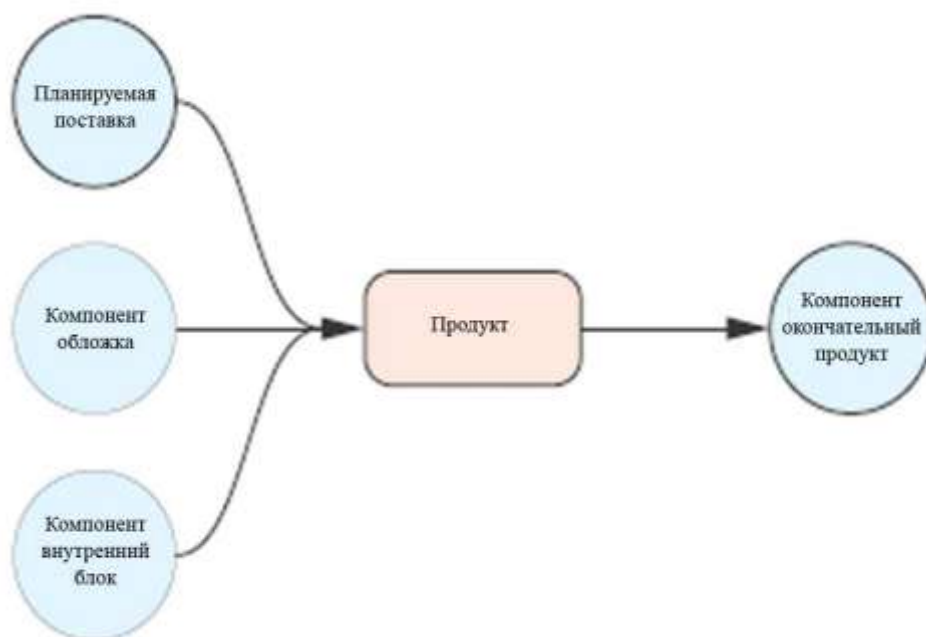


Рис.8.3 JDF – корневые ресурсы брошюры

Составляющие компоненты JDF - данных генерируется типовым способом системой AMS, что можно увидеть на рис.8.3. Основной продукт имеет два компонента ресурсов (*Component-Ressourcen*), а именно обложку и внутренний блок. Окончательным продуктом является выходной ресурс (*Output-Ressource*) типа компонент (*Component*).

Узловые пункты продукта, *GrauBoxen*, узлы процессов схематично представлены прямоугольниками на рис.8.4 и образуют древовидную структуру. Красные прямоугольники это продукт и его составляющие, желтые обозначают узлы процессов, серые – *GrayBoxes* – узлы группы процессов. В данном случае *GrauBox* является финишным узлом допечати, а печать (*ConventionalPrinting*) и фальцовка (*Folding*) соответственно четыре раза укладываются в процессы *GrayBoxes*.

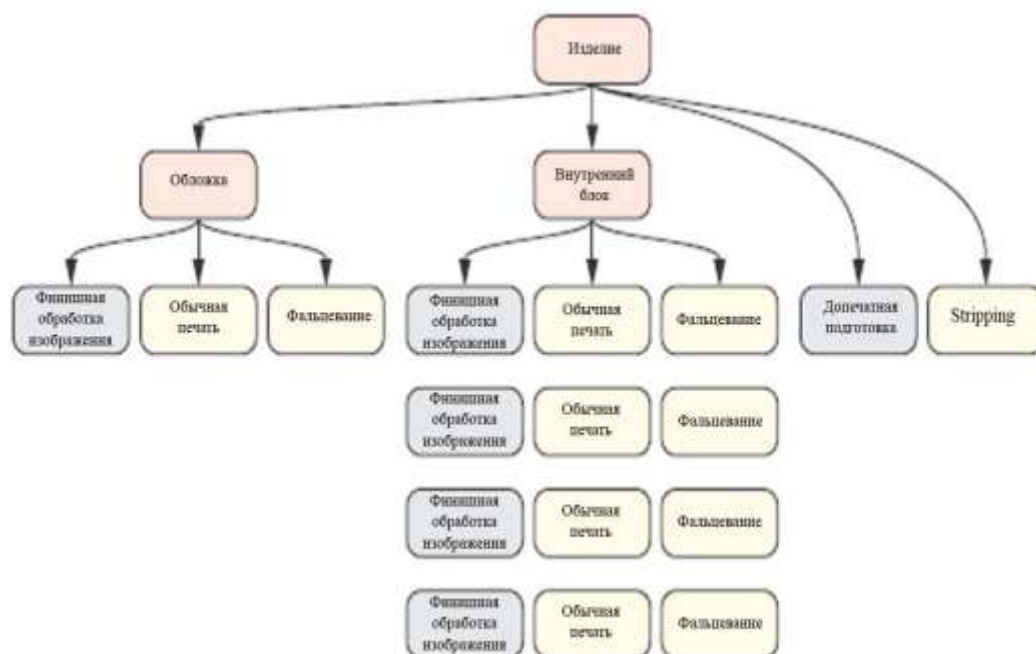


Рис. 8.4 Древоподобная структура JDF - узлов

Некоторые из представленных здесь процессов и групп процессов уже упоминались нами в материале по введению в JDF. Сейчас мы хотели бы их только кратко охарактеризовать прежде, чем будем их анализировать более глубоко:

- *PrepressPreparation*: узел *GrauBox*, который охватывают все рабочие шаги от обработки контента до спуска полос;
- *Stripping*: монтаж печатного листа;
- *FinalImaging*: узел *GrauBox*, который охватывают электронный спуск полос (*Imposition*), узел *GrauBox* растривания (*RIPping*), изготовление предварительного изображения (*PreviewGeneration*) и экспонирование формных пластин (*ImageSetting*);
- *ConventionalPrinting*: традиционная (аналоговая) офсетная печать с физических форм и увлажнением;
- *Folding*: фальцевание.



Рис. 8.5 Ресурс узла GrayBox допечатных процессов

Узел процессов группы *PrepressPreparation* очень просто структурируется, как это видно на рис.8.5. Ресурс ввода информации *RunList* определяет контент работы, переданной клиентом.

```
<RunList Class="Parameter" ID="_001"  
NPage="106" Status="Available" />
```

Рис. 8.6 Пример RunList для ввода PrepressPreparation

На момент приема заказа, зачастую, известно только количество страниц и нет еще окончательного названия и спецификации данных *контента (Content-Daten)*, которые в дальнейшем заполняются (рис.8.6). Ресурс вывода информации *RunList*, напротив, предоставляет для производства уже подготовленные страницы, которые нормализуются, трансформируются в цветоделенном пространстве (цветоделенные изображения) и должны быть готовы к использованию.

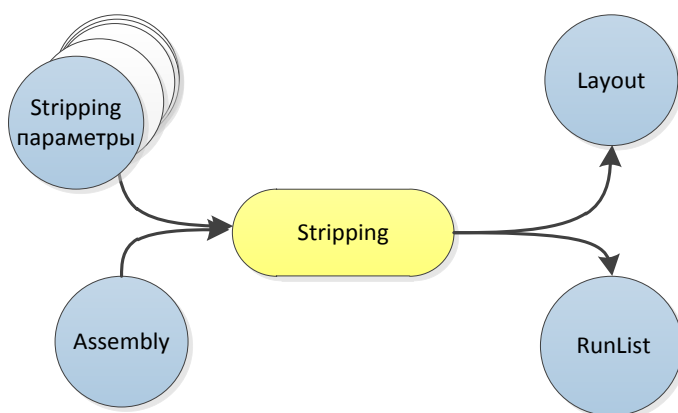


Рис. 8.7 JDF - процесс монтажа печатного листа

Stripping – процесс (скоростной ввод данных параллельными потоками) ресурсов ввода и вывода информации представлен на рис. 8.7. Для каждой сигнатуры ресурса ввода существует параметр *StrippingParams*, в нашем примере их 5. Он, в сущности, определяет позиции на печатном листе страниц тетради (на рис. 8.8. обозначены как фальцлисты) и схему фальцовки.

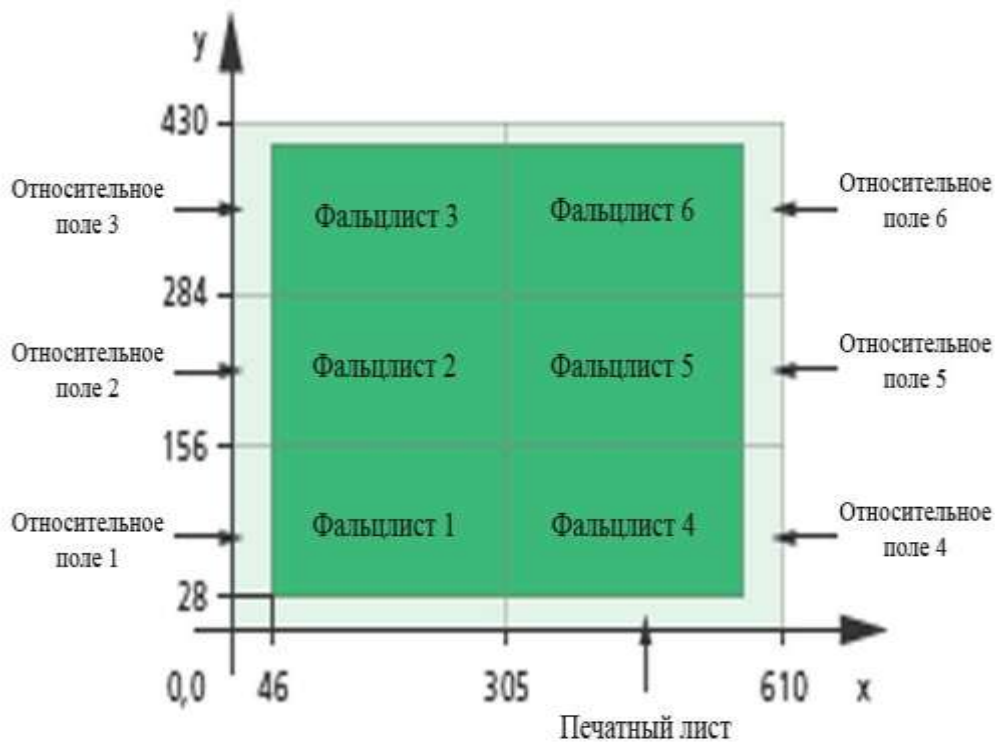


Рис. 8.8 StrippingParams содержат информации о позиции страниц на печатном листе

На рис. 8.8 отдельно изображены шесть страниц на печатном листе. Разобраться в указаниях поначалу сложно. Печатный лист делится на шесть прямоугольников, чьи размеры определены в атрибутах *RelativeBox* от 0 до 1, так как они описывают разные части печатного листа, причем первые два числа (x,y) – это величины нижнего левого угла, а следующие две цифры (x,y) – это величины правого верхнего угла. *RelativeBox* (координаты прямоугольника) содержит значения: 0.0 0.0 0.5 0.36279, они точно укладываются в нижнем левом углу. На печатном листе со стороной 430 мм. *RelativeBox* формирует первую страницу в Y - координате при значении 156 мм (0,36279 x 430 мм). В этом атрибуте *RelativeBox* расположен первый (1) лист (рис. 8.8). Аналогично можно вычислить другие прямоугольники. Итак, координаты страниц на листе по их порядку отсчитываются слева вверх с нулевой отметки и справа в X - направлении. Соответствующие записи из фрагмента описания приведены на рис. 8.9.


```

<StrippingParams Class="Parameter" ID="_200" PartIDKeys="SignatureName
SheetName" PartUsage="Explicit" Status="Available">
  <StrippingParams SignatureName="SIG001">
    <StrippingParams SheetName="Umschlag" WorkStyle="WorkAndBack">
      <BinderySignatureRef rRef="_201" />
      <Position MarginBottom="79.370" MarginLeft="130.39"
MarginRight="0.0" MarginTop="0.0" Orientation="Rotate0"
RelativeBox="0.0 0.0 0.5 0.362" />
      <Position MarginBottom="0.0" MarginLeft="130.39" MarginRight="0.0"
MarginTop="0.0" Orientation="Rotate0"
RelativeBox="0.0 0.362 0.5 0.660" />
      <Position MarginBottom="0.0" MarginLeft="130.393" MarginRight="0.0"
MarginTop="51.023" Orientation="Rotate0"
RelativeBox="0.0 0.660 0.5 1.0" />
      <Position MarginBottom="79.37" MarginLeft="0.0"
MarginRight="130.393" MarginTop="0.0" Orientation="Rotate0"
RelativeBox="0.5 0.0 1.0 0.3627" />
      <Position MarginBottom="0.0" MarginLeft="0.0"
MarginRight="130.393" MarginTop="0.0"
Orientation="Rotate0" RelativeBox="0.5 0.362 1.0 0.660" />
      <Position MarginBottom="0.0" MarginLeft="0.0" MarginRight="130.393"
MarginTop="51.023" Orientation="Rotate0"
RelativeBox="0.5 0.660 1.0 1.0" />
      <StripCellParams BleedFace="7.086" BleedFoot="7.086"
BleedHead="7.086" BleedSpine="0.0" Spine="0.0" TrimFace="11.338"
TrimFoot="11.338" TrimHead="11.338" TrimSize="340.157 340.157" />
      <MediaRef rRef="_202">
        <Part SheetName="Umschlag" SignatureName="SIG001" />
      </MediaRef>
      <MediaRef rRef="_203">
        <Part SheetName="Umschlag" SignatureName="SIG001" />
      </MediaRef>
      <DeviceRef rRef="_204" />
    </StrippingParams>
  </StrippingParams>
  ...
</StrippingParams>

```

Рис. 8.9 StrippingParams содержат описание параметров монтажа страниц на листе

Расположение страниц (прямоугольников, единиц), находящихся в *RelativeBox*, определяется значениями краев: верхнего, нижнего, левого, правого (*MarginBottom*, *MarginLeft*, *Margin Right* и *MarginTop*), указанных в DTP - пунктах. На рис. 8.10 еще раз это продемонстрировано. Ресурс *StrippingParams*, впрочем, хороший пример одного разделенного на части ресурса (по меньшей мере, ценнее, чем в параграфе 6.14), который после ключа *PartIDKeys* - ключ *SignatureName SheetName* будет разделен.

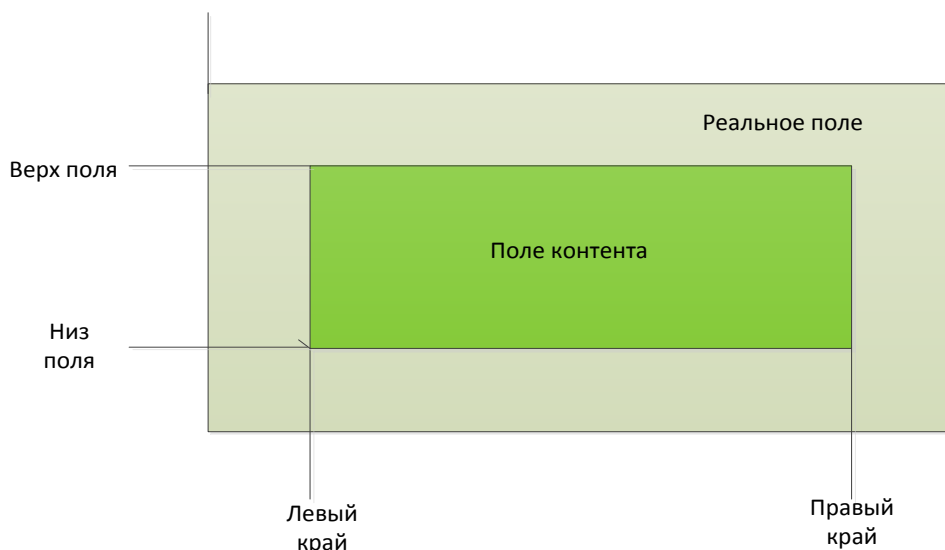


Рис.8.10 Заданные позиции страницы

В JDF - коде на рис. 8.9, кроме того, имеется несколько записей, о которых следует сказать. В элементе *StripCellParams* содержатся значения для полей за обрез, а также окончательный формат промежуточного продукта. Две ссылки на медиа-ресурсы раскрывают, с одной стороны, запечатываемый материал, а с другой - применяемые формные пластины.

Наконец, в заключение, существует еще ресурс *BinderySignature* (*сигнатура - порядковый номер печатного листа*), упомянутый ранее и наглядно представленный в записи на рис. 8.11. В фрагменте записи содержится схема спуска полос. В этом случае загрузка данных в систему осуществляется просто из каталога видов фальцев. Альтернативно схема спуска полос может быть записана непосредственно в подэлементах (*SignatureCell*) от *BinderySignature*. Ресурс *BinderySignature* соответствует одной обычной фальц - структуре, который может быть использован при различных видах фальцовки независимо от размеров страниц и печатных листов.

```
<BinderySignature Class="Parameter" DescriptiveName="F04-01_ui_2x1"
FoldCatalog="F4-1" ID="_201" NumberUp="2 1" Status="Available" />
```

Рис. 8.11 Фрагмент ресурса BinderySignature

Отметим ресурс *Assembly*, представленный один раз на рис. 8.7, который содержится в JDF - документах. С использованием версии JDF 1.2 указанный ресурс может

присутствовать в собранных заданиях для процессов совместной передачи или комплектовки тетрадей.

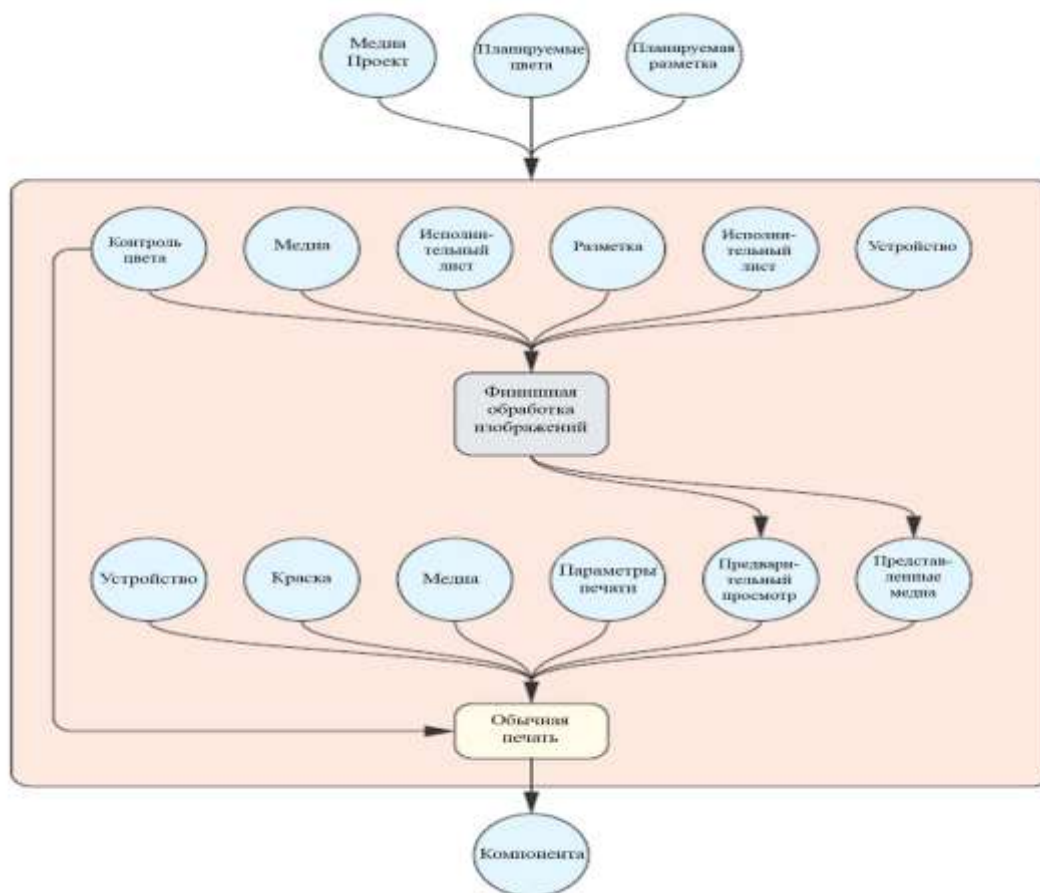


Рис. 8.12 Схема процессов производства основного продукта (блок) и обложки с ресурсами

На рис.8.12 продемонстрирована схема группы узлов (*ProductIntent*) с одной *GrayBox* и процессами их подключения к ресурсам. Четыре ресурса вне красного прямоугольника указывают на ввод и вывод ресурсов узлов *ProductIntent*. В качестве ввода служат ресурсы сверху, которые описывают конечную продукцию. В *LayouIntent* указывается *величина конечного формата*, в *ColorIntent* – *красочность*, *MediaIntent* – *запечатываемый материал*. Выводной ресурс *компонент* (*Component*) представляет продукт – обложка. Этот ресурс является вводным ресурсом последующих процессов, определяемых узлами *ProductIntent*, которые и определяют окончательный продукт.

GrayBox и процесс связаны рядом ресурсов. Ресурсы, безусловно, должны быть расположены не только, как показано на рис.8.12, в пределах имеющихся *ProductIntent* - узлов для обложки, но также могут лежать в плоскости выше корневого элемента, а значит *ProductIntent* - узлах итогового продукта. Стрелки между ресурсами и *GrayBox*, собственно говоря, дают только представление о связях (*ResourceLinks*).

На первый взгляд, может показаться, что рис. 8.12 является ошибочным, так как ресурс ввода информации *RunList* для *GrayBox* изображен дважды. На самом деле, это задумано специально, так как содержание ресурсов отличается. В отдельности *GrayBox* имеет следующие ресурсы ввода:

- *Device*: характеристики экспонирующего устройства;
- *RunList*: данные, которые содержат подготовленные данные контента, как вывод от *PrePressPreparation*;
- *Layout*: сигнатуры совокупного продукта;
- *RunList*: файл, который содержит метки для печатных листов;
- *Media*: описание печатных форм;
- *ColorantControl* красочность отдельных сигнатур.

Отдельные ресурсы описывают не только составную часть продукта - обложку, а могут описывать и весь продукт. Ресурсы (*Layout*, *RunList*, *ColorantControl*) должны использоваться в JDF-элементах совокупного продукта, чтобы ресурсы частичного продукта могли взаимодействовать с ними. Из ресурсов, представленных на рис.8.12 вверху, только два одновременно имеют статус *Available: Device* и *ColorantControl* (*Available - доступный*). Другие ресурсы, которые уже формально размещены, не загружены еще некоторыми важными элементами. Поэтому для них установлен статус *Unavailable* (*отсутствуют*). AMS еще не знает, к примеру, не только имя файла, но и место хранения данных контента, а также не может обеспечить запись характеристик пластин для печатных форм, использующихся в производстве.

GrayBox «*Imposition RIPping PreviewGeneration ImageSetting*» в отдельности не выводится. Устанавливаются только выводные ресурсы *GrayBox*, а именно *ExposedMedia* и *Preview*, которые используются как вводные ресурсы для следующего процесса. Статусы этих двух ресурсов должны быть также *Unavailable* (*отсутствуют*), так как ресурс *GrayBox* еще не запущен, точнее не преобразовывался в процессы и не может с их стороны выводиться. Как мы отмечали в 6 главе, *GrayBox* принципиально не выполнимы.

Следующий процесс *ConventionalPrinting* (*традиционная печать*) имеет в качестве входного ресурса:

- *Exposed Media*: печатные формы для этой работы;
- *Preview*: изображение для предварительного просмотра цветоделения (для расчета установок красочных зон);
- *ConventionalPrintingParams*: указания типа печатной машины (листовая, рулонная) или с переворотом;
- *Media*: указания запечатываемого материала;
- *Ink*: указания печатной краски;
- *Device*: указания для печатной машины;
- *ColorantControl*: указания по красочности для каждой сигнатуры.

```

<JDF DescriptiveName="Umschlag" Status="Waiting" Type="Product"...>
  <ResourceLinkPool>
    ...
  </ResourceLinkPool>
  <ResourcePool>
    ...
  </ResourcePool>

  <JDF Category="FinalImaging" DescriptiveName="Umschlag (CTP) "
    Status="Waiting" Type="ProcessGroup"
    Types="Imposition RIPing PreviewGeneration ImageSetting"...>
    <ResourceLinkPool>
      ...
    </ResourceLinkPool>
    <ResourcePool>
      ...
    </ResourcePool>
  </JDF>

  <JDF DescriptiveName="Umschlag" Status="Waiting"
    Type="ConventionalPrinting"...>
    <ResourceLinkPool>
      ...
    </ResourceLinkPool>
    <ResourcePool>
      ...
    </ResourcePool>
  </JDF>
</JDF>

```

Рис. 8.13 Структура элементов записей частичного продукта - обложка

В этом случае все ресурсы вплоть до *Ink*, *Device* и *ColorantControl* имеют одинаковый статус *Unavailable* (отсутствует). Структура элементов JDF - узлов частичного продукта - обложка изображена на рис. 8.13. В примере отсутствуют записи содержания ресурсов и ссылки на них. Посмотрите, тем не менее, на схему рис. 8.12. Из него становится ясно, что ресурс *InkPool* узла продукта – обложка должен выглядеть принципиально в записи, как на рис. 8.14.

```

<ResourceLinkPool>
  <ComponentLink Usage="Output" rRef="_100" />
  <LayoutIntentLink Usage="Input" rRef="_101" />
  <ColorIntentLink Usage="Input" rRef="_102" />
  <MediaIntentLink Usage="Input" rRef="_103" />
</ResourceLinkPool>

```

Рис.8.14 Ресурс LinkPool JDF-узлового продукта «Обложка»

Другие ресурсы *LinkPools* формируются аналогично. Только сами ресурсы, как составляющая часть, менее эффективны. Так, ресурс *Ink*, представленный на рис.8.15 раскрывает запись, определяющую применяемые печатные краски для каждой сигнатуры. Здесь речь идет о ресурсе, вывод которого обозначен посредством значения атрибута *PartIDKeys*.

```

<Ink Class="Consumable" ID="_104" PartIDKeys="SignatureName SheetName Side
Separation" PartUsage="Implicit" Status="Available">
  <Ink SignatureName="SIG001">
    <Ink SheetName="Umschlag">
      <Ink Side="Front">
        <Ink Separation="Cyan" />
        <Ink Separation="Magenta" />
        <Ink Separation="Yellow" />
        <Ink Separation="Black" />
      </Ink>
      <Ink Side="Back">
        <Ink Separation="Black" />
      </Ink>
    </Ink>
  </Ink>
</Ink>

```

Аналог для дальнейших 4 сигнатур внутренней части

```

</Ink>

```

Рис. 8.15 Фрагмент записи для цветовых значений отдельной сигнатуры

Само собой разумеется, что не все основные ресурсы продукта – обложка могут здесь представляться и не должны поступать дальше на JDF - узлы *Innenteil*, они становятся некоторым аналогом для записей продукта - обложка. Мы должны согласиться, что дальнейшие ресурсы JDF корневого элемента могут исчезнуть, как, например, ресурс *CustomerInfo* (*Customer- управление*), который содержит данные клиентов. Это выглядит так же, как было представлено на рис. 5.2 и 5.3. Не будем еще раз демонстрировать код и на этом закончим наш пример.

Мы хотели бы выбрать другой путь, а именно, абстрактное определение ресурса управления *CustomerInfo*, объяснить каким образом он воспроизводится в JDF - спецификации 1.4. Это должно облегчить читателю знакомство со спецификацией. Таблица, представленная на рис. 8.16, касается некоторых описаний ресурсов в спецификации.

Имя	Тип Данных	Описание
<i>Код тарификации?</i>	строка	Код тарификации данных, при выполнении Узла.
<i>Потребительский ID?</i>	строка	Потребительская идентификация, используемая приложением, которое создало Заказ. Это обычно внутреннее потребительское число системы MIS, которая создала Заказ.
<i>Потребительское имя задания ?</i>	строка	Идентификатор, для потребительского использования, относящийся к Заказу.
<i>ID идентификатор потребительского Заказа?</i>	строка	Внутренний порядковый номер в системе клиента. Это идентификатор, когда заказ размещен и затем отослан в порядке подтверждения или в виде счета.
<i>Потребительский идентификатор проекта?</i> <i>Новый в JDF 1.2</i>	строка	Внутренний идентификатор проекта в системе клиента. Это число могло бы быть если, когда заказ размещен и затем сослан на подтверждении заказа или счет.
<i>rRefs?</i> <i>Находящийся в JDF 1.2</i>	IDREFS	Массив удостоверений личности любых Элементов, которые определены как Элементы ресурса <i>Ref</i> . В версии 1.1 это был IDREF <i>ContactRef</i> . В JDF 1.2 и выше, до реализации, чтобы поддержать ссылки.
<i>Компания?</i> <i>Находящийся в JDF 1.2</i>	ref-элемент	Элемент ресурса, описывающий бизнес или организацию контакта. В JDF 1.1 и выше, присоединение <i>Компании</i> <i>Контактов</i> определено в ресурсе <i>Контакты</i> .
<i>Контакт*</i> <i>Новый в JDF 1.2</i>	ref-элемент	Контакты описания Элемента ресурса связанные с клиентом. Там должен быть один <i>Контакт</i> [содержащий (@ <i>ContactTypes</i> , " <i>Клиент</i> ")]. Такой <i>Контакт</i> определяет имя основного клиента, адрес и т.д.
<i>Потребительское сообщение *</i> <i>Новый в JDF 1.2</i>		

Рис. 8.16 Определение ресурсов управления - CustomerInfo в JDF - спецификации

В левом столбце стоят имена структурных элементов (как атрибут, определяющий свойства подэлементов или ссылку на ресурс), средний столбец отображает тип данных, а в правом столбце даются разъяснение каждому структурному элементу. Бросаются в глаза в первом столбце комментарии, выделенные голубым цветом, как *New в JDF 1.2* или

Deprecated в JDF 1.1 (*устаревший*). Последнее означает, что соответствующий структурный элемент передавался в версии 1.1, и никакой JDF - код не может содержать данные в версии выше 1.1. По причинам совместимости это должно идти естественным путем, код должен находиться в таблице спецификаций, однако мы не хотим больше использовать такие «устаревшие» понятия. Далее определяется специальный символ *как по именам ? или**. Иногда (в другом разряде спецификации) стоит также + или нет вообще никакого специального символа (рис. 12.8). Значение относится к количеству возможных экземпляров:

? – опционально;

* - никогда или неоднократно;

+ - однажды или много раз.

Если у имени нет специального знака, то это структурный элемент обязательный и должен точно попасть в атрибут *CustomerID* хотя бы раз и не должен встречаться в подэлементе *Contact* один или много раз.

Тип данных *String* (*строка*) – это просто строка символов, *Refelement* – это номер элемента или временная ссылка на элемент (разумеется, это отсутствует на рис. 8.16) должна непосредственно стоять.

Приведем краткое объяснение структурных элементов, которые также актуальны и для версии 1.4:

- *BillingCode*: номер счета для работы;
- *CustomerID*: номер клиента MIS;
- *CustomerJobName*: имя клиента;
- *CustomerOrderID*: номер заказа клиента;
- *CustomerProjectID*: проектный номер клиента;
- *Contact*: ссылка на элемент, который описывает контакт;
- *CustomerMessage* описание сообщений клиенту, к примеру, автоматическая

отправка электронной почты.

Для элементов *Contact* и *CustomerMessage* в спецификации имеются собственные таблицы. Так как оба элемента имеют следующие подэлементы (*Address*, *ComChannel*, *Company* и *Person*), то они специфицированы в других таблицах. Ссылки на них нами не вводились, чтобы не запутать читателя (хотя это может произойти), но и потому, что используются подэлементы от разных родительских элементов, при такой технике они могут находиться в спецификации только один раз. К примеру, если элемент *ComChannel* (*модель канала*) является подэлементом *Contact* (*контакт*), также как и от элемента

CustomerMessage (сообщение управления). Фактически читатель не должен следовать только по ссылкам вниз (к возможным дочерним элементам), но и некоторым образом вверх, т.к. имеются так называемые «абстрактные ресурсы», описываемые атрибутами, которые могут иметь все ресурсы. Эти общие атрибуты (как например, *ID*, *Class* или *Author*) не будут больше представлены в таблицах конкретных ресурсов. Похожее происходит и в JDF-узлах.

До сих пор мы обсуждали только поток информационных данных от MIS к допечатной ступени, а не противоположное направление, для которых решающей является отчетная калькуляция. У этого пути два возможных варианта, существующих параллельно:

- MIS принимает JDF - сообщения от системы производства;
- производственная система изменяет JDF - сообщения и транслирует обратно в

MIS.

При обратном получении JDF - данных от системы производства, прежде всего, собираются в *AuditPools* (*нуле аудита*), как это изложено в главе 6.

8.3 Документы MIS ICS

Значение интерфейса между системами управления заказами/информационно-управляющими системами (AMS/MIS) и производством очень важно, его определяют ICS – документы спецификаций (параграф 6.7). В настоящее время спецификации ICS актуальны в версии 1.3, правда, они относительно недолговечны и рекомендуется читателям следить за появлением новой версии в Интернете [12]. Отметим их для информационно – управляющих систем (MIS):

- *MIS ICS – общая спецификация MIS;*
- *MIS to Prepress ICS – спецификация MIS допечатного производства;*
- *MIS to Conventional Printing-Sheet-Fed ICS – спецификация MIS традиционной листовой печати*
- *Newspaper: MIS to WebPress ICS - новая спецификация MIS для распределенной печати с использованием Интернета;*
- *MIS to Finishing ICS – спецификация MIS отделочных процессов.*

Кратко рассмотрим первый документ - *MIS ICS* - спецификации, остальные будут представлены в следующих главах с 9 по 11. Документ *MIS ICS* содержит общие требования к информационно-управляющей системе (MIS), которые не являются специфическими для этапов допечати, печати или дальнейшей обработки оттисков. В *MIS ICS*, похоже, как и в *JMS ICS* определяются различные совпадающие ступени. Здесь

существует три уровня, которые вторично относятся к JMF -сообщениям, но мы будем представлять только два первых уровня (изображенных на рис. 8.17) :

- Уровень 1: MIS передает JMF - сообщения через HotFolder (горячую папку) на *Worker(исполнитель)* (рис. 8.17 Уровень1);
- Уровень 2: Система MIS может дополнительно генерировать JMF - запросы с помощью *NodeInfo* - элемента, чтобы установить фиксированный коммуникационный канал от *Worker* к *MIS* и обратно (рис. 8.17 Уровень 2a). Этот способ соответствует Уровню 1 в JMS ICS. Исполнитель (*Worker*) не должен быть в этом случае *HTTP* - сервером и не может принимать и обрабатывать пакеты *HTTP*.

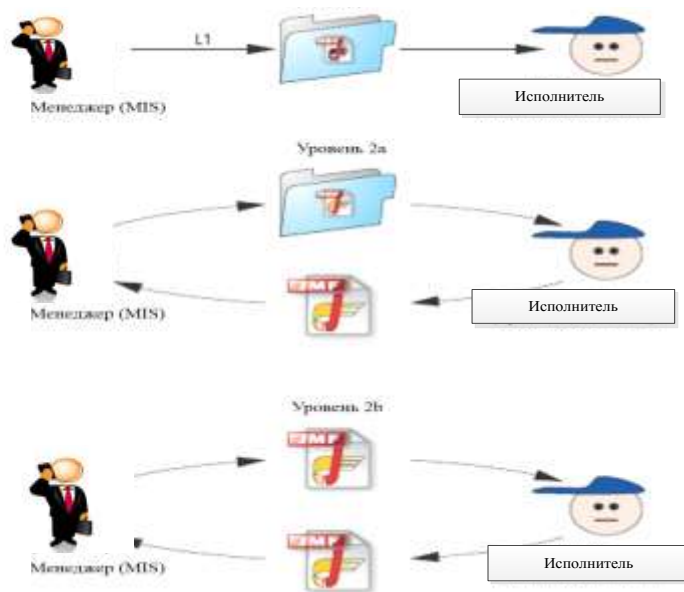


Рис. 8.17 Уровень 1 и 2 в MIS ICS

Альтернатива представлена на иллюстрации - Уровень 2 (рис. 8.17 Уровень 2b). MIS напрямую пересылает JDF - сообщения. При этой опции исполнитель (*Worker*) должен быть *HTTP* - сервером, что и предполагает Уровень 2 от JMF ICS



Рис. 8.18 Исполнитель должен давать ответ на запрос менеджера

Повышаются и другие требования к JMF. Например, когда информационно-управляющая система (MIS) отправляет запрос - ресурс исполнителю (*Worker*) по настройке, эта информация о ресурсах, которые ранее использовались или создавались, пересылается обратно (рис. 8.18). Программное обеспечение MIS, которое удовлетворяет *Уровню 2 MIS ICS*, должно осуществить соответствующие запросы (через *NodeInfo* или непосредственно через JMF - сообщения). *MIS ICS Уровень 2 – Worker (исполнитель)* должен принять этот запрос, интерпретировать и ответить. Подобное относится и к запросам относительно состояния устройств и *Jobstat*. Другими словами: отслеживание работ и контроль состояния устройств предусмотрено *MIS ICS Уровня 2*, а так же MIS – программным обеспечением для контроллеров и устройств, используемых на производстве. На практике существуют интерпретации JDF - рабочий поток, которые эти функции не поддерживает.

Однако определяются не только требования к JDF - элементам в *MIS ICS*, а также ко многим другим в JDF - формате. Должно определяться и много маленьких деталей. Так, например, предписывается, что в каждом корневом JDF - элементе, созданном MIS, вводится специальный атрибут (*ICSVersions*) *ICS Уровня*, после этого формируется JDF - документ. Далее каждый корневой JDF - элемент должен иметь два специальных ресурса, как ввод, а именно *ресурс CustomerInfo*, в котором хранятся данные *клиентов* и ресурс *NodeInfo*, который содержит данные *календарного планирования* и опционально одно или несколько JDF - сообщений. Каждый производимый MIS корневой JDF - узел, является продукт - узлом, и характеристики продуктов хранятся в *Intent* - ресурсе. Для этого каждый JDF корневой элемент должен иметь, по крайней мере, как ввод *Intent* - ресурс. Если бы не было этих ресурсов, были бы продукты без свойств, а это довольно бессмысленно. Каждый корневой JDF - элемент имеет выводной ресурс *Component*, который представляет конечную продукцию, о чем мы ранее уже говорили.

Каждый исполнитель (*Worker*) должен содержать в JDF – процесс - узле информацию о начале, окончании, продолжительности и тому подобных процессах в *Audit*-элементах. В *MIS ICS Уровня 2* должны содержаться еще последующие *элементы Audit*-ресурсов, вносимые в *AuditPool*. Они содержат сведения от исполнителя (*Worker*) об израсходованных материалах при выполнении производственного процесса (печатных формах, бумаге, краске).

8.4 Print Talk и JDF - интерфейс клиентов

Спецификация Print Talk названа по имени одноименной организации, созданной американской ассоциацией NPES (Association for Suppliers of Printing, Publishing and Converting Technologies). В 2004 году Print Talk интегрировалась в консорциум CIP 4 и была определена как независимая организация. Print Talk- спецификация основывается на языке cXML, что уже обсуждалось в разделе 5.4.

Некоторые системы управления заказами (AMS) могут считывать JDF -документы, посланные клиентом, и из них формировать новый ассортимент или заказ. Клиент, например, может сформировать JDF-документ в программе Acrobat, которая описывает продукт (продукт – узел). Этот документ затем импортируется в систему управления заказами (AMS). При этом AMS получает необходимое описание продукта извне, не раскрывая дальнейшие коммерческие пути. Однако у этого способа не прямое решение. Job - идентификатор размещается клиентом и не вызывает сомнений в области указанных номеров у системы - MIS типографии. В этом есть смысл, если типография предварительно принимает заказ, размещая файл JDF-данных в системе - MIS без большого объема контента, она посредством электронной почты рассылает клиентам информацию, которая потом ими считывается в программе Acrobat, и при этом клиент может дополнить данные заказа. Наконец, клиент может послать эти расширенные JDF - данные обратно в типографию, которая автоматически приводит в порядок информацию о заказе.

Не только конечные пользователи, но и уполномоченные агентства по приему заказов могут сообщать параметры заказа через JDF - сообщения в типографию и обмениваться ими. При этом агентства имеют возможность получать определенные параметры (например, срок поставки, объем тиража) автоматически из системы клиента, дополнять их и передавать в типографию.

Сценарий по обмену информацией при этом реализуется в электронном виде. С помощью формата JDF реализуются коммуникации между клиентами и типографией. Print Talk идет в этом вопросе на один шаг дальше. Принцип наглядно объясняется на рис. 8.19.



Рис. 8.19 Print Talk – коммуникации между клиентом и типографией

Деловые вопросы, такие как запросы ассортимента или предложения о заказе поступают через Интернет и позволяют общаться сотрудникам типографии с клиентами. Заказы, их подтверждение в получении, документы поставок и счета могут пересылаться подобным образом. Посредством интеграции Print Talk – интерфейса в MIS - системы во многом упрощают административные шаги. В принципе, это ведет к решениям по открытию портала под лозунгом Web-to-Print.

Принципиально имеются две конфигурации решений: либо клиент обменивается сообщениями непосредственно с сервером локальной сети типографии, который предлагает ему такую услугу или же имеется посредник между ними. Во втором случае имеется Web - брокер, который получает ценовые предложения через протоколы Print Talk от различных типографий. Затем по его запросу они формируют предложения клиентам, полностью автоматизируя их отправку в режиме «реального времени» или под контролем сотрудника с отсрочкой по времени, посылая их через протокол Print Talk Web - брокеру. Брокер, со своей стороны, фильтрует предложения и по-новому подготавливает их для передачи в режиме online клиентам. Этот опыт сегодня достаточно широко распространен, например, в туристической отрасли. Автоматизированные запросы цен, естественно, обостряют еще более конкурентную борьбу между производителями в ценовой области. Если коммуникации между клиентами и типографией проходят традиционным образом, то техническая их реализация может осуществляться по-разному. Либо клиент в программе просмотра готовит Print Talk - документы и отправляет их на сервер типографии, или проводится Интернет-общение типографии с клиентом. В этом случае сначала Web-to-Print - сервер типографии генерирует Print Talk – сообщение. В последнем случае Print Talk - протокол выступает только как некий посредник типографии и представляет услуги интерфейсов между программным обеспечением Web-to-Print и MIS. Эта ситуация изображена на рис. 8.20

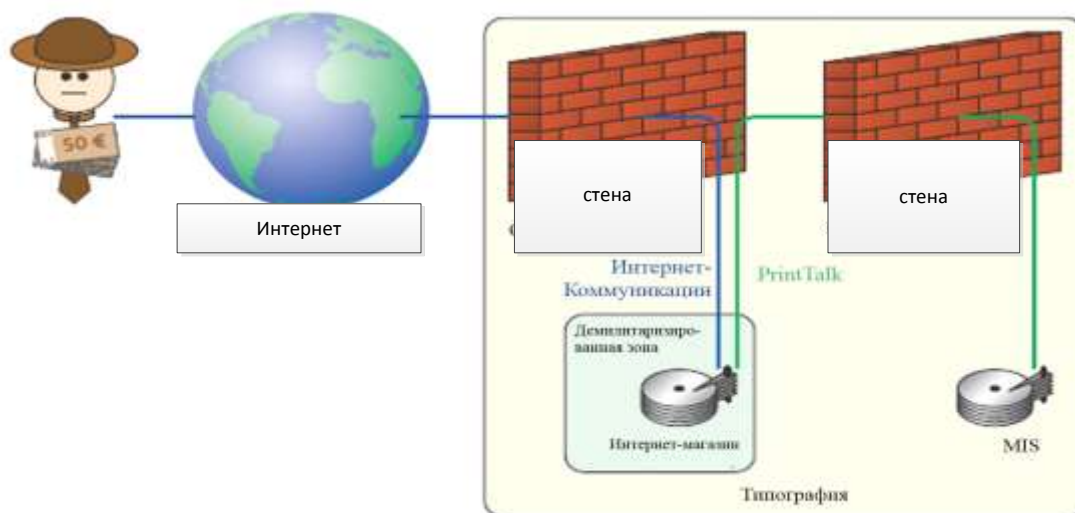


Рис. 8.20 Web - сервер и MIS типографии. Print Talk – протокол, применяемый в обмене информацией

Естественно, сервер MIS должен находиться в пределах закрытой локальной информационной сети типографии. Сервер, который устанавливается для реализации задач Интернет - магазина, как правило, находится в открытом доступе. Потенциальный клиент типографии обменивается сообщениями с системой заказа посредством Интернета в online – режиме. Данные передаются в MIS на основе Print Talk - протокола. При этом могут заказываться различные продукты с помощью HTML и, тем не менее, MIS определяет через Print Talk *BusinessObject*-элементы, а также описание JDF – продукта. Эта конфигурация способствует независимому интерфейсу между Web-порталом и MIS в то время, пока поддерживаются Print Talk и JDF.

С помощью Print Talk могут создаваться Business-to-Business-(B2B)- коммуникации между Web-брокером и типографией, а также Business-to-Consumer-(B2C)-интерфейс между клиентом и типографией. В настоящее время использование Print Talk незначительно, в будущем планируется его более широкое применение. Возможно, причиной нерешительного внедрения этого дополнения является то, что MIS - поставщики зачастую предлагают решения с использованием своих Web - порталов и поэтому заинтересованы только в открытых интерфейсах по отношению к информационно – управляющей системе.

```

<PrintTalk...>
  <Header>
    <From>
      ..
    </From>
    <To>
      ..
    </To>
    <Sender>
      ..
    </Sender>
  </Header>
  <Request>
    <purchaseorder...>
      <jdf:JDF...>
        <jdf...>
          </jdf...>
        </jdf...>
      </jdf...>
    </purchaseorder>
  </Request>
</PrintTalk>

```

Рис.8.21 Структура Print Talk –документа

На рис.8.21 представлена структура Print Talk – документа, очень похожая на сXML - структуру, которую мы представляли на рис. 5.7. Заголовок в Print Talk – документе формируется абсолютно идентично, как и в сXML. В Print Talk имеется Request-элемент (запрос), однако его подэлементы выглядят иначе: содержат, так называемый, *BusinessObject*. В данном примере это заказ, а в совокупности имеется 12 различных *BusinessObject*:

- *RFQ* (Request For Quote): требования предложения;
- *Quotation*: предложение;
- *PurchaseOrder*: выдача заказа;
- *Confirmation*: подтверждение получения заказа;
- *Cancellation*: отмена *BusinessObject*;
- *Refusal*: отклонение *BusinessObject*;
- *OrderStatusRequest*: запрос к статусу заказа;
- *OrderStatusResponse*: ответ к статусу заказа;
- *ProofApprovalRequest*: требование к подтверждению отказа;
- *ProofApprovalResponse*: подтверждение отказа;
- *Invoice*: счет;
- *ReturnJob*: типография возвращает работу клиенту.

```

<PurchaseOrder AgentID="CC" AgentDisplayName="CarlCool"
RequestDate="2008-11-13T11:00Z" BusinessID="A001" Currency="EUR"
Expires="2008-12-13T11:00Z ">
  <Pricing>
    <Price LineID="_1" DescriptiveName="Hard Cover Books" Amount="6800"
      Price="15000.00" />
    <Price LineID="_2" DescriptiveName="Shipping" Price="980.00"/>
  </Pricing>
  <jdf:JDF...>
    <JDF...>
      ...
    </JDF>
  </jdf:JDF>
</PurchaseOrder>

```

Рис. 8.22 Распределение поручений через протоколы Print Talk

Каждый *BusinessObject* имеет атрибуты, которые точно определяют коммерческую транзакцию. В заказе определяется, к примеру, валюта, форма оплаты и уровень цен. В примере записи на рис. 8.22 мы имеем даже две цены: одна для расчета себестоимости печатной продукции, другая для поставки. В более ранних версиях на основе JDF - формата эти задания с обозначением цен и способа оплаты отображались в ресурсе *DeliveryIntent*. Тем временем, все эти сообщения появлялись в JDF - формате и пересылались в Print Talk - элементы.

Отдельные *BusinessObject* содержат со своей стороны один или несколько JDF -узлов. Элемент заказа на рис.8.22 содержит собственные три JDF - узла, причем воспроизводится только верхний узел. В JDF - элементе обозначается и сохраняется описание печатного продукта.

Частичные продукты заказа специфицируются в дальнейшем в JDF - узлах. К примеру, в ресурсе *Pool* JDF - корневого узла информация о клиенте (*CustomerInfo*), о предстоящей поставке конечного продукции (*DeliveryIntent*) стоит как ввод. Мы хотим здесь этот пример не рассматривать далее, так как он был сформулирован по правилам JDF - формата, которые описаны в последних параграфах в главе 6.

Задания для самостоятельной работы:

- установите с помощью программы Acrobat (version 7.0) файл JDF. Сообщите сами клиенту данные о материале и продукции (например, брошюра 60 страниц, СМΥК-А4 внутренний блок и обложка СМΥК+особый цвет);
- загрузите от CIP-4.org JDF – редактор (сверху вниз), откройте полученные JDF данные и проанализируйте структуру;
- переименуйте Ваш файл *.jdf в *.xml и откройте их с помощью программы просмотра;
- исследуйте файл JDF и идентифицируйте переданные сообщения стоимости брошюры в программе Acrobat.

Глава 9. Стадия допечатной подготовки

С точки зрения допечатной стадии мир JDF/JMF выглядит так, как представлено на рис. 9.1, где каждая стрелка соответствует потенциальной связи JDF/JMF. При традиционной системе организации (менеджмента) допечатного производственного потока исходные данные в JDF - формате вводятся отделом обработки заказа (стрелка 1). В зависимости от конфигурации, данные могут также передаваться от допечатной стадии в систему управления заказами (AMS). Мы уже видели, что данные в JDF - формате, которые вводятся в информационно-управляющую систему (MIS), не являются достаточными для производственного потока (исходный пункт: блок (*GrayBox*-“черный ящик”)), это значит, что стадия допечатной подготовки в разных модулях должна иметь новые JDF - структуры, а также опционально JMF - сообщения (стрелка 2). Какая-то часть принятых и генерируемых системой AMS данных передается от допечатной стадии непосредственно или через информационно-управляющую систему (MIS) в печатную машину (стрелка 3) и оборудование для дальнейшей обработки (стрелка 4).

В этой главе рассматриваются только взаимосвязи (интерфейс) 1 и 2, а связи 3 и 4, касающиеся печати и послепечатной обработки, будут рассмотрены в главах 10 и 11. Интерфейс 1 осуществляется посредством соответствующего ICS - документа, а также MIS ICS - спецификации допечатной ступени, что даны в параграфе 9.1. Внутренние связи допечатной стадии 2 рассматриваются с использованием примеров в параграфах 9.2 - 9.5.

Некоторые примеры допечатной стадии были уже рассмотрены в параграфе 3.2. При этом мы исходили из условия, что допечатная стадия получает уже готовые данные в формате PDF. Поэтому процесс набора и создание PDFE (макет) не приведен в этом списке. Эти задания типичны для акцидентной печати и могут иметь некоторые отличия. Это касается реализации выполнения контроля формы (Formproof) методами программного обеспечения (Softproof или Remote-Proof-удаленной) или дополнительной проверки постраничного воспроизведения цветов (Seitenproofs) и так далее. Производственные процессы подготовительной стадии изготовления упаковки выглядят несколько иначе. Для описания множества заданий на стадии допечатной подготовки, существует ступенчатая система управления рабочим потоком (WMS), состоящая, как правило, из разных модулей.

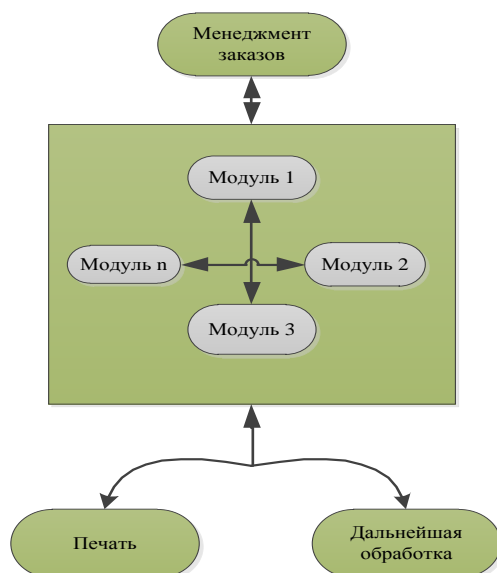


Рис. 9.1 JDF-/JMF - связи допечатной стадии

В параграфе 4.3 мы видели, что PJTF реализует связь между программным обеспечением Koordinator-Software и процессорами обработки заказов Jobticket-Prozessoren (как раз модулями – рабочими карточками) в архитектуре сквозного рабочего процесса Extreme-Workflow-Architektur. Эту функцию обеспечивает JDF/JMF (стрелки 2 на рис. 9.1). Аналогично случаю с PJTF, стандартный формат обмена данных не достаточен также и для рабочего процесса с использованием формата JDF при применении систем управления рабочим потоком (WMS) в случае объединения различных модулей разных производителей. Это возможно при учете множества специфических ограничений.

Примеры совместимых с JDF WMS (2009) продуктов:

Производитель:

Основной продукт:

AC&C HSH Group

PuzzleFlow

Avato System GmbH

PMS Production Management System

Agfa

Apogee

DALiM

PRiNTEMPO

EFI

OneFlow

EskoArtwork

BackStage Server

Fujifilm

XMF

Krause-Biagosch GmbH

KIM

LithoTechnics Oty Limited

Metrix

Heidelberger Druckmaschinen AG	Prinect
manroland	Printnet
Kodak	Prinergy
OneVision Software AG	Asura Pro
RamPage	RamPage Workflow System
Screen	TrueFlow
Xerox	FreeFlow Print Manager

9.1 Связи между MIS и Prepress

Прежде чем мы начнем рассматривать в этой части допечатную ступень *MIS-to-Prepress-ICS*, определим несколько общих положений, относящихся к интерфейсу.

В идеальном случае было бы замечательно, наряду с определением заказа и установкой в AMS данных о средствах производства, определиться с производственным процессом уже на стадии допечатной подготовки. В действительности, некоторые системы производственного процесса допечатной подготовки предлагают на основе JDF - информации, которую они получают от системы управления заказами, определенные производственные действия (шаги). Но так как менеджер отдела приема заказов не знает многих особенностей производства, то необходимо самостоятельно заполнить в системе управления рабочим потоком (WMS) эти недостающие пробелы. Производственные действия при этом не описываются в полной мере и могут быть дополнены и изменены в процессе выполнения

В целом, таким образом, операции выполняются в несколько этапов:

1. Системный администратор может с помощью сценариев определить новое направление производственного процесса;
2. WMS - система может по умолчанию выдать задания для различных операций и расходов, условий закупок, ориентируясь на типовые решения производства полиграфической продукции. Эти настройки основаны на процессы, заложенные изготовителем программного обеспечения системы управления рабочим потоком (WMS).
3. Пользователь может внести индивидуальные изменения в систему управления рабочим потоком (WMS) для индивидуальных заданий на печать.

Случай 1. Есть разные сценарии и возможности, которые имеются либо в операционной системе или от WMS - производителя системы. К первой категории относится *AppleScript* [8] или *Windows Script Host(WSH)* [34] с Script-двигателем *VBScript*

или *JScript*, в последнем это *Rules-Based Automation* (РБА) [30] в *Prinergy* (Kodak). Эти технологии – мощный инструмент и было бы полезным их использовать.

Во втором случае ответственный исполнитель по WMS определяет возможные изменения, например, изменение применяемой технологии растривания в RIP или данных треппинга (*Trapping*). Также он может установить возможность ввода дополнительных данных, автоматическую раскладку страниц (спуск полос) на печатном листе или использование в производственном процессе нового оборудования.

В третьем случае используется, как правило, выведение графического изображения предлагаемого системой управления рабочим потоком (WMS) производственного процесса на монитор пользователя. Пользователь, при этом, имеет возможность изменять отдельные рабочие шаги в соответствии с выполняемым заданием, например, исключить *Trapping* или модифицировать установленный вариант растривания в RIP.

Интерфейс между информационно-управляющей системой (MIS) и допечатным этапом (*Prepress*) является действительно очень значимым. На этом этапе происходит определение исходного процесса выполнения заказа в информационно-управляющей системе (MIS) и дальнейшая передача технических данных в систему *Prepress - WMS*, которая обеспечивает выполнения заказа. На практике, однако, при выполнении заказа могут вноситься некоторые изменения, это происходит уже на стадии допечатной обработки задания. Эти изменения могут быть для стадии допечатной обработки не существенными, например, величина тиража (хотя большой тираж и недостаточная тиражестойкость печатных форм потребуют изготовление дополнительных экземпляров форм) или значимыми, например, красочность или на какой печатной машине будет выполняться данная работа. В JDF - формате имеется возможность передачи определенной команды из MIS - системы для реализации этой возможности на стадии допечатной подготовки посредством команды *UpdateJDF-Kommando*. На практике часто применяется обратный процесс: оператор на стадии допечатной обработки вносит изменения в данные, которые необходимы для MIS - системы, чтобы исключить появления 2-х версий JDF - описания производственного процесса. Проблема внесения изменений в JDF - описания решена в настоящее время только частично и является важной темой для последующей проработки.

Rules-Based Automation

Mit RBA kann man außerhalb der üblichen Prinergy-Bedienoberfläche den Workflow von Prinergy beeinflussen und weiter automatisieren. Dazu kann man mit einem grafischen Werkzeug Regeln erstellen. Bei einer Regel wird zu einem Ereignis, das in Prinergy auftreten kann, eine Aktion assoziiert.

Zum Beispiel könnte man festlegen, dass, sollte ein schwerer Fehler beim Überprüfen der Daten auftreten (im Prinergy-Jargon: beim Refining), eine bestimmte Person im Betrieb eine E-Mail-Nachricht hierüber bekommt. So kann man den Import der Content-Daten in das Prinergy-System über einen Hot-Folder organisieren, die Datenprüfung im Hintergrund ablaufen lassen und nur im Fehlerfall nach Erhalt der E-Mail manuell eingreifen.

Ein weiteres Beispiel: Mittels Namenskonventionen der Dateinamen kann man die Seiten automatisch auf den Standbogen platzieren. Eine RBA-Regel fängt das Ereignis „Bogen gefüllt“ ab und das Ergebnis wird automatisch auf einem Formproofer ausgegeben.

Über RBA-Regeln lässt sich auch Fremd-Software in die Prinergy-Umgebung integrieren. Außerdem kann man über das grafische RBA-Werkzeug hinaus mit der Computersprache Visual Basic Workflow-Steuerungen definieren.

Rules-Based Automation

Посредством RBA возможно без изменения программного управления Prinergy вносить изменения в производственный процесс Prinergy, а также производить последующую автоматизацию. Для этого необходимо составить при помощи графического инструмента соответствующие правила. Эти правила ассоциируют результаты, получаемые при использовании модуля Prinergy.

Например, можно установить, что, в случае выявления грубой ошибки при проверке данных (на языке Prinergy: при Refining), определенный сотрудник предприятия получает об этом электронное сообщение. Таким образом, можно организовывать импорт данных контента в систему Prinergy посредством папок Hot, которые запускают проверку данных и позволяют вносить изменения вручную по получению E-Mail только в случае ошибки.

Следующий пример:

Посредством применения имен файлов можно выполнять автоматическую раскладку страниц на печатном листе. Правило RBA находит команду: „Лист заполнен“ и автоматически выдает результат на проверку формы. Посредством правил RBA возможно интегрирование чужих программных продуктов в модуль Prinergy. Кроме того, посредством графического инструмента RBA можно использовать при управлении производственным потоком язык программирования Visual Basic.

Интерфейс системы *MIS to Prepress ICS* характеризуется в основном *GrayBoxen* (“черные ящики”), т.е. информационными контейнерами, которые выстраивает MIS и которые должны быть заполнены данными на стадии допечатной обработки. Имеются 11 таких *GrayBoxen*:

- *PrePressPreparation (подготовка данных)*;

- *ImpositionSoftProofing* (введение программы проверки);
- *ImpositionPreparation* (подготовки макета);
- *ImpositionProofing* (монтаж, формирование контрольного оттиска вывод контрольного листа);
- *ImpositionRIPing* (монтаж, формирование печатной формы);
- *PlateSetting* (запись печатной формы);
- *PlateMaking* (изготовление форм);
- *PageProofing* (проверка страницы);
- *PageSoftProofing* (проверка программы проверки);
- *ContentCreation* (создание контента);
- *ProofAndPlateMaking* (проверка и изготовление печатных форм);

При описании производственного процесса на стадии допечатной подготовки *GrayBoxen* могут составляться из отдельных модулей. И, соответственно, значение этих модулей различно, что характерно для таких схем. Это сравнение более точно объясняется посредством диаграммы активности (рис. 9.2). На ней схематично представлен типичный производственный процесс, в котором каждая работа (предварительный просмотр, получение пробного оттиска, печатная форма) характеризуется своим способом вывода. RIP - процессы при этом очень разные.

Для процесса предварительного просмотра необходимо, как правило, разрешение в 72 ppi, для проверки (пробы) форм - от 600 ppi, а для вывода на формную пластину необходимо в большинстве случаев разрешение в 2400 или 2540 dpi. Кроме того, на формную пластину выводятся отдельно обработанные растрированные иллюстрации, а при проверке, напротив, - полный контент.

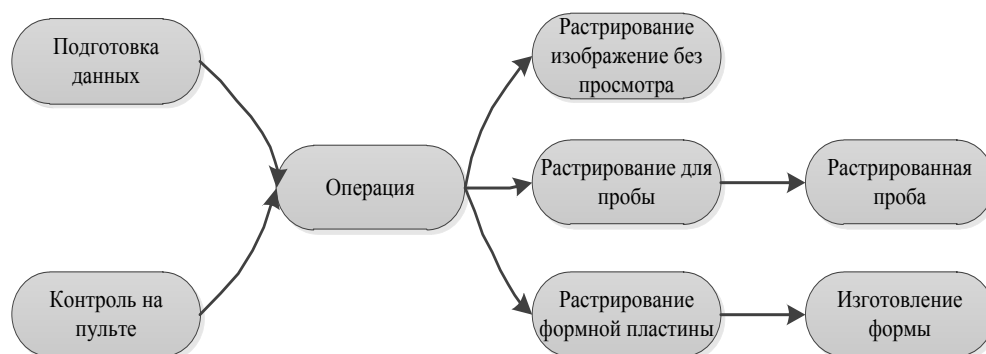


Рис. 9.2 Диаграмма активности стадии допечатной подготовки (акцидентная работа)

Контрольный вид выводится в зависимости от назначения отдельно или в совокупности.

Представленный на рис. 9.2 процесс отражен полностью в *GrayBoxen*:

- *PrepressPreparation*: подготовка данных;
- *ImpositionPreparation*: подготовка макета;
- *ImpositionProofing*: монтаж, формирование контрольного оттиска,
вывод контрольного листа;
- *ImpositionRIPing*: монтаж, формирование печатной формы,
дополнительно: формирование пробного оттиска;
- *PlateSetting*: запись печатной формы.

GrayBox PlateMaking - это совместный модуль, объединяющий *ImpositionRipping* (монтаж, формирование печатной формы) и *PlateSetting* (запись печатной формы), т.е. „большой серый блок“, который исключает два маленьких. Еще больший блок представляет собой *GrayBox ProofAndPlateMaking*, который объединяет *ImpositionProofing* (монтаж, формирование контрольного оттиска) и *PlateMaking* (изготовление форм).

Система *MIS-to-Prepress-ICS* устанавливает для каждого *GrayBox* те свойства, которые определяют, прежде всего, возможности ввода и вывода данных, а также их свойств. Три *GrayBoxen*: *ImpositionRipping* (монтаж, формирование печатной формы), *PlateSetting* (запись печатной формы) и *PlateMaking* (изготовление форм) представлены с их возможными ресурсами на рисунках, начиная с 9.3 по 9.6, причем ресурс *Plate-Making* уже был представлен на рис. 6.18. Обычно *ImpositionRIPing* (монтаж) и *PlateSetting* (запись печатных форм) соединяются последовательно. (запись печатных форм) соединяются последовательно.

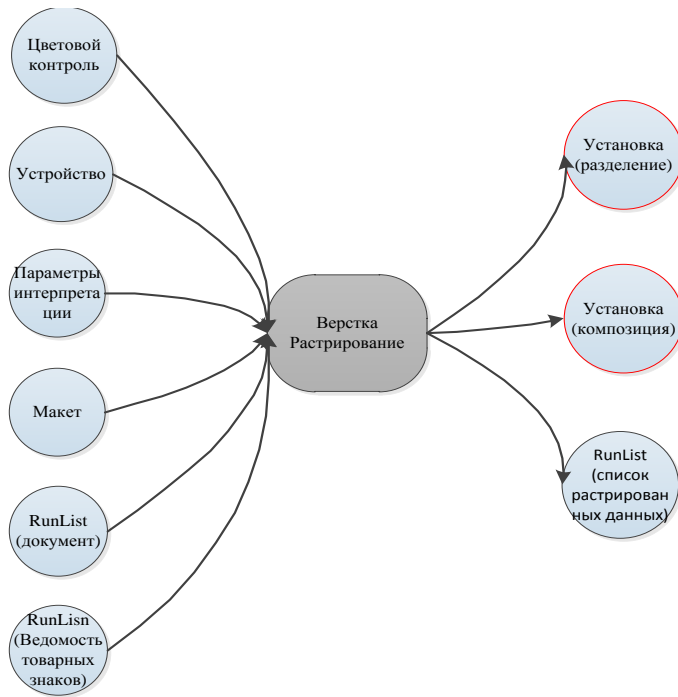


Рис. 9.3 GrayBox ImpositionRIPping

На рисунках синим цветом выделен ресурс *RunList (Resource)*, посредством которого представлен типичный процесс растрирования в TIFF - формате, что позволяет обеспечивать передачу данных между обоими модулями.

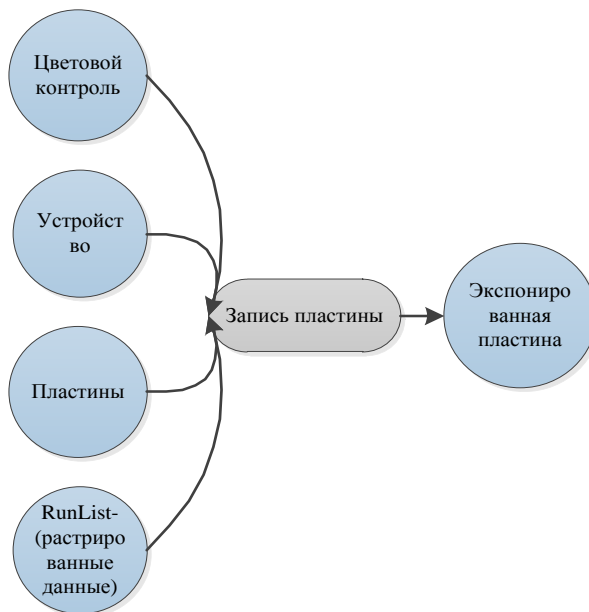


Рис.9.4 GrayBox PlateMaking охватывает функционирование ImpositionRIPping

Рис. 9.6 показывает, как *PlateMaking* (изготовление печатных форм) объединяет *GrayBoxen* - составляющие: *ImpositionRIPing* (монтаж, формирование печатной формы) и *PlateSetting* (запись печатных форм). При этом происходит передача данных от *ImpositionRIPing* с использованием его, как для ввода данных от *PlateSetting* (запись печатных форм), так и для вывода при *PlateMaking* (изготовление форм). Эти два модуля ресурсов Preview-Ressourcen показаны на иллюстрации красным цветом. Кроме того, *PlateMaking* (изготовление форм) должен получить все данные ввода *ImpositionRIPing* (монтаж, формирование печатной формы) и *Plate-Setting* (запись печатных форм), которые представляют собой только промежуточный результат и показаны голубым цветом.

Программный продукт информационно-управляющей системы (MIS) позволяет по-разному описывать производственный процесс допечатной подготовки, это видно на рис. 9.6, причем нанесенные стрелки характеризуют только последовательность формирования *GrayBoxen*. В зависимости от задания *менеджер*, т.е. подсистема MIS, может создать различные варианты построения *GrayBox-линий*. *Исполнитель (Worker)*, т.е. система производственного процесса допечатной стадии, должна объединять все версии и обеспечить их преобразование в основные узлы JDF-процесса в течение процесса производства.

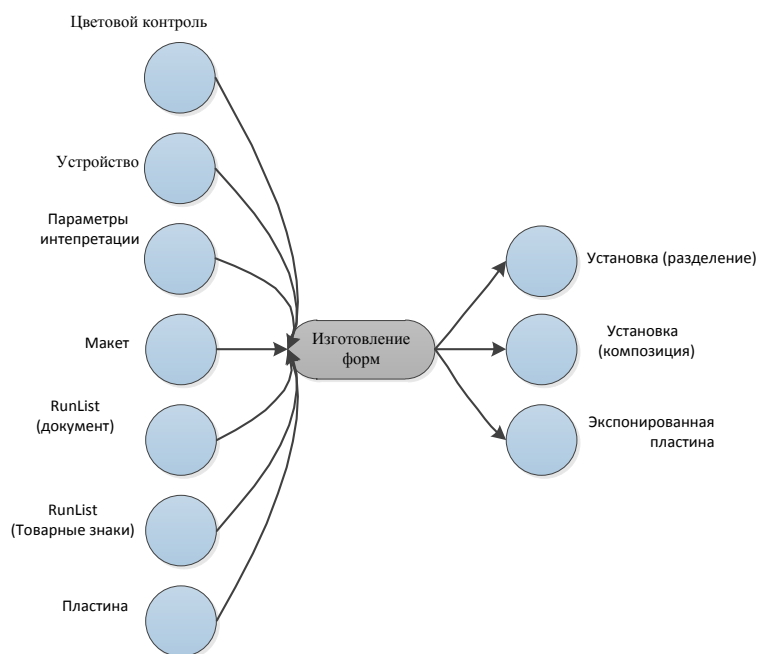


Рис 9.5 GrayBox PlateMaking охватывает функционирование PlateSetting

Аналогично в системе *MIS-to-PrePress-ICS* определены несколько находящихся друг над другом уровней. Все представленные *GrayBoxen* расположены в 1 уровне (и вместе

автоматически имеют уровень 2). Уровень 2 характеризуется дополнительными особенностями версий, которые могут встречаться вследствие языковых вариантов, а также необходимости построения комплексного варианта производства, при котором на одном листе могут быть расположены разные части продукта (например, обложка и основной текст).

В заключение необходимо отметить, что система *MIS-to-PrePress-ICS-Papier* охватывает также заданные величины других документов ICS -спецификаций, на которых они базируются. Таким образом, документы *Base-ICS* и *JMF-ICS* используются предположительно на уровне 2 – а именно на обоих уровнях системы *MIS-to-PrePress-ICS*.

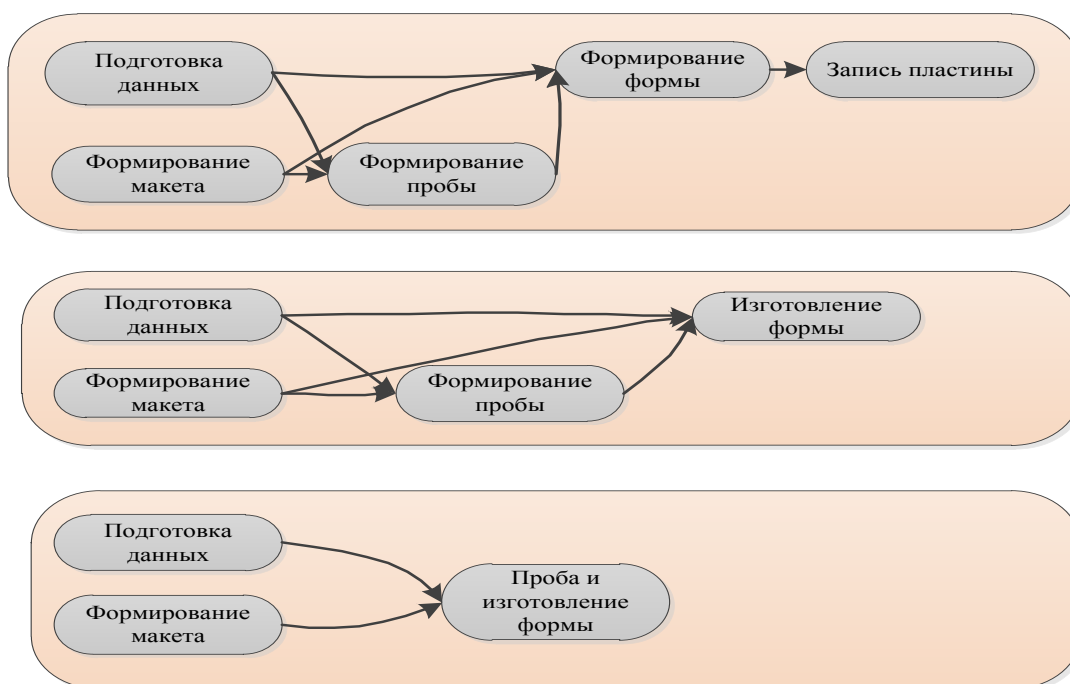


Рис. 9.6 Три разных возможности для GrayVoxen

Эти уровни могут быть совершенно разными. При *MIS-ICS*, которая является также предпосылкой для *MIS-to-PrePress-ICS*, ситуация описывается как и ожидалось: уровень 1 от *MIS-ICS* – является предпосылкой для уровня 1 от *MIS-to-PrePress-ICS* и уровень 2 от *MIS-ICS* - предпосылкой для уровня 2 от *MIS-to-Prepress-ICS*.

9.2 Монтаж

В этой части изложения речь идет главным образом о создании монтажного листа, часто называемом макетом. При этом основными задачами являются:

- размещение печатного листа на печатной форме;

- установка размеров листа и страниц;
- применение схем спуска полос.

Монтаж состоит из следующих трех рабочих этапов, которые выполняются последовательно друг за другом:

- создание монтажного листа (макета);
- расстановка страниц / полос на монтажном листе;
- спуск полос, что включает расчет структуры данных, необходимых для экспозиции одной или нескольких печатных величин кантов и расстояний между полосами (клапан, полоса нанесения клея, обрезка ...), размещение контрольных элементов (контрольная шкала, регистровая шкала, фальц, обрезка, разрезные и боковые метки, сигнатура листов ...) на печатном листе и печатной форме.



Рис. 9.7 Модель процесса ресурсов монтажа

Множество из этих данных может предоставить система управления заказами (AMS). В предыдущей главе мы уже довольно подробно анализировали модуль *ресурсов*

параметров процесса сборки (StrippingParams-Ressourcen), в котором аккумулируются эти данные. Сильно упрощенная структура производственного процесса показана на рис. 9.7.

Мы хотели бы подчеркнуть, что структуры JDF не представляют все процессы и ресурсы производственного процесса, а только его общие задачи.

Такая последовательность позволяет реализовать возможность автоматизации процессов производства. Без связи между JDF - сообщениями и системой AMS переход на новый монтажный лист сопровождался бы необходимостью ручного занесения в программу монтажа всех установок (Montage-Software). Возможно, также сохранение данных монтажного листа и их последующее использование для аналогичных работ непосредственно или с небольшой коррекцией. При JDF-интеграции между системой AMS и допечатной стадией все данные поступают непосредственно от системы управления заказами, и монтажный лист составляется автоматически. На практике в офсетной печати каждый составленный таким образом монтажный лист контролируется оператором, что позволяет вносить, при необходимости, незначительные изменения. Предпосылкой для использования этого способа является то, что контрольные метки позиционируются относительно размеров листа, а не так, как было раньше, с жесткой привязкой к координатам на листе. Для листов разного размера необходимы метки на разных позициях, а так как AMS задаются только размеры листа, а не координаты позиций меток, то метки должны расставляться автоматически.

Этот процесс определяется в настоящее время, собственно, программным продуктом обеспечения верстки и монтажа. Начиная с версии JDF 1.4, посредством MIS могут быть определены динамические марки в параметрах сборки (StrippingParams) и для их использования необходимо только запустить процесс Stripping. Однако, при этом необходимо, чтобы был подготовлен персонал и соответствующее программное обеспечение по работе с заказами.

Теперь мы проведем более точный анализ структуры JDF, приведенной на рис. 9.7. На рис. 8.7 был представлен процесс создания монтажного листа со всеми входными и выходными параметрами, т.е. *сборки (Stripping)*.

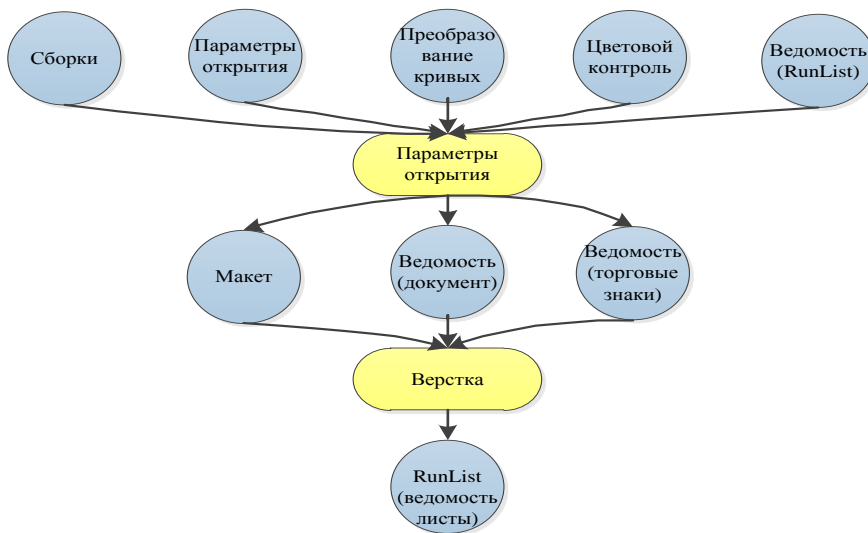


Рис. 9.8 Монтаж в модуле JDF

Рис. 9.8 дает более полное представление об этом процессе. Значок „?“ обозначает на этой иллюстрации опциональные ресурсы. Как видно, здесь по сравнению с рис. 8.7 использованы для процесса сборки (*Stripping*) модули ввода данных *TransferCurvePool* (кривых), *ColorantControl* (красочность) и *RunList* (документ). Модуль *TransferCurvePool* содержит, в основном, *трансферные кривые*, которые могут также называться *тональные корректурные кривые* (градационные характеристики) или *калибровочные кривые* процесса. Они, собственно, важны только для RIP-процесса, и поэтому мы их опишем точнее только в разделе 9.4. Здесь эти модули используются только для (если необходимо) получения сведений о трансформации координат между монтажным листом и печатной формой. В *Colorant-Control* устанавливается красочность фрагментов, что обеспечивает также воспроизведение соответствующих красочных меток на *монтажном листе*. Существует также возможность использования для *процессов сборки (Stripping-Prozess)* модуля ресурсов *RunList-Ressource*, который описывает *печатный документ* с возможностью *постраничного чтения*. Это позволяет осуществлять постоянный визуальный контроль данных монтажного листа на мониторе.

Модуль ресурсов *RunList-Ressource* является, в этом случае, опциональным для *Imposition-Prozess(процесса монтажа)*. Это значит, что этот модуль может, но не обязательно, быть выходным модулем процесса сборки (*Stripping-Prozess*) (поэтому „?“). В этом случае *RunList* (как это и есть) должен быть модулем *ввода (Input-Ressource)* для ресурса монтажа (*Imposition-Prozess*), иначе невозможно будет выполнить спуск полос. В этом отношении мы снова заключили в скобки вопросительный знак на рисунке. Модуль *Stripping-Prozess* позволяет получать один (или несколько) файлов, которые имеют все

контрольные метки в запрограммированных позициях на печатной форме. Это описывается в модуле *Ressource RunList* (марки). Посредством JDF - формата может передаваться функциональность и позиция контрольных меток, но не их вид. Это выполняется отдельно, например, в формате PDF.

В заключение мы хотели бы показать на рис. 9.9 еще один пример выполнения *монтажа* полос, в котором метки расставлены для 6 страниц. Такое размещение меток называется на языке JDF *ContentObject* (*содержание*). Как видно, монтаж выполнен в соответствии с заданным *расположением имен страниц (SheetName Side)*. Это должно быть так, потому что размещение страниц, а также позиции меток могут зависеть от сигнатуры, от запечатывания листов, а также от односторонней или двухсторонней печати.

```
<Layout Class="Parameter" DescriptiveName="Eostände" ID="300"
PartIDKey="SignatureName SheetName Side" Status="Available">
  <Layout Name="Signature-1" SignatureName="Signature-1"
  <Layout DescriptiveName="Umschlag" Name="Umschlag" SheetName="Umschlag"
  :SourceKeyStyle="WorkAndBack"
  :SurfaceContentsBox="0 0 2111.81 1714.96">
    <Media Class="Consumable" Dimension="2111.81 1714.96"
    :MediaType="Plate" />
    <Layout DescriptiveName="Schöndruck" Side="Front">
      <ContentObject CTM="1 0 0 1 316.06 188.50">
        ClipBox="307.55 180 1036.06 537.16"
        PositionX="Center" PositionY="Center"
        TrimCTM="1 0 0 1 316.06 188.50" TrimSize="711.49 340.15" />
      <ContentObject CTM="1 0 0 1 1084.25 188.50">
        ClipBox="1075.74 180 1804.25 537.16"
        PositionX="Center" PositionY="Center"
        TrimCTM="1 0 0 1 1084.25 188.50" TrimSize="711.49 340.15" />
      <ContentObject CTM="1 0 0 1 316.06 585.35">
        ClipBox="307.55 576.85 1036.06 934.01"
        PositionX="Center" PositionY="Center"
        TrimCTM="1 0 0 1 316.06 585.35" TrimSize="711.49 340.15" />
      <ContentObject CTM="1 0 0 1 1084.25 585.35">
        ClipBox="1075.74 576.85 1804.25 934.01"
        PositionX="Center" PositionY="Center"
        TrimCTM="1 0 0 1 1084.25 585.35" TrimSize="711.49 340.15" />
      <ContentObject CTM="1 0 0 1 316.06 982.20">
        ClipBox="307.55 973.70 1036.06 1330.86"
        PositionX="Center" PositionY="Center"
        TrimCTM="1 0 0 1 316.06 982.20" TrimSize="711.49 340.15" />
      <ContentObject CTM="1 0 0 1 1084.25 982.20">
        ClipBox="1075.74 973.70 1804.25 1330.86"
        PositionX="Center" PositionY="Center"
        TrimCTM="1 0 0 1 1084.25 982.20" TrimSize="711.49 340.15" />
      <MarkObject CTM="1 0 0 1 0" ClipBox="0 0 2111.81 1714.96">
        <RegisterMark Center="1900.86 661.94" Class="Parameter"
        :Rotation="0" />
        <RegisterMark Center="210.94 661.94" Class="Parameter"
        :Rotation="0" />
        <ColorControlStrip Center="1059.94 1347.76"
        :Class="Parameter" :Rotation="0" :Size="1486.16 28"
        :StripType="FOGRA_6_F74_740x10" />
      </MarkObject>
    </Layout>
    ...
  </Layout>
</Layout>
```

Рис. 9.9 Возможности JDF-Code при монтаже СТМ

Макет листа в качестве образца представлен на рис. 9.10. Обратите внимание, что в этом примере величина *SurfaceContentsBox* (*площадь, поверхность контента*) соответствует величине (*Dimension*) печатной формы (*Media*). Точка отсчета находится для обеих величин внизу слева. *Площадь (SurfaceContentsBox)* может быть и меньше, например, как самый маленький прямоугольный блок, который охватывает все печатные элементы на листе (= *Bounding Box*). Свойства *контента (ContentObject)* указаны ниже:

CTM - модуль трансформации матрицы (*Current Transformation Matrix*) был рассмотрен в параграфе 4.3 при представлении элементов в PDF. Это название перешло к нам из версии Post-Script и выполняет в JDF - записях аналогичные функции: определяет позицию, ориентацию и величину размещаемых меток на *SurfaceContentsBox* (площади листа), а также на печатной форме. Смотрите также учебный модуль в конце главы.

ClipBox - определяет величину и позицию размещения меток в пунктах DTP. Все элементы страницы вне *ClipBox* отрезаются. Позиция относится к *Surface-ContentsBox*.

PositionX, *PositionY* - способ выравнивания содержания в пределах координат размещения меток.

TrimSize - отрезной размер. Величина *Trimbox*, это размер страниц после обрезки или конечный формат печатной продукции. Величина указана в пунктах DTP.

TrimCTM - позиционирование отрезного размера (*Trimbox*) на *SurfaceContentsBox*.

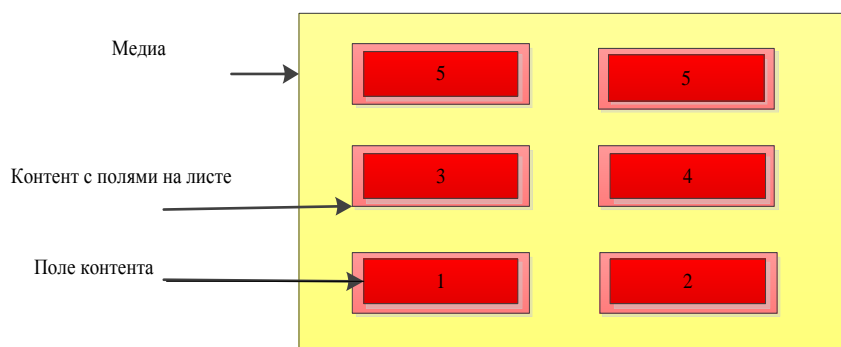


Рис. 9.10 TrimSize и ClipBox как составляющие ContentObjectClipBox

На рассматриваемом объекте (*MarkObject*) нанесены только контрольная шкала „FOGRA_6_F74_740x10“ с ее позициями, а также 2 регистровые марки.

Одно общее указание кажется нам еще важным: модуль Layout-Ressource претерпел существенные изменения в разных версиях JDF - сообщениях. Поэтому JDF - описания более старых версий могут выглядеть иначе, чем представлено здесь.

9.3 Треппинг

При многокрасочной офсетной печати может случаться, что наложение красок на оттиске абсолютно не соответствует заданным параметрам. Эти отклонения в наложении происходят, прежде всего, вследствие растяжения бумаги, а также и из-за погрешностей в печатной машине. Проявление этой неточности – это явление "отсвечивание" "(Blitzer)",

как показано на рис. 9.11. Поэтому модуль Trapping при определенных условиях позволяет избежать этого эффекта. Общее правило для полупрозрачных красок: более светлые краски наносятся на темные (см. рис. 9.12). Это позволяет частично исключить даже при малейших нарушениях точности нанесения краски такое явление как "отсвечивание" "Blitzer". При этом проявляется другое негативное явление, могут бросаться в глаза, вследствие треппинга, контуры мест стыка. Поэтому необходимо ширину треппинговых позиций выбирать как можно меньшей, ориентировочно соответствующей максимальному значению погрешности. Только в случае, если объект находится на черном фоне, ширину участка треппинга вследствие отсутствия его влияния можно увеличить. При непрозрачных красках, таких тонов как под золото или серебро, это правило не работает. При печати непрозрачными красками из очевидных оснований никогда нельзя использовать их для нанесения на полупрозрачные краски, правильно - наоборот. При использовании двух непрозрачных красок первоначально наносится основная краска, а затем - последующая.



Рис. 9.11 Отсвечивание, возникающее из-за погрешностей нанесения краски и смещений при наложении

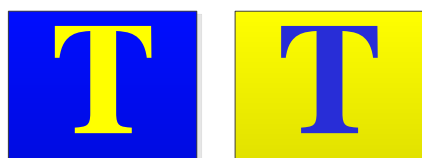


Рис. 9.12 Результат нанесения более светлой краски на темную

Для треппинга необходимо учитывать множество деталей, как, например, заданная Trap-ширина (дословно ловушка). При этом отсутствуют типовые решения, так как они определяются многими факторами (бумага, печатная машина, технология печати и др.). Данные треппинга могут быть описаны в модуле *TrappingDetails* (детали треппинга), который является модулем ввода данных процесса *Trapping* (рис. 9.13).

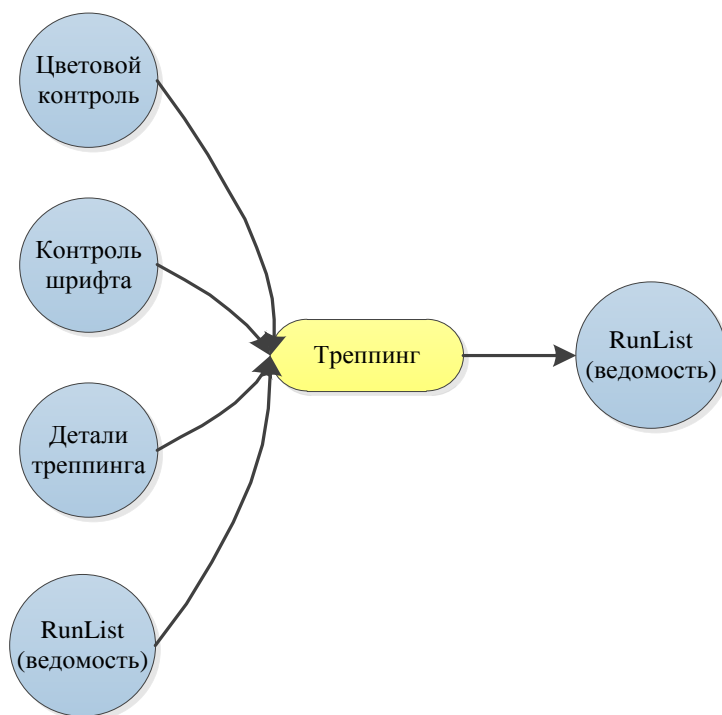


Рис. 9.13 Узел процесса треппинга

В *ColorantControl* (красочность) находятся характеристики покровных и полупрозрачных красок, которые используются для выполнения заказа. В модуле ресурсов *FontPolicy* (шрифтовые установки) заносятся данные о том, что должно происходить, если необходимые шрифты не представлены в данной библиотеке (остановить процесс, выбрать запасные шрифты), чтобы выполнить процесс создания модуля *RunList* для заданий на печать. Показатели процесса *Trapping* – это *RunList*, только с введенными данными контента.

Рис. 9.14 показывает пример записи деталей треппинг - ресурса (*TrappingDetails*) и их вид для последующей передачи в программу-двигатель (*Trapping-Engine*). Однако

практика показывает, что Trapping-Engine, как и RIP, не может быть сконфигурирован независимо от системы управления рабочим потоком. Другими словами: связь между Trapping-Engine и системой допечатной подготовки является специфичной у каждого разработчика систем, даже если она базируется на JDF - формате.

```
<TrappingDetails Class="Parameter" DefaultTrapping="true" ID="_333"
  Locked="false" Status="Available">
  <TrappingOrder>
    <SeparationSpec Name="Black" />
    <SeparationSpec Name="Cyan" />
    <SeparationSpec Name="Magenta" />
    <SeparationSpec Name="Yellow" />
    <SeparationSpec Name="Gold-Rouderfarbe" />
  </TrappingOrder>
  <TrappingParams>
    BlackColorLimit="0.95"
    BlackDensityLimit="1.6"
    BlackWidth="1.3"
    ImageToImageTrapping="false"
    ImageToObjectTrapping="true"
    ImageTrapPlacement="Normal"
    StepLimit="0.25"
    TrapWidth="0.25"
    ... />
</TrappingDetails>
```

Рис. 9.14 Часть приложения TrappingDetails

Теперь мы хотели бы объяснить отдельные детали и элементы, приведенные в качестве примера на рис. 9.14. Посредством модуля *DefaultTrapping* можно установить должны ли страницы полностью (значение - *true*) или только их определенные фрагменты (значение - *false*) быть обработаны. В подмодуле *TrappingOrder* устанавливается последовательность наложения красок – она должна задавать последовательность нанесения краски при печати. Как было установлено, эта последовательность при полупрозрачных красках имеет особую важность. В модуле параметров - *TrappingParams* задаются многие параметры для реализации процесса Trapping. В фрагменте записей рассматриваются только некоторые из них.

Атрибут *BlackColorLimit* указывает, с каким разрешением необходимо растривание при использовании *черной краски* – здесь при 95% градации серого. В *BlackDensityLimit* происходит аналоговое ступенчатое разделение цвета определенной плотности в зависимости от определенной плотности черного цвета. Значение определяется

приложением *Black-Width*, причем ширина треппинга черного цвета должна составлять 1,3-кратную величину от ширины треппинга остальных цветов. В *ImageToImageTrapping* и *ImageToObjectTrapping* можно установить необходимость обработки соседних иллюстраций или их фрагментов относительно других объектов (графика, шрифты). Так как в приведенном примере границы между изображениями и объектами подвергались треппингу (*Wert true*), то в *ImageTrapPlacement* указывается, как это должно происходить. Здесь выбрано стандартно, т.е. применены стандартные правила, которые основываются на светопропускании прозрачных и последовательности нанесения непрозрачных красок. Можно выбрать также, например, что объект принципиально наносится на изображение (*Choke*). Цветовой оттиск должен иметь отчетливую границу между двумя соседними фрагментами для последующего треппинга. Иначе может случиться, что треппинг будет выполнен для переходного участка. Такая граница устанавливается посредством *StepLimit*. В *TrapWidth* оговаривается ширина треппинга. Значение указывается в DTP-пунктах.

9.4 RIPing и изготовление печатной формы

Перед производственной системой всегда стоит задание распределить на отдельные процессы, описанные посредством MIS блоки *GrayBoxen*, такие как *PlateMaking* (*изготовление форм*) (рис. 6.17) или *Imposition-RIPing* (*монтаж, формирование печатной формы*) и *PlateSetting* (*запись печатных форм*) (рис. 9.4 и 9.5), на один или несколько комбинируемых процессов или группу процессов, которые состоят из подпроцессов. Возможны также 3 другие комбинации. Процессы, которые описаны таким образом, могут осуществляться, в принципе, совместно, как части единой системы, в которой выходные данные одного процесса будут входными данными для следующего. При этом речь идет в большинстве случаев о ресурсах *RunList* (*документ*), т.е. о промежуточных результатах. Нужно также иметь в виду, что при комбинированных процессах промежуточные результаты могут отсутствовать. Некоторые из этих частей были уже представлены, например, *Interpreting* (*интерпретация*), *Rendering* (*рендеринг*) и *Screening* (*скриннинг*) (рис. 3.10), другие упоминались, например, *Imposition* (*снук полос*) и *ImageSetting* (*экспонирование*) (параграф 6.1). На рис. 9.15 представлена еще раз полная возможная структура для процесса растривания RIP и получения печатной формы.

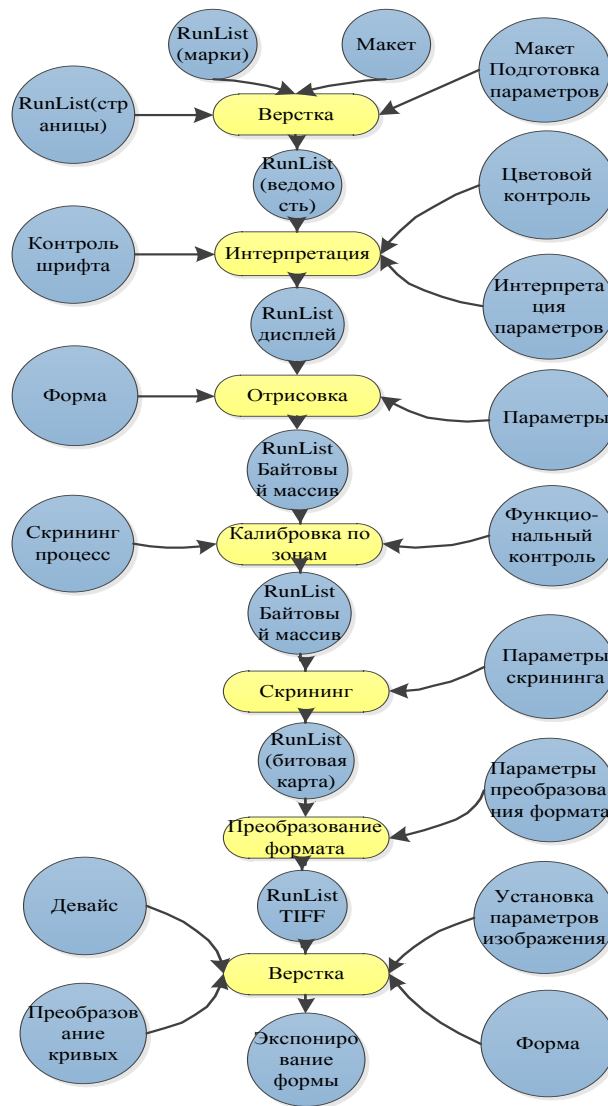


Рис. 9.15 Модель JDF для производства печатной формы

Далее детально остановимся коротко только на не рассмотренных ранее процессах.

Процесс *ContoneCalibration* (область калибровки) отвечает за применение *трансферных кривых* в RIP. Он получает от приложения *Rendering-Prozess* определенную информацию (*Bytemap*) по каждому шагу, т.е. для полутонового изображения (градация от 0 и до 100%) со стандартными (СМУК) или специальными (например, *Pantone*) цветами. Результатом этого процесса является структура данных в одной и той же форме, изменению по необходимости подлежит только величина градации. Как может изменяться величина градации, приведено в ресурсе *TransferFunctionControl*.

Описание трансферных (градационных) кривых аналогично описанию в PPF (рис. 4.11). Реально для каждого шага могут применяться несколько кривых. Чаще всего изготовление печатных форм сопровождается линеаризацией посредством *передаточных*

функций (Transferfunktion), это значит, что определяется полутоновая градация, значение которой используется для минимизации разницы между входными данными RIP и полученными значениями градации на форме. Наряду с кривыми линейаризации, которые зависят только от записывающего устройства, пластины и процесса проявления, существуют также трансферные кривые для печати, которые зависят от многих параметров, например раstra записи формы, запечатываемого материала, печатной машины, технологии печати, краски. Обе трансферные кривые просчитываются совместно и исполняются.

Модуль *FormatConversion-Prozess* точно исполняет то, что заложено в его названии: *конвертация одного формата данных в другой*. Здесь данные (Bitmap) производителя превращается в файл TIFF, который является входным сигналом для управления CtP - устройством. Так как эти данные содержат определенную информацию (Bitmap) в виде бинарного набора пикселей (только s/w и без серых тонов), то данный формат носит название TIFF-B. Этот формат является стандартным модулем связи Quasi-Standard между системой организации рабочего потока (Workflow) и системой записи. Это позволяет подключать к модулю системы без большого напряжения различные записывающие устройства. Следствием этого является возможность записи печатных форм отдельно от системы управления рабочим потоком. CtP - устройства и их специфическое программное обеспечение, как правило, не взаимодействуют при этом с JDF/JMF - файлами. В этом случае производственный процесс на базе JDF подключается к подготовке файла TIFF-B с последующей записью формы и генерацией соответствующего сообщения. Это также реализуемо при применении *MIS-to-PrePress-ICS*.

При помощи аналогичной модели на базе JDF возможно также создать оттиск на цифровой печатной машине. Однако на практике принято, что данные для цифровой печати аккумулируются еще раз в приложении Front-End цифровой печатной машины. При этом оператор имеет возможность перед запуском на печать, выполнить, например, сортировку заказов в соответствии с используемой бумагой.

На рис. 9.16 представлен аналогичный рис. 9.15 комбинированный процесс. Здесь повторяется только *ресурс LinkPool*, а не *ресурс Pool*. Необходимо учесть, что обозначенные красным цветом блоки на рис. 9.15 не находятся в *ресурсе LinkPool* слева, а также другие приложения в *ресурсе Pool*, что для комбинированных процессов допустимо.

```

<JDF Activation="Active" ID="_091" JobPartID="_1003,I" Status="Part"
Type="Combined" Type="Imposition Interpreting Rendering ColorCalibration
Screening PreviewGeneration FormatConversion ImageSetting" Version="1.3">
...
  <ResourceLinkPool>
    <RunListLink CombinedProcessIndex="0" ProcessUsage="Document"
Usage="Input" rRef="_77" />
    <RunListLink CombinedProcessIndex="0" ProcessUsage="Marks"
Usage="Input" rRef="_21" />
    <LayoutPreparationParamLink CombinedProcessIndex="0" Usage="Input"
rRef="_561" />
    <LayoutLink CombinedProcessIndex="0" Usage="Input" rRef="_99" />
    <ColorantControlLink CombinedProcessIndex="1" Usage="Input"
rRef="_83" />
    <FontPolicyLink CombinedProcessIndex="1" Usage="Input" rRef="_563" />
    <TransferCurvePoolLink CombinedProcessIndex="6" Usage="Input"
rRef="_71" />
    <InterpretingParamLink CombinedProcessIndex="1" Usage="Input"
rRef="_85" />
    <RenderingParamLink CombinedProcessIndex="2" Usage="Input"
rRef="_86" />
    <MediaLink CombinedProcessIndex="2" ProcessUsage="Paper" Usage="Input"
rRef="_79" />
    <ScreeningParamLink CombinedProcessIndex="3,4" Usage="Input"
rRef="_89" />
    <TransferFunctionControlLink CombinedProcessIndex="3" Usage="Input"
rRef="_34" />
    <ImageSetterParamLink CombinedProcessIndex="6" Usage="Input"
rRef="_24" />
    <DeviceLink CombinedProcessIndex="6" Usage="Input" rRef="_395" />
    <MediaLink CombinedProcessIndex="6" ProcessUsage="Plate" Usage="Input"
rRef="_3085" />
    <FormatConversionParamLink CombinedProcessIndex="75" Usage="Input"
rRef="_47" />
    <ExposedMediaLink CombinedProcessIndex="6" ProcessUsage="ExposedMedia"
Usage="Output" rRef="_83" />
  </ResourceLinkPool>
...
</JDF>

```

Рис. 9.16 Фрагмент записей для комбинированного процесса изготовления печатной формы

Как пример на рис. 9.17 приведены особенности ресурса *FormatConversionParams* (параметры конвертации формата). Здесь описана задача, где соответствующий процесс должен воспроизвести изображение в формате TIFF из специфического заданного объема информации.

```

<FormatConversionParams Class="Parameter" ID="_47" Status="Available">
  <FileSpec Class="Parameter" MimeType="application/octet-stream"
ResourceUsage="InputFormat" />
  <FileSpec Class="Parameter" MimeType="image/tiff"
ResourceUsage="OutputFormat" />
</FormatConversionParams>

```

Рис. 9.17 Ресурс *FormatConversionParams*

Эту часть мы хотели бы закончить следующим указанием: как уже подчеркивалось, JDF - коммуникация всеобъемлющего модуля системы управления рабочим потоком, RIP и выводного устройства не всегда может быть реализована на базе действительно открытых

интерфейсов. Это значит, что производитель использует для этого, по возможности, формат JDF (а также другие коммуникационные возможности, например, банки данных), но только с (совершенно законной) целью управления его собственными программными модулями. Чужое программное обеспечение при этом не рассматривается. В этом отношении результат наших исследований также не настолько удивителен. Мы установили, что формат JDF применяется везде в качестве «креативного» продукта. Особенно это касается комбинируемых процессов, при которых промежуточные результаты не должны быть зафиксированы. Почему, к примеру, получаемые промежуточные результаты должны быть внесены в ресурсы *RunList*, если система управления рабочим потоком имеет установленный тип действий, и работа сохраняется в определенной структуре или банке данных, или данные находятся только на рабочем сервере (RAM).

9.5 Proof и разрешение на печать

Проба (Proof или в немецком языке „Pruefdruck“, пробный оттиск) должен как можно более точно соответствовать последующим печатным листам тиража. Для ее получения применяются различные способы:

- контроль цвета, контроль наложения цветов или контактный способ получения контрольного листа (цифровым методом);
- контроль листа, также контроль печатной формы;
- контроль программными методами (Softproof);
- контроль растровой точки;
- машинный контроль или получение пробного оттиска на печатной машине (выполняется, если необходимо в глубокой или флексографской печати);
- аналоговый контроль (с использованием фотоформ.)

При этом первые три технологии играют важнейшую роль, а последние применяются в настоящее время очень редко.

Получение контрольного оттиска - это часть общего процесса организации контроля качества. При этом, естественно, необходимо контролировать и другие составляющие этого процесса, такие как: печатные формы, печатный тиражный оттиск или также отдельные компоненты печатных продуктов. Цель состоит в постоянном контроле процесса, при котором клиент/заказчик или сотрудник фирмы, могли бы оценить выполняемую работу и принять решение о ее продолжении или прекращении. На языке JDF – команды разрешения для дальнейшего продолжения процесса производства или его окончания. Эта тема раскрывается в книге при рассмотрении ресурсов *ExposedMedia* (можно назвать *носитель информации*), которые позволяют представлять контрольные

цветовые шкалы или пробный оттиск, или даже *RunLists*, где имеются все данные программного контроля (Softproof).

На рис. 9.18 приведена данная ситуация еще раз графически. Реально имеются один или несколько ресурсов, которые могут использоваться. Ресурс *ApprovalParams* (согласование) содержит детальную информацию о параметрах процесса изменения, это касается того, кто может вносить изменения (описано в подэлементе *Contact*) и какое положение занимает при этом эта персона или эти персоны. Это означает, кто в конечном итоге, обладает преимуществом по отношению к другим.

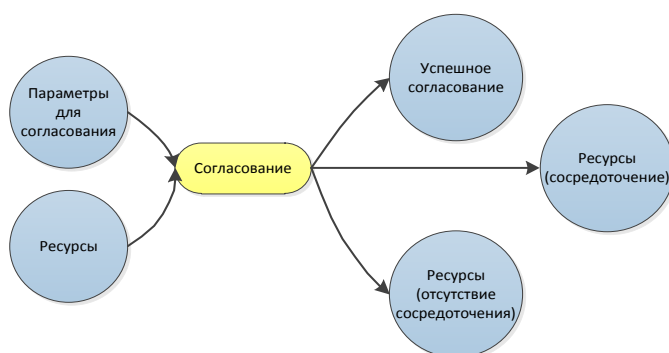


Рис. 9.18 Процессы для изготовления пробы

Вывод от параметров процессов *Approval* - процессов (согласование) состоит из разрешающих или не разрешающих ресурсов. *Статус-атрибут* ресурса имеет перед процессом *Approval* значение *Draft* (предварительный), после этого *статус Available* (принятый) либо *Rejected* (блокированный). В *ApprovalSuccess* дополнительно отмечено, кто точно подписывает или отклоняет пробу, приводятся возможные комментарии, а также связь с данными, которые были завизированы ответственной персоной.

```
<JDF Category="DigitalPrinting" ID="_1" JobID="_2" Status="Ready"
Type="Combined" Types="LayoutPreparation Imposition Interpreting Rendering
ImageSetting Approval"...>
...
</JDF>
```

Рис. 9.19 Запись команд на выполнение Proof

Процесс *Approval* (согласование) можно определять, как последний этап комбинированного процесса. Следовательно, *Hardcopy-Proof* для цифрового принтера - это комбинированный процесс, который заканчивается обоими процессами „*ImageSetting Approval*“ (согласование экспонирования форм), как это показано в записи на рис. 9.19. При машинной пробе комбинированный процесс охватывает в аналоговом

виде процессы *ImageSetting* с добавлением *ConventionalPrinting* (традиционная печать) вместо *ImageSetting*. С процессами традиционной печати - *ConventionalPrinting*, а также цифровой - *DigitalPrinting* познакомимся в следующей главе.

Для контроля изображений по цвету, а также для записи печатных форм, необходимы, как правило, коррекции цветового пространства. Для этого определяются цветные профили, посредством которых описывается цветовое пространство устройств. В качестве исходных данных (цифровая камера, сканер) используются специфически связанные с устройством данные RGB, которые необходимо посредством определенного профиля превратить в независимые от этого устройства данные. Для изготовления печатной формы необходим также еще второй профиль, который описывает процесс печати. Для получения контрольного профиля, характеризующего цветопередачу, необходим еще один профиль – профиль цветопередачи *Prooferprofil*.

Коррекция цвета выполняется при использовании формата JDF посредством процесса *ColorSpaceConversion* (преобразование цветковых профилей), который проводится, как правило, перед *трэппингом* (*Trapping*). Самый важный исходный ресурс - это *ColorSpaceConversion-Params*, в котором установлены *параметры и способы преобразования цветковых профилей*.

Пример 9.20 показывает такой ресурс, элементы и свойства которого будут описаны в дальнейшем.

```
<<ColorSpaceConversionParams ColorManagementSystem="Adobe" ID="R6"
Class="Parameter" Status="Available">
  <ColorSpaceConversionOp SourceCS="Gray" SourceObjects="All"
  Operation="Untag" />
  <ColorSpaceConversionOp SourceCS="RGB" SourceObjects="All"
  IgnoreEmbeddedICC="false" RGBGray2Black="true" RGBGray2BlackThreshold="1"
  Operation="Tag" RenderingIntent="Perceptual" >
    <FileSpec ResourceUsage="SourceProfile"
    URL="file://Server1/ICCProfiles/sRGBprofile.icm" />
  </ColorSpaceConversionOp>
  <ColorSpaceConversionOp SourceCS="CMYK" SourceObjects="All"
  Operation="Untag" />
  <FileSpec ResourceUsage="FinalTargetDevice"
  URL="file://Server1/ICCProfiles/CoatedFOGRA39.icm" />
</ColorSpaceConversionParams>
```

Рис. 9.20 Фрагмент записей элементов, ресурсов и атрибутов преобразования цветового пространства при Softproof

Остановимся далее на некоторых определениях в записях.

ColorManagementSystem: в записи приводится предпочтительная система для пересчета цветового пространства ("Adobe ");

FileSpec: в атрибуте приведен профиль ICC для определения требуемых значений;

ColorSpaceConversionOp: точно описаны операции по коррекции цветного пространства, причем:

* в *SourceCS* устанавливается исходное цветовое пространство;

*в *SourceObject* выполняется спецификация вида объекта, относительно которого производятся операции, например текст, виды штриховой графики, *ImagePhotographic* (фотография) и т.д. или же все, если это относится ко всем объектам;

*в модуле *Operation* определяется, будут ли профили учтены в графических элементах (*tag*) или наоборот (*untag*);

*в *IgnoreEmbeddedICC* указывается, необходимо ли учитывать предложенные профили (*true*) или нет (*false*);

*в *RGBGray2Black* может указываться, необходимо ли серый RGB-цвет при конвертации в CMYK повторно отображать на черном канале К (*true*) или нет (*false*), причем в случае *true*;

*приложение *RGBGray2BlackTreshhold* дополнительно может вводить ограничения по цветовой тональности в этот процесс; при значении 0 - только черный цвет RGB будет в полном объеме, при значении - 0,5 все значения серого в RGB составляют 50%, при значении 1- все серые RGB - тоны отображаются на канале К;

*в *RenderingIntent* устанавливается необходимый для управления цветом приоритет при *Gamut Mapping* (перерасчет одного цветового пространства в другое).

В примере удаляются все цветовые профили, которые касаются графических элементов цветового пространства CMYK или серого, и для всех элементов RGB создается ассоциируемый профиль, если для них он до сих пор еще не создан. Модуль коррекции цветового пространства описывает здесь процесс изготовления формных пластин офсетной технологии печати.

Задания для самостоятельной работы:

- изобразите монтажный лист с указаниями согласно рис. 9.9. Учтите при этом толкование трансформированных матриц согласно рис. 9.21. Каждая точка (x, y) в 2-мерном пространстве формально описывается вначале как 3-мерная точка (x, y, 1) (для возможности переноса). Трансформированная матрица - это последующая матрица 3x3, причем правая колонка имеет постоянные значения 0, 0, 1. Поэтому эти 3 величины не включаются всегда с одинаковыми значениями в *Current Transformation Matrix* (CTM) и остаются еще 6 значений не задействованных.

$$\begin{pmatrix} x & y & 1 \end{pmatrix} \begin{pmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & 1 \end{pmatrix} = \begin{pmatrix} a_{11}x + a_{12}y + a_{31} \\ a_{21}x + a_{22}y + a_{32} \\ 1 \end{pmatrix}$$

CTM = "a₁₁ a₁₂ a₂₁ a₂₂ a₃₁ a₃₂"

CTM = "1 0 0 1 a b" entspricht Verschiebung von (x,y) auf (x+a, y+b)

Рис. 9.21 Вид трансформированной матрицы

После вычисления матрицы третье измерение можно снова удалить, при этом видно, что, например, при CMT = "1 0 0 1 a b" необходимо, собственно, только перенесение координаты на величину (a, b).

Глава 10. Печать

На первый взгляд удивительно, что в формате данных JDF существуют только три различных процесса в области печати: *ConventionalPrinting* (традиционная - аналоговая печать с физической печатной формы: традиционная офсетная печать с увлажнением, флексография, глубокая и трафаретная печать), *DigitalPrinting* (цифровая печать) и *Varnishing* (лакирование). Для сравнения в допечатной подготовке и послепечатной обработке существует множество процессов, которые приведены в приложении к этой книге в немецком переводе названий процессов на основе JDF. Причина этого заключается в том, что рабочие процессы в области печати, в отличие от допечатной подготовки и послепечатной обработки, связаны между собой и не могут быть выполнены изолированно, самостоятельно и в любом порядке (см. параграф 3.3).

Мы будем обсуждать в этой главе лишь JDF - применение, которое представляет интерес для печати. Ранее (особенно в главе 9) мы раскрыли общие коммуникации между JDF/JMF - устройствами (в данном случае пульт управления печатной машины) и агентами и контроллерами (информационно-управляющая система или сервер системы рабочего потока). Для того чтобы новые возможности JDF - применения не были полностью вырванные из контекста, мы хотели бы вновь коротко рассмотреть типичные сценарии такого рода связей: ответственный контроллер печатной машины сохраняет данные JDF - данные первоначально в "горячей папке" (Hotfolder), которые постоянно считываются JDF - модулем пульта управления печатной машины. Параллельно контроллер отправляет команду JMF (*SubmitQueueEntry*) на пульт управления, который, в свою очередь, использует содержащийся там URL - адрес для загрузки файла JDF с файлового сервера, к которому имеют доступ обе стороны. Файлы JDF содержат кроме стандартной информации, такой как *имя клиента (Customer Information)* и *идентификационный номер задания (Job ID)* все описания ресурсов, необходимых для процесса печати, которые будут рассмотрены в следующих разделах. При сборе данных задания программа - агент/контроллер должна отправить с помощью JDF (*NodeInfo*) или через сообщение JMF запрос *фиксированного канала связи (persistent channel)* на пульт управления печатной машины (*Subscription*). При этом будет "созданный" канал, подтверждаемый данными *Response* - JMF и периодически отправляющий на

агент/контроллер запрошенные JMF - сигналы посредством протокола HTTP. Это происходит до тех пор, пока программа - агент/контроллер не пошлет команду, что больше нет необходимости получать сигнал о *закрывании канала (StopPersistentChannel)*. Сигналы подтверждаются программой - агент / контроллер или записываются на временное хранение на пульте управления, что на английском языке называется достаточно точно *fire and forget* (переведем как: *сделать и забыть*). Помимо этого, во время процесса печати, JDF - модуль на пульте управления фиксирует все данные протокола в качестве *элементов аудита* в *AuditPool* данных JDF - файла. После того, как команда выполнена, с пульта управления печатной машины передаются измененные файлы JDF обратно программе - агент/контроллер через папку *Hotfolder* или посредством JMF - сообщения.

10.1 Традиционные технологии печати

Некоторые печатные машины традиционной печати позволяют выполнять подготовку печатных форм и дальнейшую послепечатную обработку непосредственно в машине. Например, печатные машины, в которых непосредственно записываются формы, или рулонные офсетные машины, которые имеют в линии фальцевальные и обрезные модули. JDF - процесс *ConventionalPrinting* (*традиционная печать*) не включает в себя эти процессы, что требует на практике комбинирования процесса печати с процессами допечатной подготовки и послепечатной обработки. Для цифровой офсетной печати - это комбинированный процесс формирования печатной формы (*ImageSetting*) и традиционной печати (*ConventionalPrinting*). При рулонном офсете объединялись бы процессы печати (*ConventionalPrinting*) и послепечатной обработки (*WebInlineFinishing*).

В то время как на стадии допечатной подготовки задается множество параметров печатного процесса и последующей обработки, процесс печати непосредственно имеет очень простой интерфейс. Он получает указания от подсистемы допечатной подготовки информационно-управляющей системы (MIS) и, кроме печатных оттисков и отчетов о состоянии своей деятельности, больше ничего для других не производит. Не удивительно, что процесс *традиционной печати (ConventionalPrinting)* имеет много (дополнительных) ресурсов ввода (Input-Ressourcen), но только один выходной ресурс (Output-Ressource).

Какой вид информации, спецификаций и подготовительных работ других подразделений (особенно из MIS или подсистемы допечатной подготовки) имеют значение для печатного отдела?

JDF- системы для офсетной печати (2009 г.)

Производитель:	Продукт:
Akiyama	InkZone
Goss International	Goss Web Center
Heidelberger Druckmaschinen	Prinect Press Manager
manroland	printnet PressManager Sheetfed
Komori Lithografic Presses	K-Station
König & Bauer AG (KBA)	JDFLink für Logotronic
Mitsubishi Heavy Industries	IPC Server II
Ryobi Druckmaschinen	MIS Connection Software
Shinora Machinery Company	Shinora CIP4 Center / Station
Sakurai	InkZone Perfect

Для начала, независимо от того, получили ли Вы данные с помощью JDF или в любой другой форме отметим позиции, имеющие значение:

1. печатные формы;
2. печатная краска;
3. информация о запечатываемом материале (размер, сорт бумаги ...);
4. определение одно- или двухсторонней печати;
5. определение красочности работы;
6. административные данные (номер заказа, заказчик, условия приёмки, тираж, срок изготовления ...);
7. предварительная зональная настройка красочного аппарата в офсетной печати;
8. пробный оттиск (контроль формы, контроль контракта, образцы, пробный оттиск, контроль изображения);
9. позиции контрольных элементов на печатном листе;
10. определение условий печати (*Lab* - цветовое пространство или цветовой тон краски, растискивание, контрольно-измерительные шкалы ICC.....).

Только первые шесть пунктов в этом списке являются обязательными, следующие два - по крайней мере, широко распространены, последние два - играют особую роль. Позиции контрольных элементов во многих случаях определяются даже печатником печатной машиной или при изготовлении печатных форм. Встроенные или автономные измерительные системы (если они вообще существуют) могут читать и интерпретировать контрольные элементы только на определенных местах. Лишь немногие современные машины могут выполнять обратный процесс - позиционировать соответственно

установленные измерительные инструменты и определять положение контрольных элементов в виде метаданных. Кроме того, установка заданий для печати зависит, как правило, от стандартов, определяющих в этом отношении мира офсетных технологий печати.

В пункт 6 мы также включили сроки изготовления готовой печатной продукции, которые, как правило, устанавливаются при оформлении заказа. Ответственные за планирование отдельных процессов при производстве продукции должны придерживаться этих сроков. Мы хотели бы рассмотреть следующие три возможности:

- планирование осуществляется не на основе JDF, а, например, посредством аналоговых таблиц этапов выполнения плана, или посредством программных продуктов, которые посылают на пульт управления машины протокол, предлагаемый производителем продукта;
- планирование последовательности работ выполняется в модуле информационно-управляющей системы, и он передает технические и административные данные заказа в формате JDF под контролем MIS непосредственно на пульт управления машины;
- один из производственных отделов, составляющий график загрузки машин, получает данные из информационно – управляющей системы.

Первый случай не представляет интереса для нашей темы. Во втором случае система передает JDF - *данные заказа* непосредственно на пульт управления печатной машины, в третьем случае - JDF контактирует сначала с программным продуктом планирования производства. JDF - данные от информационно – управляющей системы имеет разные адреса и разное содержание.

На рис. 3.12 уже были представлены основные устройства ввода и вывода данных для офсетной печати. На рис. 10.1 мы показываем еще раз этот рисунок, но теперь с названиями JDF - ресурсов, с большей частью которых мы познакомились в последних разделах. Некоторые возможные ресурсы ввода JDF - данных мы не указывали, так как они необходимы только в особых случаях. Например, при обработке уже запечатанных листов необходим *Component-Ressource* (*компонент* - ресурс) при вводе данных (Input) Для флексографской печати должен быть указан двухсторонний скотч, который требуется для монтажа флексографских печатных форм на цилиндрах или гильзах, что требует дополнительных *Media-Ressource* типа *MountingTape* (*монтаж*). Другие устройства ввода данных, при необходимости *предварительного просмотра* изображения оттиска на пульте управления машины, такие как *Preview-Ressource* распространены довольно широко. В последующем содержании этого раздела мы детальнее рассмотрим различные ресурсы

процессов *традиционной печати (Conventional Printing)*, в том числе некоторых, которые не указаны на рис. 10.1.

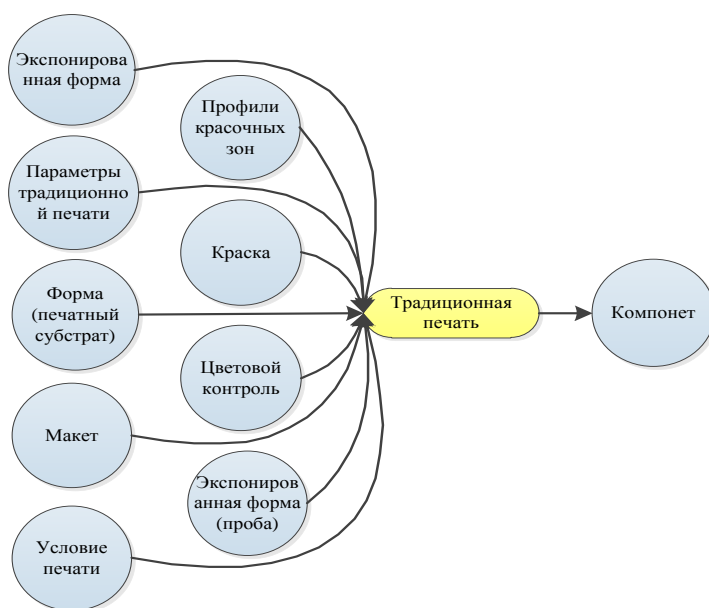


Рис. 10.1 Ресурсы традиционной офсетной печати

Параметры и условия печати. JDF - модель устанавливает границу между параметрами печати (*Conventional Printing-Params*) и условиями печати (*Print Condition*). Для настройки печатной машины в обязательных ресурсах *параметров печати (Conventional Printing Params)* имеются некоторые основные параметры. В них может быть указан тип сушки (УФ, ИК, сушка оттисков горячим воздухом), возможности отключения системы увлажнения для офсетной печати без увлажнения или установлена максимальная скорость листовой печати (отт./час) или угловая скорость (количество оборотов в ед. времени) при рулонной печати. Указывается также технология печати (офсетная, флексо, глубокая или трафаретная печать) и тип печатной машины (листовая печатная машина, рулонная с одной или несколькими печатными формами на формном цилиндре ...). Эти общие данные, как правило, заданы четкими стандартами или по умолчанию, для того, чтобы задание соответствовало печатной машине. В реальности эти атрибуты играют второстепенную роль. Важнее иметь представление о возможностях данной машины. Это раскрывается в конце параграфа 11.2. Существуют также зависимые от заказа задействуемые параметры печатной машины, которые хранятся в данном ресурсе, как, например, ресурс *WorkStyle (стиль работы)*, в котором есть указание на одностороннюю или двухстороннюю печать. На рис. 10.2 можно увидеть, что ресурс *WorkAndTurn (очередность)* подан в качестве метки, которая означает *переворот* (при

печати с оборота передняя кромка листа (клапан) остается как при печати по лицевой стороне, в то время как боковая кромка листа меняется; при печати с оборота печатная форма не меняется). В ресурсах также может быть указано, что клиент или лицо, ответственное за качество, может дать команду на «печать» печатной машине. Статус процесса *традиционной печати* (*ConventionalPrinting*) будет *остановлен*, а *статус деталей* (*StatusDetails*) – в положении *ожидания согласования* *WaitForApproval*.

```
<ConventionalPrintingParams Class="Parameter" ID="_400"  
PrintingType="SheetFed" Status="Available" WorkStyle="WorkAndTurn" />
```

Рис. 10.2 Параметры ресурса традиционной печати

В то время как ресурс (*ConventionalPrintingParams*) *параметры традиционной печати* имеет исходные значения для основных настроек печатной машины, в ресурсе *состояния печати* (*PrintCondition*) должны быть программно заданы параметры краски, ее плотности и растискивания при печати. В этом ресурсе можно специфицировать также спектрофотометрические данные измерения цвета (цветовая температура, фильтр, углы - 2 ° или 10 °, измерение по мокрой или сухой краске, исходные данные измерения). На практике, однако, этот ресурс мало используется, поскольку офсетная печать осуществляется, как правило, в соответствии со стандартами ISO 12647 [24] и, в этом случае, эти данные при заказе не нужны.

```
PrintCondition Name=" Papiertyp_1_ISO_12647-2" Class="Parameter" ID="_4711"  
PartIDKeys="Side Separation" Status="Available">  
<PrintCondition Side="Front">  
<PrintCondition Separation="Cyan"  
AimCurve="0.0 0.0 0.4 0.56 1.0 1.0" />  
<PrintCondition Separation="Magenta"  
AimCurve="0.0 0.0 0.4 0.56 1.0 1.0" />  
<PrintCondition Separation="Yellow"  
AimCurve="0.0 0.0 0.4 0.56 1.0 1.0" />  
<PrintCondition Separation="Black"  
AimCurve="0.0 0.0 0.4 0.53 1.0 1.0" />  
</PrintCondition>  
</PrintCondition>
```

Рис. 10.3 Заданные кривые растискивания при печати

На рис. 10.3 показан пример записей для заданной величины растискивания. Значения можно толковать по аналогии с рис. 4.11.

Предварительная зональная настройка красочного аппарата для офсетной печати.

Печатные краски для офсетной печатной машины дукторного типа выбираются в соответствии с печатным сюжетом (исключение: анилоксовые красочные аппараты, а также, так называемые, короткие красочные аппараты). Отдельные зоны подачи краски устанавливаются по ширине печати. В каждой из этих зон регулировочный винт воздействует на красочный нож или красочный шибер дукторного цилиндра так, чтобы подавалось только определенное количество краски. Величина зазора, как правило, устанавливается электронным способом. Предварительная настройка красочных зон красочного аппарата является частью настройки печатной формы.

Расчет значений предварительной настройки зон красочного аппарата с помощью данных допечатной подготовки находит широкое применение в формате PPF (Portable Print Format), как было указано в параграфе 4.2. Это сегодня настолько распространено, что многие продолжают использовать и далее формат PPF, а не JDF. Надо признать, что функциональные возможности от применения формата JDF не возрастают. То есть переход с одного формата данных на другой не приносит никакой выгоды для пользователя. Однако, преимуществом формата JDF есть его структурная реализация в информационно - управляющей системе, так как данные могут обрабатываться и передаваться далее совместно со всеми другими данными в формате JDF и не требуется никакой дополнительной обработки. В частности, нет необходимости устанавливать PostScript Interpreter для интерпретации файлов в формате PPF.

Расчет настроек зон красочного аппарата как JDF - модель представлена на рис. 10.4. Ресурс (*Preview-Resource*) предоставляет возможность *предварительного просмотра* всего монтажного листа (как правило, только с разрешением 50 пикселей на дюйм). Проще говоря, при расчетах настроек зон красочного аппарата необходимо задать количество пикселей на единицу площади и зону предварительного просмотра. В *параметрах расчета красочных зон* (*InkZoneCalculationParams*) определяются отдельные зоны, ширина которых может быть разной в зависимости от типа и производителя печатной машины.

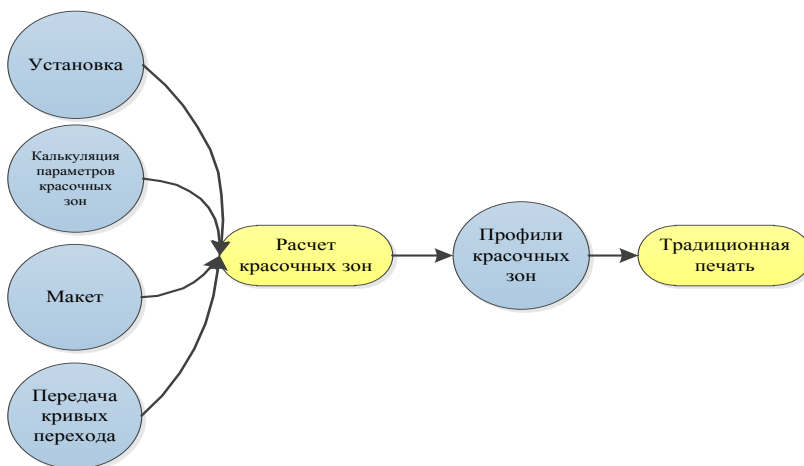


Рис. 10.4 Расчет настройки зон красочного аппарата при офсетной печати

В примере, приведенном на рис. 10.5., ширина зоны составляет 3,25 см ($92.125984 / 72$ x 2,54 см) и в общей сложности – 23 зоны по оси цилиндра. То, что в направлении подачи бумаги есть только одна зона, размер которой соответствует максимальному формату печати в направлении размотки - формально правильно, хотя вписывание в нее кажется странным. Расположение зон специфично для каждой печатной машины, предварительный просмотр позволяет видеть только лист. Для расчета настроек зон красочного аппарата, необходимо задать положение листа относительно зон. Это возможно проще всего, если рассматривать положение листа относительно печатной формы во время печати. Описание этого процесса можно найти в атрибуте *SurfaceContentsBox*, который упоминался уже несколько раз в этой книге как ресурс *Layout-Ressource*. И, наконец, необходимо иметь возможные кривые перехода, которые позволили бы изменять интенсивность тона при растривании во время предварительного просмотра (см. параграф 4.2).

```

<InkZoneCalculationParams Class="Parameter" ID="_25" Status="Available"
ZoneHeight="1451.338583" ZoneWidth="92.125984" Zones="23" ZonesY="1" />
  
```

Рис. 10.5 Параметры зональной настройки подачи краски

Значения предварительных данных зональной настройки красочного аппарата в формате JDF находятся в ресурсе *зональных профилей* (*InkZoneProfile*). При этом величина установки зазора дозировки краски указывается не в микронах, так как его

величина зависит от печатной машины. Вместо этого, каждое из предварительных значений зональной настройки красочного аппарата представляет собой цифру между 0 и 1, что является характеристикой усредненного заполнения площади растровыми точками в данной зоне.

Для получения процентного значения, полученный фактор следует умножить на сто. Истинное значение величины зазора можно представить с использованием трансферной кривой, которая определяется для заданных условий печати (краска, бумага) один раз для конкретной машины и не является составляющей частью описания в формате JDF. При неизменном изображении, то есть при одинаковых предварительных настройках параметров краски, именно величина подачи краски для белой бумаги должна быть выше, чем для цветной. Эти различия определяются для конкретной печатной машины, сохраняются в системе данных и используются в соответствующем заказе при печати. Таким образом, красочный аппарат может быть настроен на запечатываемый материал заранее, без получения пробного листа. Для того чтобы система управления всегда применяла данные в зависимости от условий печати, она должна иметь возможность использовать эти сведения из описания задания в формате JDF. В случае если параметры системы управления неверные или вообще отсутствуют (если они не соответствуют специфике заказа), это может привести к необходимости проведения множества дополнительных регулировок (контрольный оттиск, просмотр листов с целью сделать замеры или оценить, подрегулировать, сделать новые оттиски), что связано с дополнительными временными и материальными затратами.

На рис. 10.6 продемонстрирован пример записей кода для *профиля красочной зоны (InkZoneProfile)*. Ресурс разбит на разделы, включая и деление, так как предварительные значения подачи краски для каждой из них разные. На примере указаны значения (параметры) для черной краски. 23 значения в приложении атрибута *зональной установки X (ZoneSettingX)* указывают параметры площади запечатывания отдельных участков, начиная слева и ограничиваясь двумя знаками после запятой: 0,49 %, 0,86 %, 4,09 %, 5,25 % и так далее. Другой атрибут по оси *Y (ZoneSettingY)* содержит среднее значение площади запечатывания всех участков, в данном примере это 4,12 %.

```

<InkZoneProfile Class="Parameter" ID="_3" PartIDKeys="SignatureName SheetName
Side Separation"...>
<InkZoneProfile SignatureName="S1">
<InkZoneProfile SheetName="Umschlag">
<InkZoneProfile Side="Front">
<InkZoneProfile Separation="Black" Status="Available"
ZoneWidth="92.125984" ZoneSettingsX="0.0049479166666669755
0.008625565610862806 0.040955175339369494
0.05258389894419598 0.036772011689294073
0.04679616798642827 0.03305889423077225
0.04669306184012362 0.023023897058826546
0.03196920955882654 0.026367305335598803
0.022763303638766233 0.01726845305430168
0.027979296285825045 0.03681620003771036
0.0802906414969863 0.07557833710407524
0.07194204845399976 0.05891538225867557
0.08456872171945984 0.08174302413273286
0.029526241987182487 0.009106099170440482"
ZoneHeight="1451.338583"
ZoneSettingsY="0.04123003711309234" />
</InkZoneProfile ... />
...
</InkZoneProfile>
</InkZoneProfile>
</InkZoneProfile>
</

```

Рис.10.6 Значения предварительной настройки зон красочного аппарата

Контрольные шкалы и тест - объекты. Обычно на формные пластины экспонируют много контрольных элементов. Они отличаются функционально:

- контрольные элементы для изготовления печатных форм;
- контрольные элементы и элементы управления при печати;
- контрольные элементы и элементы управления для послепечатной обработки

продукции.

Важными для этого раздела являются только контрольные элементы для печати. Из них наиболее значимые:

- приводочные и регистровые метки (приводочные кресты);
- контрольные шкалы печатного процесса;
- боковые или накладные метки, которые также могут называться захватывающими или тянущими марками;
- сигнатура печатного листа.

Конечно, метки и тест - объекты должны быть размещены на определенных участках.

В листовой печати страничные метки размещаются непосредственно на краю листа,

контрольные метки печати параллельно цилиндрам в середине или же в конце листа и так далее. Метки наносятся на запечатываемый материал, чтобы иметь возможность посредством оптических или других измерительных приборов выполнять проверку сотрудниками, причем, последующая перепроверка не требуется. Только в случае применения автоматических измерительных устройств в линии (Inline) или вне ее (Offline), становится важным для восприятия и настройки точное размещение контрольных элементов на запечатываемом материале. Соответствующие измерительные системы постоянно совершенствуются, их становится все больше, например, Inline - измерительные системы для контроля подачи краски, применяемые в офсетной, флексографской и глубокой печати. В офсетной печати встречаются Inline - контрольные системы на спектрофотометрической основе. Печатаемые приводочные метки часто считываются inline-передающими ПЗС - камерами с целью автоматической для листовой или рулонной печати окружной, поперечной или диагональной приводки. Оператор допечатной подготовки устанавливает место расположения контрольных меток на монтажном листе посредством программы монтажа полос. В «динамических» метках, которые рассматривались в разделе 9.2, точное расположение определяется вначале только посредством программного продукта.

```

<Layout Class="Parameter" DescriptiveName="Lay" ID="_111" Name="Lay"
PartIDKeys="SignatureName SheetName Side" Status="Available">
<Layout Name="SIG1" SignatureName="SIG1">
<Layout DescriptiveName="Blätter" Name="Blätter" SheetName="Blätter"
SourceWorkStyle="WorkAndTumble" Status="Available"
SurfaceContentsBox="0 0 2111.81102362 1714.96062992">
<Media Class="Consumable" Dimension="2111.81102362 1714.96062992"
ID="_3062" MediaType="Plate" Status="Available" />
<Layout DescriptiveName="Schöndruck" Side="Front"
Status="Available"
SurfaceContentsBox="0 0 2111.81102362 1714.96062992">
...
<MarkObject CTM="1 0 0 1 0 0"
ClipBox="0 0 2111.81102362 1714.96062992" Ord="0">
<RegisterMark Center="2028.42519684 659.11417322"
Class="Parameter" MarkType="8AR_R_106x164" Rotation="0" />
<RegisterMark Center="83.38582677 659.11417322"
Class="Parameter" MarkType="8AR_L_106x164" Rotation="0" />
<ColorControlStrip Center="1059.44881889 809.11811023"
Class="Parameter" Rotation="0" Size="1798 28"
StripType="FOGRA_5_F74_740x10" />
</MarkObject>
</Layout>
</Layout>

```

Рис. 10.7 Метки в Layout-Ressource

Таким образом, позиции меток представляют типичный пример, в котором данные должны рассчитываться в соответствии с назначением. Это задание может быть

реализовано, конечно, на основе данных в формате JDF и на рис. 10.7 мы видим некоторые возможности ресурсов типа *MarkObject* (марки), которые встроены в ресурс *Layout-Ressource*.

Печатные формы и запечатываемый материал. Печатные формы (пластины, формные цилиндры машин глубокой печати, гильзы или сетка) встречаются двух типов: заготовки печатных форм (формные материалы) и готовые печатные формы, которые содержат данные об изображении. В офсетной печати можно говорить сокращенно об экспонированных и неэкспонированных пластинах, при этом правильно было бы использовать термин «иллюстрированные», так как не все пластины чувствительны к видимому спектру. Изменение пластин связано с процессом освещения (экспонирования).

```
<ExposedMedia Class="Handling" ID="_123" PartIDKeys="SignatureName SheetName
Side Separation"...>
<ExposedMedia SignatureName="Signatur_1">
<ExposedMedia SheetName="Umschlag" Status="Available">
<ExposedMedia Side="Front" Status="Available">
<ExposedMedia Separation="Cyan" Status="Available" />
<ExposedMedia Separation="Magenta" Status="Available" />
<ExposedMedia Separation="Yellow" Status="Available" />
<ExposedMedia Separation="Black" Status="Available" />
</ExposedMedia>
<ExposedMedia Side="Back" Status="Available">
<ExposedMedia Separation="Cyan" Status="Available" />
...
</ExposedMedia>
<MediaRef rRef="_1234">
<Part SheetName="Umschlag" SignatureName="Signatur_1" />
</MediaRef>
</ExposedMedia>
</ExposedMedia>

<Media Brand="745x605_Azura" Class="Consumable" Dimension="2111.811
1714.961" ID="_1234" MediaType="Plate" PartIDKeys="SignatureName SheetName"
Status="Available">
<Media Class="Consumable" Dimension="2111.81102362 1714.96062992"
SignatureName="Signatur_1" Stsatu="Available">
<Media Brand="745x605_Azura" Class="Consumable"
Dimension="2111.811 1714.961" MediaType="Plate" SheetName="Umschlag"
Status="Available" />
```

Рис. 10.8 Данные о формных пластинах для процесса печати

Неэкспонированные пластины описываются в JDF - ресурсе термином *Media*, при этом атрибут *тип носителя информации MediaType* должно соответствовать *форме (Plate)*. *Запечатываемый материал (MediaType)* определяется одинаковым ресурсом. В приложении *MediaType* необходимо задавать вид запечатываемого материала, например, *бумага (Papier)*, *фольга Foil (Folie)* или *гофрированный картон (Wellpappe)*.

Для печати необходимы экспонированные пластины (если не брать во внимание слепые пластины газетного производства). Они введены в ресурс *ExposedMedia-Resource*. На практике типографии используют только один вид пластин, размер которых четко предписан печатной машиной. Собственно говоря, печатнику нужно знать, какие пластины предназначены для данного заказа (*available*), экспонированные или неэкспонированные. Существуют устройства, которые автоматически передают заказ на печатную машину, если все формные пластины монтажного листа или печатные формы имеются в наличии. На рис. 10.8 продемонстрирована соответствующая запись для этого процесса. Из примера видно, что существует также ссылка на ресурс - *Media*, в котором задан вид пластин и их размеры.

На рис. 10.9 в JDF - коде дана возможность получения сведений о запечатываемом материале для обложки посредством ресурса - *Media*:

- размер листа: 61 x 43 cm (*Dimension*);
- класс бумаги 1 согласно ISO 12647-2:2004 (*Grade*);
- направление подачи бумаги параллельно к длинной кромке (*GrainDirection*);
- толщина бумаги 215 микрон (*Thickness*);
- вес бумаги 215 г/м2 (*Weight*);
- лицевая сторона глянцевая (*FrontCoatings*);
- обратная сторона матовая (*Back Coatings*);
- марка Cromolux (*Brand*).

```

a Class="Consumable" ID="_111" MediaType="Paper"
Keys="SignatureName SheetName" Status="Available"...>
a Class="Consumable" SignatureName="Signature_1" Status="Available">
a SheetName="Umschlag" Class="Consumable" ProductID="_999"
:"Available" Dimension="1729.1338582677165 1218.8976377952756"
:"1" GrainDirection="LongEdge" Thickness="215.0" Weight="215.0"
oatings="Glossy" BackCoatings="Matte" Brand="Chromolux" />
a>
a>

```

Рис. 10.9 Данные о запечатываемом материале

В этом ресурсе может быть размещена и другая информация, например, цветовая характеристика бумаги или степень ее непрозрачности. К сожалению, эти параметры (только изредка) не предоставляются поставщиками бумаги (например, в ценовых данных).

Цвет и контроль цвета.

Под термином «цвет» могут быть заданы различные характеристики: определение красочности страницы листа, последовательность наложения красок в печатной машине, свойства печатных красок, названия красок в каталогах, цветовая модель устройства вывода и так далее. В записях на основе формата JDF эти параметры могут быть также размещены в различных ресурсах. В ресурсе *краска (Ink-Resource)* приведен перечень печатных красок. В обычном случае это стандартные триадные краски, как это представлено на рис. 8.15. Здесь печать на лицевой стороне - многокрасочная (СМΥК), а печать на оборотной стороне – однокрасочная, цвет только черный. Разумеется, могут быть указаны также специальные краски или лаки. Применяемые названия должны соответствовать названиям краски в ресурсе *пула красок (ColorPool-Ressource)*. Здесь приведены все краски, которые используются в заказе, а при определенных обстоятельствах также и те, которые не используются в печатной машине. Все краски точно специфицированы, например, по *типу красок (ColorType)*, где приводятся сведения о кроющих и других характеристиках (*Normal*) или их нейтральности по цветности (*NeutralDensity*). В разделе допечатной подготовки мы уже видели, что эти свойства красок особенно важны для треппинг-процесса. Здесь могут быть также записаны Lab - или СМΥК - параметры красок, а также параметры специальных красок. Рис. 10.10 является простым примером ресурса *ColorPool-Ressource*. Параметры краски в пространстве СМΥК соответствуют приближенно значениям тональности СМΥК в процентах, нормированные относительно 1 вместо 100. *Туп краски (Color Type)* на примере, за некоторым исключением, всегда *стандартный (Normal)*.

```
<ColorPool Class="Parameter" ID="_500" Status="Available">
  <Color CMYK="1.0 0.0 0.0 0.0" ColorType="Normal" Name="Cyan"
  NeutralDensity="0.61" />
  <Color CMYK="0.0 0.0 0.0 1.0" ColorType="Normal" Name="Black"
  NeutralDensity="1.7" />
  <Color CMYK="0.0 1.0 0.0 0.0" ColorType="Normal" Name="Magenta"
  NeutralDensity="0.76" />
  <Color CMYK="0.0 0.0 1.0 0.0" ColorType="Normal" Name="Yellow"
  NeutralDensity="0.16" />
  <Color CMYK="1.0 0.0 1.0 0.0" ColorType="DieLine" Name="ProofColor" />
  <Color CMYK="0.2 0.3 0.4 0.5" ColorType="Normal" Name="PANTONEDeepBlue
</ColorPool>
```

Рис. 10.10 ColorPool-Ressource

Специальная краска используется только после пробы. Поэтому, эта краска не может быть отнесена к используемым краскам, для нее готовят печатную форму отдельно, если нужно ее печатать.

Последним ресурсом, который имеет отношение к определению краски, является ресурс *цветового контроля* (*ColorantControl-Ressource*) (см. рис. 6.18 параграфа 6.4; рис. 8.12 параграфа 8.2, а также 9.8 и 9.1 параграфа 9.1). Он предоставляет всю информацию, необходимую для совмещения характеристик красок (*Content-Daten*) с характеристиками красочных заданий устройства. Эта информация для печатной машины практически несущественна. Тем не менее, этот ресурс необходим для процесса печати, как подэлемент, называемый *DeviceColorant-Order*, который задает последовательность наложения красок, обычно это черная, голубая (циан), пурпурная (маджента) и желтая для листовой офсетной печати (рис. 10.11).

```
<ColorantControl Class="Parameter" ID="_2201" PartIDKeys="SignatureName
SheetName Side" ProcessColorModel="DeviceCMYK" Status="Available">
  <ColorPoolRef rRef="_500" />
  <DeviceColorantOrder>
    <SeparationSpec Name="Black" />
    <SeparationSpec Name="Cyan" />
    <SeparationSpec Name="Magenta" />
    <SeparationSpec Name="Yellow" />
    <SeparationSpec Name="PANTONEDeepBlue" />
  </DeviceColorantOrder>
  ...
</ColorantControl>
```

Рис. 10.11 ColorantControl-Ressource

Цветовая проба и предварительный просмотр. Для оценки качества печатнику необходим пробный оттиск, который представляет собой форматный оттиск и, как правило, выводится с использованием струйного принтера. Как только введен заказ, на пульте управления печатной машины высвечивается конфигурация - уменьшенная заставка изображения. В некоторых случаях (как в типографии Y в главе 2.2) на пульт управления, который должен иметь стандартизованное освещение и откалиброванные мониторы, цветовая проба выводится с помощью программ. Для процесса согласования используются нормативнокрасочный лист, пробные оттиски и уже напечатанные образцы. Как пробы на субстрате (твердая проба), так и образцы предварительного просмотра могут

быть описаны с помощью ресурсов, представленных в формате JDF. Цветопроба, как правило, представляется с помощью ресурса *Component* или посредством программного обеспечения с помощью ресурса *RunList*, как это приведено в разделе 9.5. В некоторых случаях (прежде всего в упаковочном производстве) пробы получают посредством экспонирующих устройств или специальных устройств получения цветопроб, которые могут обрабатывать растровые изображения (Rasterproofs). В таких случаях цветопробы демонстрируются посредством специальных вариантов ресурса *ExposedMedia-Resource*.

Компонент. Результатом процесса печати является оттиск, промежуточный продукт общего полиграфического процесса. На языке JDF это значит, что устройство вывода *ConventionalPrinting-Prozesses* – это *компонент* (*ComponentType*) соответствующий *запечатанному листу - оттиску*. Этот компонент характеризует не только запечатанный лист, но ещё и его *состояние*, что показано на рис. 10.12. В ссылке на компонент, то есть на *ComponentLink*, это отличие становится явным: оттиски разделяются на макулатурные (*Waste*) и тиражные (*Good*). И для каждого вида указывается допустимое и нужное количество в тираже. Так при 2040 тиражных оттисках запланировано 364 макулатурных листа. Общеизвестным является тот факт, что группа (*пул*) *AmountPool* представляет собой в некоторой степени счетчик ресурсов, который используется или воспроизводится во время производственного процесса. В принципе, *AmountPool* – это только тип карточки с заметками, на которую счетчик ресурсов может записывать свои значения.

```

<Component Class="Quantity" ComponentType="Sheet" ID="_123"
PartIDKeys="SignatureName SheetName Condition" Status="Unavailable">
<Component SignatureName="Signature_1">
<Component SheetName="Umschlag" />
</Component>
</Component>
<ComponentLink Amount="1" Usage="Output" rRef="_012">
<AmountPool>
<PartAmount Amount="2040.0">
<Part Condition="Good" SheetName=" Umschlag "
SignatureName=" Signature_1" />
</PartAmount>
<PartAmount Amount="364.0">
<Part Condition="Waste" SheetName=" Umschlag "
SignatureName=" Signature_1" />
</PartAmount>
</AmountPool>
<Part SheetName=" Umschlag " SignatureName=" Signature_1" />
</ComponentLink>

```

Рис.10.12 Компоненты печатного листа

Формат JDF преимущественно используется в офсетных типографиях и чаще всего в листовой офсетной печати. В рулонной офсетной, глубокой, флексографской и трафаретной печати он распространен меньше и его применение ограничено информационно – управляющей системой (MIS) и допечатными процессами (см. параграф 1.2). Факты использования формата JDF в процессах глубокой, флексографской и трафаретной печати на данный момент нам не известны, хотя эти технологии печати относятся к *традиционным технологиям (Conventional Printing)*. Основные производители печатных машин для этих технологий печати до сих пор уделяли данной теме очень мало внимания. Многие из них даже не являются членами организации CIP4. Это объясняется многими причинами:

- первые разработчики формата JDF были одновременно производителями офсетных печатных машин, поэтому требования других технологий печати начали учитываться гораздо позже и в меньшей мере;
- в глубокой и флексографской печати допечатными процессами и самой печатью часто занимаются разные предприятия, что затрудняет процесс интеграции;
- в офсетной печати использование данных по зональной установке величин подачи краски по умолчанию существенно повышает эффективность оборудования и снижает затраты, что для других технологий печати незначительно;
- стандарты печати фактически существуют только для технологий офсетной печати с увлажнением, для других технологий используются стандарты издательства или вообще специфические стандарты клиента;
- разнообразие материалов для печати намного больше, чем в офсетной печати, что затрудняет использование автоматизированных систем;
- при тиражах в офсетной печати, которые по сравнению с глубокой и флексографской печатью, как правило, меньше, наладочное время имеет большее значение; - в области глубокой печати существует очень мало больших типографий, которые располагают собственными системами управления производственным процессом.

Однако существует вероятность, что подобная ситуация изменится уже в ближайшем будущем. Ведь JMF - сообщения об использовании материалов, актуальном состоянии печатных технологий и потребностях рынка в любом случае интересны современному производству.

10.2 Цифровая печать

Цифровая печать предпочтительна при небольших тиражах, коротких сроках и персонализации заказов. По этой причине максимально всеохватывающая автоматизация является важным условием рентабельности данной технологии печати. Следовательно, формат JDF соответствует поставленной цели.

На сегодняшний день известны две основных сферы использования цифровой печати, которые, однако, достаточно сложно разграничить:

- офисная канцелярия и копи-центры;
- профессиональная печать и цифровые системы печати.

Цифровые системы печати отличаются от принтеров в копи - центрах, в первую очередь производительностью, большим количеством опций, точностью цветовоспроизведения и расширенным набором встроенных опций дальнейшей обработки. Данные категории прослеживаются в двух соответствующих документах спецификаций (ICS): *Office Digital Printing ICS* и *Integrated Digital Printing (IDF) ICS* соответственно. Интегрированные системы цифровой печати на основе формата JDF, часто являются составляющими „Hybrid-Workflows“ (гибридный рабочий поток) - системы управления производственным процессом, в которой информация готовится как для офсетной, так и для цифровой печати.

Интегрированные печатные системы	Цифровая печать для бюро
LayoutPreparation	LayoutPreparation
Imposition?	Imposition
ColorSpaceConversion?	
Interpreting	Interpreting
ColorSpaceConversion?	
Rendering	Rendering
ColorSpaceConversion?	
Screening?	
Imposition?	
DigitalPrinting	Digitalprinting?
Folding?, Stitching?, Trimming?, Hole-	Folding?
Making?, CoverApplication?, SpineTaping?	Stitching?

Таблица 10.13

Использование формата JDF способствует применению цифровых технологий не только в изготовлении офсетных форм, но и в гравировке формных цилиндров глубокой печати. Ведь и в этой сфере могут возникнуть ситуации, когда информация в формате JDF экспортируется из информационно – управляющей системы в систему гравировки, а после изготовления печатных форм передается обратно в систему MIS.

Цифровая печать отличается от офсетной печати, с точки зрения технологических процессов, в основном, двумя особенностями:

- в цифровой печати не требуется изготовление постоянных физических печатных форм – соответственно отпадает необходимость в использовании свойственного формату JDF процесса *ImageSetting*;

- в устройствах цифровой печати часто встроены модули для процесса дальнейшей обработки.

Второе отличие указывает на то, что цифровая печать в программах в формате JDF представлена комбинированным процессом, в ходе которого продукты промежуточных этапов нет необходимости описывать. Возможные процессы по очередности отображены в таблице рис. 10.13, в соответствии с обоими документами ICS.

В первую очередь в интегрированных печатных системах обращает на себя внимание тот факт, что процессы *Imposition* и *ColorSpaceConversion* упоминаются несколько раз и отмечены как опциональные знаком вопроса. При этом подразумевается, что оба процесса при последовательной обработке данных могут находиться на разных позициях, а также то, что каждый из них должен находиться на определенной заданной позиции.

Процессы дальнейшей обработки опциональны и могут выполняться практически в заданной последовательности. Очевидно, что цифровые офисные принтеры не располагают системой контроля цвета и имеют меньше возможностей для дальнейшей обработки полученных оттисков .

За счет множества составляющих процессов, подобный комбинированный процесс может располагать большим количеством ресурсов ввода. На выходе комбинированного процесса всегда присутствует ресурс *Component*. Если выводятся промежуточные результаты процесса, то они обычно представлены ресурсами типа *Component* или *RunList*.

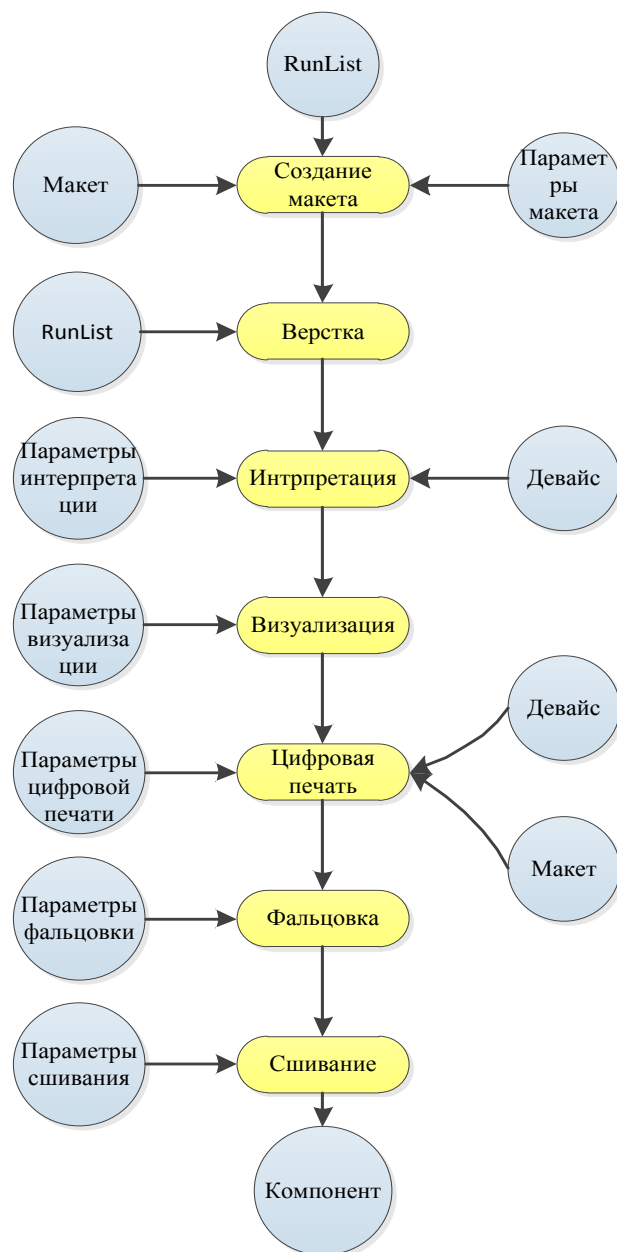


Рис. 10.14 Практический пример схемы цифровой печати

На рис. 10.14 показан пример, который, возможно, не отображает все варианты схем цифровой печати, однако взят из практики, а на рис.10.15 представлен соответствующий код записи.


```

<?xml version="1.0" encoding="UTF-8"?>
<JDF Status="Ready" Type="Combined" Types="LayoutPreparation Imposition
Interpreting Rendering DigitalPrinting Stitching Folding" Version="1.3" ...>
...
<ResourceLinkPool>
<ComponentLink Amount="65" CombinedProcessIndex="5" Usage="Output"
rRef="_2503">
<Part SignatureName="Sig1" />
<Part SignatureName="Sig2" />
<Part SignatureName="Sig3" />
</ComponentLink>
<LayoutPreparationParamsLink CombinedProcessIndex="0" Usage="Input"
rRef="_2504" />
<MediaLink CombinedProcessIndex="0 4" Usage="Input" rRef="_0918" />
<DigitalPrintingParamsLink CombinedProcessIndex="4" Usage="Input"
rRef="_2507" />
<DeviceLink CombinedProcessIndex="2 4" Usage="Input" rRef="_2509" />
<RenderingParamsLink CombinedProcessIndex="3" Usage="Input"
rRef="_2511" />
<InterpretingParamsLink CombinedProcessIndex="2" Usage="Input"
rRef="_2510" />
<StitchingParamsLink CombinedProcessIndex="5" Usage="Input"
rRef="_2513" />
<StitchingParamsLink CombinedProcessIndex="6" Usage="Input"
rRef="_2514" />
<RunListLink CombinedProcessIndex="0 1" ProcessUsage="Document"
Usage="Input" rRef="_9437" />
</ResourceLinkPool>
</JDF>

```

Рис. 10.15 Практический пример кода цифровой печати

Процесс монтажа (*LayoutPreparation*) аналогичен описанному в параграфе 9.2 процессу *Stripping*. Результатом обеих процессов является макет, необходимый для спуска полос. Оба располагают ресурсами ввода, которые обеспечивают их информацией, необходимой для генерирования макета. Для процесса *Stripping* этот ресурс называется *StrippingParams(параметры)*, для процесса генерирования макета соответственно *LayoutGenerationParams*. Оба процесса отличаются способом применения. Подготовка макета используется в цифровой печати, в то время как процесс *Stripping* характерен для допечатных процессов в офсетной печати, связанных с информационно – управляющей системой (MIS). Возможно, что следующая версия JDF внесет изменения в данную область, ведь уже в версии 1.4 в описании ресурса *LayoutPreparationParams* было указано, что в последующих версиях этот параметр может отсутствовать.

Персонализация. О персонализированной печати - *Variable-Data Printing (VDP)* говорят, когда отдельные элементы (текст, графика, рисунки, изображения) отличаются на каждой странице (единице использования) оттиска. Обычно VDP представляет собой

печать постоянной части, идентичной на каждом листе, и переменной части, поставляемой базой данных или специальным файлом.

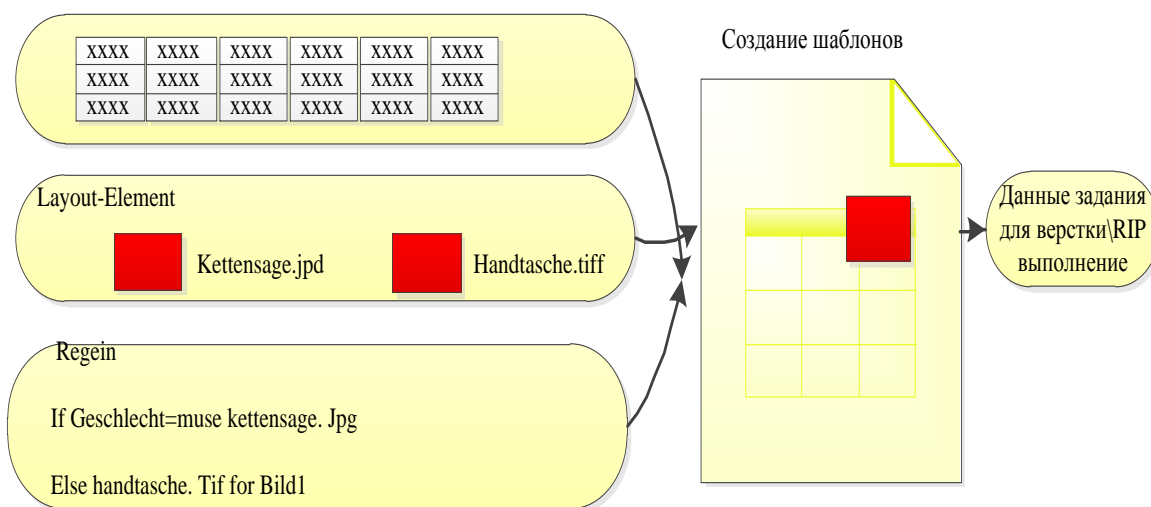


Рис. 10.16 Частные задачи при персонализированной печати

База данных (перевод верхней части рис. 10.16)

Фамилия	Имя	День	Пол	Место	Улица	Индекс
Кул	Карл	19.03.1963	м	Eisleben	Am Kaltenbach	6295
Фриш	Фредерика	11.11.1961	ж	Kaltenbronn	Eisigweg 42	76593

Задание на персонализированную печать в большинстве случаев состоит из следующих составляющих (рис. 10.16):

- данных на изготовление информационного фонда (базы данных или таблицы) с адресатами;

- задания дизайна оригинала для страницы или другой единицы использования с размещением постоянных и переменных элементов, чаще всего с помощью плагинов (Plug-Ins) макетных программ;
- правил для соединения переменной части оригинала с информационным фондом;
- объединения всей информации для вывода на растровый процессор.

Для выхода персонализированных данных на растровый процессор можно, как правило, использовать формат PDF. Однако в таком случае каждая страница/каждый лист требуется заново обработать на растровом процессоре, включая постоянные элементы. Типичная переменная составляющая является не более чем отбором комплекта немногих постоянных элементов. С этой точки зрения подобные манипуляции были бы неэффективными. Именно поэтому следует использовать особые языки и соответствующие растровые процессоры, позволяющие одноразовый перенос и обработку на растровом процессоре переменных данных, и в тоже время многократное и вариативное использование постоянной части и графических элементов.

Из всех специфических форматов, которые мы не рассматриваем детально, следует выделить основанный на XML язык разметки оригинала печати *Personalized Print Markup Language* (PPML) [11] [36] [37] [38] с его подвидом PPML/GA (*изобразительный оригинал*). Другие варианты: формат данных PPML/VDX (*Variable Data Exchange*) (*обмен переменной информацией*), базирующийся на форматах PPML и PDF, а также недавно выпущенный фирмой Adobe специализированный формат PDF/VT (*переменный транзакционный*). Некоторые специалисты в этой сфере предвещают, что формат PDF/VT [25] в будущем вытеснит формат PPML, в первую очередь за счет интегрирования в *Print Engine 2* (технология OEM-RIP от Adobe Systems [1]).

Формат PPML, как и JDF, был представлен в 2000 году. Этот программный язык был разработан организацией PODi (Digital Printing Initiative), основанной в 1996 году. Является ли на данный момент PPML конкурентом для JDF в сфере цифровой печати или они дополняют друг друга? Чтобы ответить на этот вопрос, следует детальнее рассмотреть основные свойства PPML.

PPML позволяет интегрировать цифровые объекты в различных форматах, такие как рисунки, тексты, страницы (EPS, PDF, JPEG, TIFF,...) в файл PPML. При этом объекты не обязательно встраиваются в файл PPML, на них можно дать ссылку. В файле PPML эти объекты можно размещать, трансформировать и обрезать. В PPML также можно определить объект как многократный, с дальнейшей одноразовой обработкой растровым процессором и сохранением в кэше для повторного использования. То есть, графические

объекты сохраняются не в формате PPML, а в других стандартных форматах. PPML описывает только расположение данных объектов на странице.

Сильно упрощая сказанное: PPML просто описывает страницы VDP, как и PDF; отличие в большей структуризации при описании. Кроме того, как следует из первых глав, производственный процесс в JDF – рабочий поток начинается часто на готовых заказах клиента – следовательно, PPML и JDF являются лишь дополняющими форматами. Однако это не совсем правильно по нескольким причинам:

- JDF описывает также разработку страниц, что выражено процессом *LayoutElementProduction*;

- PPML охватывает опционально и метаданные производственного процесса, как, например, описание макета верстки [37].

Следовательно, хотя оба формата имеют общие черты, они во многом дополняют друг друга. Например, свойства запечатываемого материала в PPML не рассматриваются отдельно, в отличие от JDF, где для этого используются *Ressource Media*. Хотя файлы PPML могут вмещать данные в формате JDF или иметь ссылку на них, данная опция была упразднена в версии 2.2, поскольку PPML в дальнейшем должен использоваться только для описания контента. В свою очередь, ресурс *RunList* может ссылаться на данные в формате PPML. Этот ресурс является вводным для пользователя PPML. В работе [39] детально рассмотрено сочетание обоих форматов. Так, атрибут *IgnorePDLImposition* в ресурсе JDF *LayoutElement* определяет источник информации о макете листа – данные в формате PPML или JDF соответственно.

На рис. 10.17 изображена JDF - модель для VDP. Ресурс *RunList* справа представляет данные в PPML или другом языке VDP. Со своей стороны, он является типичным вводом для процесса спуска полос. Слева представлен процесс *DB – DocTemplateLayout*, позволяющий получить оригинал для постоянных и переменных элементов с уже заданным определением связи с базой данных. Для выполнения данной задачи необходимо задать правила переноса текстовых и графических элементов в оригинал. Поскольку на практике правила описываются приложениями разным образом, в правилах DB им отводится роль комментария. В ресурсе *DBSchema* указывается тип базы данных (SQL, XML или разделенные запятой фрагменты текста) и опять же соответствующий комментарий к схеме.

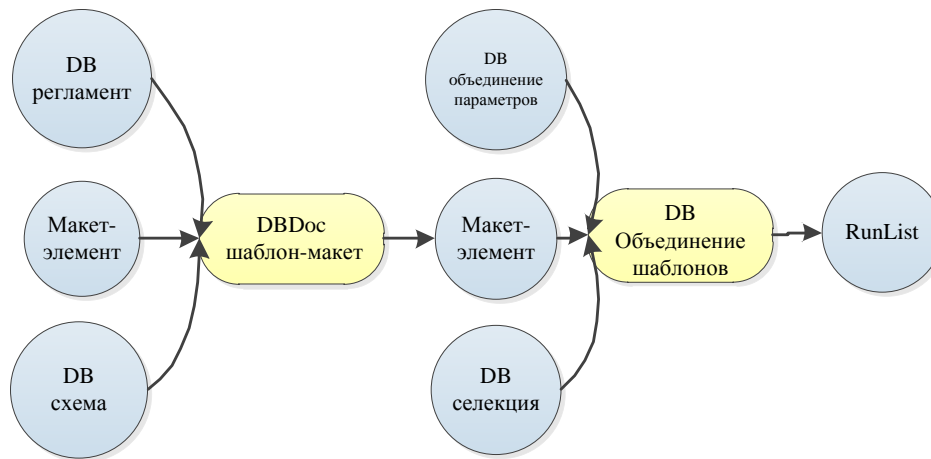


Рис. 10 .17 JDF - модель для персонализированной печати

В ресурсе *LayoutElement* описаны отдельные элементы контента, необходимые для оригинала в качестве переменных элементов, они указаны в URL. Процесс *DBDocTemplateLayout* производит новый ресурс *LayoutElement*, представляющий оригинал. Однако, чтобы обозначить его как оригинал, в атрибут *Template(шаблон)* необходимо ввести значение *True (верный)*. Цель процесса *DBTemplateMerge* – перевести оригинал на VDP - язык, например, PPML. Для этого необходима информация, где в файловой системе должен/должны храниться файл(ы), который(е) можно найти в *DBMergeParams*. В ресурсе *DBSelection (выбор)* содержатся URL - базы данных, индексы наборов баз данных и доступ к базам данных на специфическом языке базы данных (например, SQL).

Глава 11. Отделочные процессы

До того как был разработан формат JDF, в типографиях уже имелись решения для отделочных машин для его использования. Например, скоростная резальная машина получала команды через задания в формате PPF, как было описано в параграфе 4.2. Были разработаны и индивидуальные решения, при которых самые важные данные для фальцевания (формат листа, схема фальцовки и вид фальца) кодировались для выполнения заказа в коде EAN, данные отображались в паспорте заказа и считывались с помощью считывателя штрихового кода в фальцевальной машине. Кроме того, были и другие решения. Для бумагорезальной машины или для фальцевальной машины готовилась программа резки или фальцовки. Однако, чтобы не загружать программные модули этих машин работой по подготовке заданий, использовались и используются для этих целей специальные компьютеры, в которых установлены программы описания заданий для оборудования и затем они через сеть передают на него необходимые данные. Это приводит к сокращению времени наладки машин. Кроме того, обособленные компьютеры собирают соответствующие заказу данные характеристик производственных машин (резальных и фальцевальных, вкладочно – швейно – резальных автоматов и др.): время наладку, статус машины, производительность и так далее. К тому же производители машин разработали в плане автоматизации собственные решения, а также протоколы обмена данными программ подготовительной работы и машинами (как между пультом управления печатной машины и самой печатной машиной). Однако такие решения широко не распространены. Следует отметить, что сегодня к применению формата JDF для управления оборудованием в отделочных процессах относятся с некоторой осторожностью.

Для создания интерфейсов в системе JDF/JMF – рабочего потока отделочного производства имеются два кандидата:

- консоль управления машин послепечатного производства;
- отдельные станции для подготовительной работы для одной или нескольких отделочных машин.

В первом случае пользователи, использующие CIP4, могут применять JDF - модули непосредственно на производственной машине. Во втором случае готовые производственные программы с отдельного сервера передаются через локальную сеть в

машины, где производственные данные JMF – сообщения и JDF – структуры преобразуются в специфические команды производителя машин. Как видим в обоих случаях формат JDF можно использовать для задания программ работы машин.

Рис. 11.1 схематично показывает обе конфигурации. Различие между обоими решениями состоит, прежде всего, в том, что во втором случае отдельная станция может сортировать заказы и при необходимости распределять их на различные машины. Благодаря этому разгружаются операторы машин. В первом варианте при JDF/JMF - интерфейсе, который установлен на машинах, последовательность производства должна задаваться непосредственно информационно – управляющей системой (MIS) или оператором. При определенных обстоятельствах возникает необходимость обслуживать несколько JDF/JMF - интерфейсов, если они встроены в производственные машины, и из-за этого обновление программного обеспечения может быть затруднено.

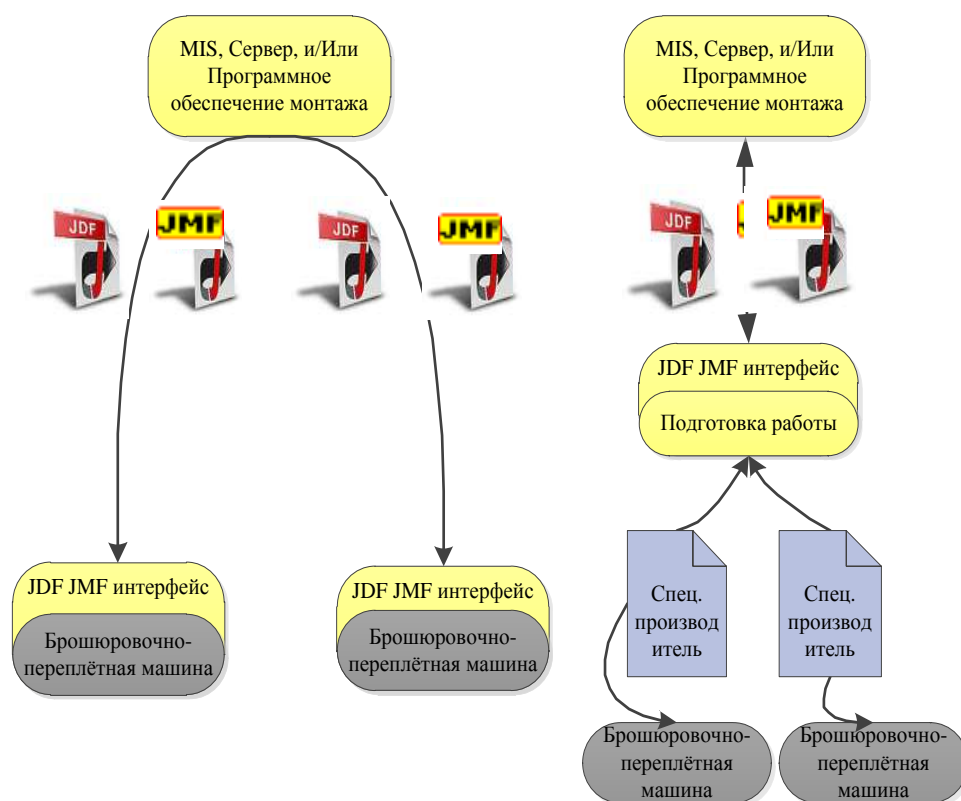


Рис. 11.1 Два варианта конфигурации JDF-/JMF – интерфейсов в отделочных процессах

Примеры совместимого с JDF продукта фирм(2009)	
Производитель:	Основной продукт:
Ferag AG	iQ
Heidelberg Druckmaschinen	Prinect Postpress Manager
Hohner Maschinenbau Gmbx	ixFrame (der ixact Gmbx)
Horizon International Inc.	i2i
MBO	Datamanager
Muller Martini	Connekx

С другой стороны, решение с наличием станции для подготовки работ имеет смысл лишь в том случае, если с её помощью можно управлять несколькими производственными машинами одного производителя.

Оборудование процессов послепечатной обработки получают JDF/JMF - информацию либо от MIS, сервера рабочего потока, либо непосредственно от программного модуля машины. Возможны конфигурации, при которых установочные данные сначала передаются в MIS или в сервер рабочего потока, чтобы они со своей стороны управляли брошюровочно-переплётным оборудованием. Другая возможность состоит в том, что информационно – управляющая система задает все процессы. Тогда MIS непосредственно может задавать стандартные размеры листов, их обрезки и фальцовки и для операторов не имеется возможностей для изменения заданных параметров. Это применимо при выпуске однотипной продукции.

11.1 Одноножевая бумагорезальная машина.

Незапечатанные бумажные листы, а также промежуточные продукты (как, например, оттиски), режутся, как правило, резаком, называемым также одноножевой бумагорезальной машиной. Обрезка бумажного листа перед процессом печати служит с одной стороны для того, чтобы получить необходимый его формат, с другой стороны, чтобы получить правильный угол или параллельность для одного или нескольких печатающих и отделочных устройств. И, напротив, оттиск разрезается для того, чтобы получить несколько листов, фальцуемых в тетради, из большого листа, а также для того, чтобы привести формат продукта к заданному формату при трёхсторонней обрезке после фальцовки.

В записи JDF обозначение ресурса незапечатанных листов это *Tun- Media*, промежуточные, а также конечные продукты - это *Tun –Component*. Таким образом, процесс резки (*Cutting*) получает в виде ввода либо ресурса *Media*, либо ресурса

Component и производит тот же самый тип ресурсов в качестве вывода. На рис.11.2 можно увидеть эту модель.

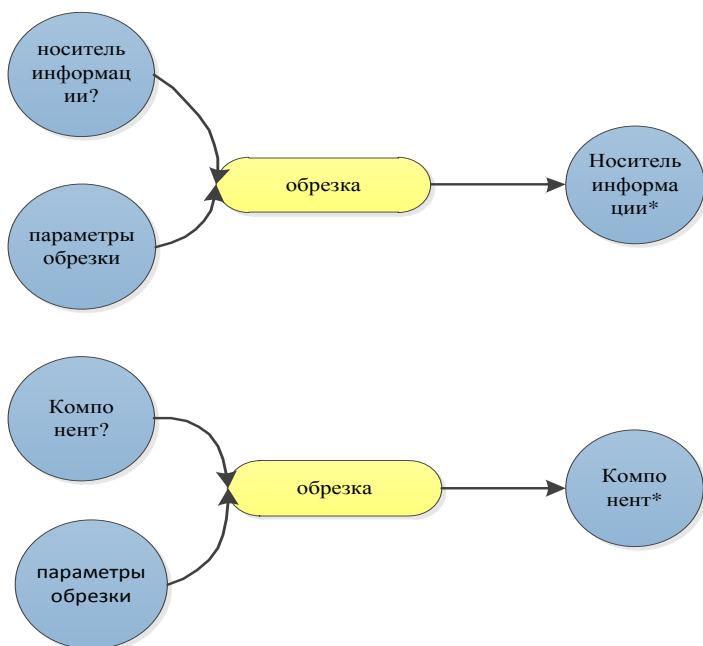


Рис. 11.2 JDF - модель процесса обрезки несфальцованного листа (сверху) и соответственно промежуточного - или конечного продукта (внизу)

Вопросительные знаки, которые стоят при ресурсах *ввода* (*Input-Ressourcen*), означают опциональность и должны интерпретироваться таким образом, чтобы точно одна из двух опций была выполнена обязательно. Ресурсы *вывода* (*Output-Ressourcen*) обозначены звёздочкой, которая стоит или у нескольких ресурсов или ни у одного. Также следует добавить, что можно ввести только ресурс вывода, который по типу идентичен ресурсу ввода, и должен присутствовать. Самая важная для процесса резки информация находится в *параметрах резки* (*CuttingParams*). Как в формате PPF, также и в JDF можно определить *блоки обрезки* (*CutBlok*) (см. параграф 4.2 и, в частности, рис. 4.12). При этом только уточняется, где нужно резать, но не устанавливается последовательность резки. Эта информация может устанавливаться в ресурсе *параметры резки* (*CuttingParams-Ressource*) альтернативно к блокам резки. Каждое описание резки - это элемент типа *резка* (Typ *Cut*). В прежних версиях формата JDF можно было записывать позиции меток резки в ресурсах *параметры резки* (*CuttingParams-Ressourcen*), однако, с версии 1.3 эта информация содержится в ресурсе *Layout-Ressource*, как мы видели на рис. 10.7 для контрольного элемента печати.

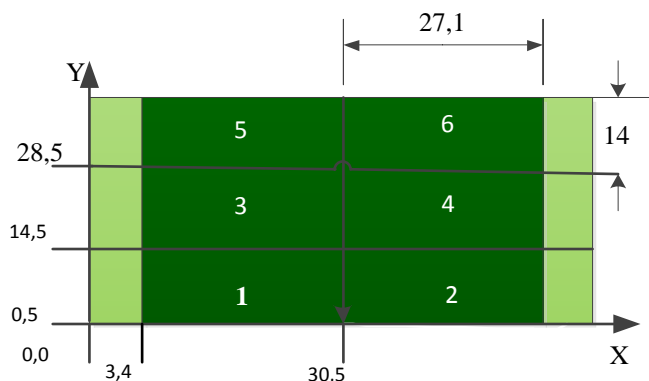
Рис. 11.3 раскрывает код JDF - ресурса *параметры резки* (*CuttingParams-Ressource*), который содержит шесть блоков резки, а именно для каждой обложки один блок. Для

наглядности на рисунке отсутствует несколько свойств, а все значения ограничены двумя числами после запятой - в реальности значения вносятся с более высокой точностью. Размер блока 768.18 x 396.84 DTP пунктов соответствует 271 x 140 мм. Размер блока охватывает лицевую и оборотную стороны, корешок брошюры и 1 см обрезки сверх конечного формата. С помощью атрибута *BlockTrf* задается позиция каждого блока. Для матрицы, которая стоит в значении атрибута применено определение как на рис.9.21.

```
<CuttingParams Class="Parameter" ID="_77" SheetName="Umschlag"
Status="Available">
<CutBlock BlockName="Umschlag_1" BlockSize="768.18 396.84"
BlockTrf="1.0 0.0 0.0 1.0 96.37 14.17" />
<CutBlock BlockName="Umschlag_2" BlockSize="768.18 396.84"
BlockTrf="1.0 0.0 0.0 1.0 864.56 14.17" />
<CutBlock BlockName="Umschlag_3" BlockSize="768.18 396.84"
BlockTrf="1.0 0.0 0.0 1.0 96.37 411.02" />
<CutBlock BlockName="Umschlag_4" BlockSize="768.18 396.84"
BlockTrf="1.0 0.0 0.0 1.0 864.56 411.02" />
<CutBlock BlockName="Umschlag_5" BlockSize="768.18 396.84"
BlockTrf="1.0 0.0 0.0 1.0 96.37 807.87" />
<CutBlock BlockName="Umschlag_6" BlockSize="768.18 396.84"
BlockTrf="1.0 0.0 0.0 1.0 864.56 807.87" />
</CuttingParams>
```

Рис. 11.3 Определение блоков резки

Пункты DTP переводятся в сантиметры и получается расположение соответствующее рис.11.4.



1,2,3,4,5,6 - обложки

Рис. 11.4 Позиция разрезки блоков на печатном листе

Трёхсторонняя резка позволяет получить конечный формат. Однако этот рабочий процесс описывается не *процессом резки (Gutting-Prozess)*, а процессом *подгонки*

(*TrimmingParams*). Соответственно вместо параметров резки (*CuttingParams*) будут параметры подгонки (*TrimmingParams*). В нашем примере был бы занесён конечный формат 120 x 120 мм.

сигнатура с 24 страницами
текущий номер из 24 страниц
3 части вдоль
4 части поперек

короткий край листа
длинный край листа
порядковый номер сгиба
готовый формат сфальцованного листа

устройство
фальц вверх
фальц вниз

фальц вверх
после 1/3 длинного края лист
фальц вниз
после 1/3 длинного края листа
направление фальца поворачивается на 90°
фальц вверх
после половины более короткого края листа
фальц вниз
после одной четверти более короткого края листа

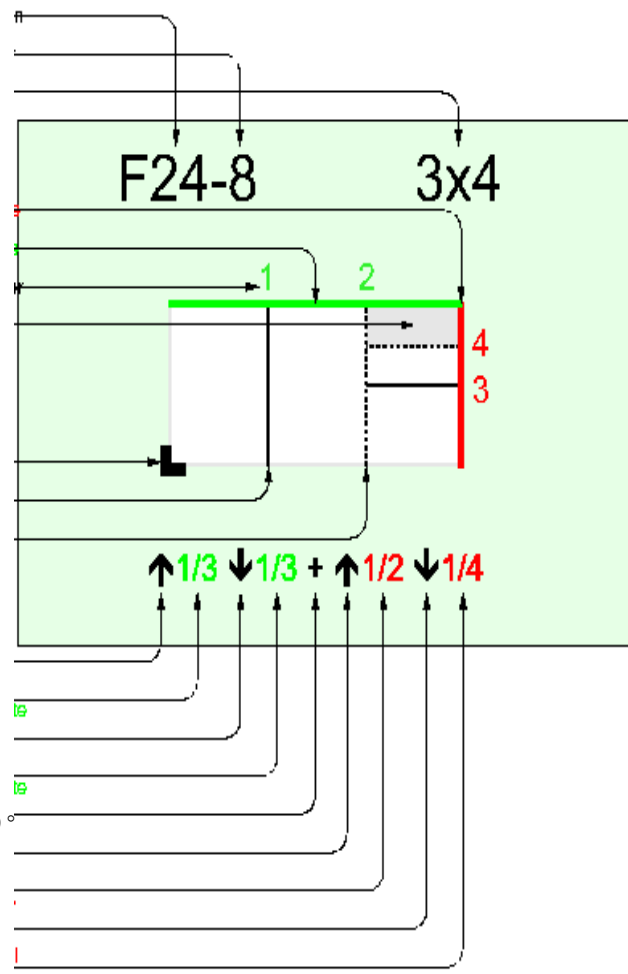


Рис. 11.5 F24- 8 из каталога видов фальца спецификации 1.4

11.2 Фальцевальная машина.

В современных фальцевальных машинах многие действия, которые раньше устанавливались исключительно вручную, теперь могут задаваться в окне диалога и затем автоматически регулироваться. В качестве примера можно назвать положение бокового упора в накладе, регулировку валиков в соответствии с толщиной бумаги или позицию переднего упора при кассетной фальцовке. Другие установки не управляются автоматически, например, такие как подача воздуха для разделения листов. Но и здесь было бы допустимо, чтобы предварительная установка на определенные сорта бумаги или даже специальной бумаги сохранялась и в последующем, на основе информации о выполненном заказе или JDF - данных вызывалось необходимое программное задание и устанавливались соответствующие рабочие команды.

Обычно в фальцевальных машинах с автоматической установкой формат листа и вид фальца вводится в меню задания, после чего электронная система наладки проводит их регулировку. После пробного фальцевания происходит точная подрегулировка, то есть, за несколько итераций проверяются и корректируются отклонения от меток фальца. Так как сначала в качестве параметров задается только формат листа и вид фальца, необходимо корректировать не только смещение бумаги, но и ошибки в предшествующих технологических операциях, вызванные печатью. Учету подлежат и различия отдельных полосных длин посредством соответствующего позиционирования переднего или заднего фальца, заданного при спуске полос (расположение впереди или сзади захвата). Другими словами, данные номера из каталога и формат фальцуемого листа, в принципе, недостаточны (рис. 11.5) и точная юстировка может оказаться гораздо дороже, чем основная установка.

Подобная дополнительная регулировка необходима, если в JDF - задании внесена только схема фальцовки и формат листа. Однако некоторые системы задают точные позиции фальцовки и последовательность в виде JDF - информации, где высчитывается упомянутая позиция впереди или сзади. В этом случае затраты на дополнительную регулировку значительно меньше.

На рис. 11.6 устанавливается вид фальца для 24-страниц с номером фальца по каталогу F24-8 в ресурсе *параметры фальцовки (FoldingParams)*. Во многих случаях это вся информация, которая передается. В данном примере дополнительно уточняются позиции фальца в виде *фальцевальных элементов (Fold-Elementen)*. Каждый *ресурс фальцовки (Fold-Ressource)* определяет одну фальцевальную операцию, причём последовательность *элементов фальцовки (Fold-Elemente)* устанавливает и последовательность фальцовки.

```
<FoldingParams ID="FOLD_A-1" Class="Quantity" Status="Available"
FoldCatalog="F24-8" SheetLay="Left">
<Fold From="Front" To="Up" Travel="943.9370079811025"/>
<Fold From="Front" To="Down" Travel="1887.8740161811025"/>
<Fold From="Left" To="Up" Travel="907.0866141732283"/>
<Fold From="Left" To="Down" Travel="1360.6299241732283"/>
</FoldingParams/>
```

Рис.11.6 Параметры фальцовки

Если, как в нашем примере, номер фальца по каталогу и отдельные позиции фальца указаны в агенте/контроллере, то предпочтительными являются *элементы фальцовки* (Fold-Elemente) с JDF - устройства. Таким образом, требуется JDF -спецификация. Атрибут (From) - от указывает положение, с которого начинается фальцовка, атрибут (To)- на указывает направление фальцовки. На рис. 11.7 можно увидеть задание краев, переднюю и тыльную стороны. Значения координат, как обычно, указаны в пунктах DTP.



Рис.11.7 Обозначение краев фальцуемого листа

При первой операции фальцовки в примере на рис. 11.5 лист фальцуется при $x = 33,3$ см (соответствует 2,54-943,937007981102 5:72) в y - направлении, причем передняя часть кладётся над тыльной частью. Третий шаг фальцовки происходит при $y = 32$ см, причем левую часть листа необходимо положить на правую часть листа.

Если позиция фальца внесена в *параметры фальцовки (FoldingParams)*, то это является только предварительной установкой. Так как не только уже упомянутое смещение бумаги может привести к тому, что "теоретические" значения должны быть скорректированы, на них влияет и вес бумаги и производительность фальцевальной машины, которые могут изменять координаты фальца. И до тех пор, пока не производится автоматически регулировка выравнивания, оператор должен соответствующим образом проводить постоянно подрегулировку фальцевальной машины.

Часто фальцевальные машины, в зависимости от заказа, комплектуются из различных модулей. Если в JDF - задание вводится вид фальца или загружается ресурс *параметры фальцовки (FoldingParams-Ressource)*, который нельзя реализовать в существующей конфигурации, то непосредственно в фальцевальной машине появится сообщение об ошибке. Сотрудники, которые вносят задание, должны знать возможности фальцевальной машины, а также конфигурацию для выстраивания последовательности ее загрузки заказами. Эффективное использование техники требует как учета возможностей программного обеспечения, так и конфигурации машин и соответственно возможностей JDF-задания.

Это не так тривиально, как кажется на первый взгляд, так как продукт характеризуется совокупностью параметров, которые важны для того, чтобы решить, может ли он быть сфальцован на определенной машине или нет. Так, например, фальцевальная машина может обрабатывать бумагу 250 г / м^2 , а также проводить 6-операционную фальцовку, но не то и другое одновременно. Имеется JDF-спецификации относящиеся к этой проблеме под понятием *возможности устройства (Device Capabilities)*. При этом устройство может определять, какие узлы, элементы, свойства и их значения, ресурсы и JMF – файлы оно поддерживает. Так, например, максимальный формат сфальцованного листа может передаваться в MIS или производственный контроллер. Однако возможности для построения системы гораздо обширнее и могут быть заданы многие другие параметры производственного процесса. Кроме того, могут быть сообщены специфические для JDF - данных детали, а именно могут ли устройства программного обеспечения *Grayboxen* обрабатывать комбинированные процессы или общие групповые узлы процесса. На рис. 7.3 была приведена принципиальная ситуация на эту тему.

11.3. Вкладочно - швейно – резальный агрегат.

Вкладочно - швейно – резальный агрегат состоит из трёх модулей:

- сборочный модуль;

- швейный аппарат с обратным стежком;
- триммер для трёхсторонней обрезки.

При комплектовании блока разные сфальцованные листы (тетради) вставляются друг в друга, так что образуется блок. В основном, установленные по порядку самонаклады, заполняются вручную или автоматически, причем необходимо соблюдать правильную последовательность загрузки тетрадей. На предприятии сфальцованные листы разъединяются, открываются в середине и кладутся на затл.

В то время как тетради транспортируются от одной секции самонаклада к другой, они последовательно укладываются друг на друга. Во время последнего процесса, при необходимости, на блок накидывается обложка. Количество разных сфальцованных листов (тетрадей), которые собираются, естественно, зависит от объема печатного продукта, а также от количества имеющихся в распоряжении секций. Наладка вкладочно - швейно - резального агрегата включает в себя установку имеющихся секций на формат сфальцованных листов.

В швейном аппарате с обратным стежком (по-другому сшивальная машина) проволочная скоба, закрепленная в головке, прокалывает тетради, а ножки загибаются. При этом количество, позиции, ширина и формы проволочной скобы могут варьироваться от заказа к заказу. В блоке каждая скоба устанавливается собственной головкой, которая отрезает на правильную длину швейную проволоку, идущую с катушки, подгибает скобу, проталкивает ножки скобы через корешковое поле и, наконец, загибает ножки и заканчивает работу.

Задача триммера для трёхсторонней обрезки: обрезать скрепленный печатный продукт (блок и обложка) на конечный формат и при необходимости одновременно разрезать фальцы, чтобы можно было раскрывать страницы. Для этого обрезаются три стороны (верхняя часть, нижняя часть, передняя часть), кроме корешка. Некоторые машины для трёхсторонней обрезки делают возможным также разрезку блока посередине (при работе с двойниками), что приводит к увеличению тиража выпускаемой продукции.

Как и ожидается, в JDF - файлах важно представить три процесса для вкладочно - швейно - резального агрегата: *сборка (Collecting)*, *сшивание (Stitching)* и *обрезка (Trimming)*. Так как все три модуля машины, как правило, соединены в единое целое, то целесообразно описывать эти три процесса комбинированным процессом или групповым процессом.

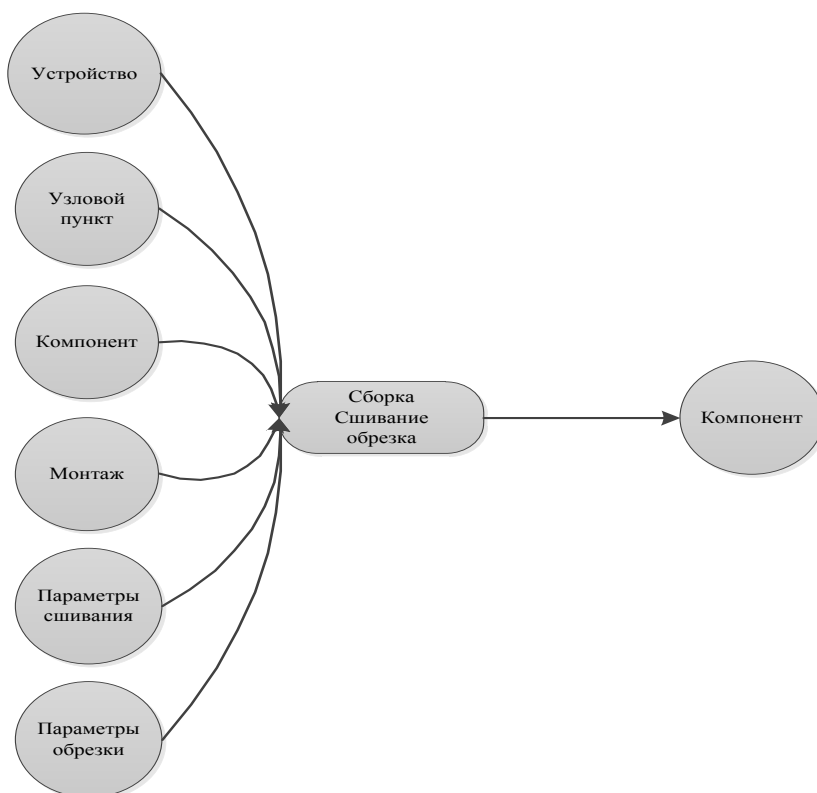


Рис.11.8 Узел *GrayBox* сборки тетрадей (объединение в блок, шитье, трехсторонняя обрезка)

На рис.11.8 представлен узел *GrayBox* с ресурсами ввода и вывода. Ниже коротко поясним эти ресурсы:

- ресурс *устройство (Device-Ressource)* содержит наименование устройств и производительность вкладочно-швейно-резального агрегата;
- в *узловом ресурсе - режим (ModeInfo)* отражено ориентировочное время наладки и производства;
- *компонент (Component)* содержит описание сфальцованных листов;
- *монтаж(Assembly)* определяет последовательность наложения сфальцованных листов;
- в *парамetre сшивания (StitchingParams)* отмечается количество и форма скоб;
- *параметр обрезки (TrimmingParams)* содержат в качестве информации размер конечного формата печатного продукта, на который происходит обрезка.

Собственно для *процесса сборки (Collecting-Process)* необходим также ресурс *параметры сборки (CollectingParams-Ressource)*, он представлен также в JDF -

спецификации - разумеется, без свойств и подэлементов. Он является только контейнером для расширения и не выполняет другие функции. Последовательность собираемых сфальцованных листов определяется не в этом ресурсе, а в ресурсе *монтаж* (*Assembly-Ressource*). В него внесен элемент *секции монтажа* (*AssamblySection-Elemente*), а их последовательность указывает на последовательность наложения тетрадей, причем первый элемент представляет внешнюю позицию, а последний элемент - внутреннюю позицию в печатном продукте. Рис.11.9 представляет эту ситуацию с тремя сфальцованными листами блока и одним для обложки. При *селективном подборе* (*Selective Binding*), когда сфальцованные листы собираются переменным способом, процесс *сборки* (*Collecting-Process*) получает еще дополнительные ресурсы ввода (*DBправило-DBRules* и *DBвыбор-DBSelection*). Тем самым можно задавать различные варианты последовательности для сборки отдельных печатных продуктов.

```

<Assembly Class="Parameter" ID="_019069" Order="Collecting"
Status="Available">
  <AssemblySection AssemblyIDs="Cover_B_1"
DescriptiveName="F04-01_ui_2x1_1" />
  <AssemblySection AssemblyIDs="Text_1_B_2"
DescriptiveName="F08-07_li_2x2_2" />
  <AssemblySection AssemblyIDs="Text_2_B_3"
DescriptiveName="F08-07_li_2x2_3" />
  <AssemblySection AssemblyIDs="Text_3_B_4"
DescriptiveName="F08-07_li_2x2_4" />
</Assembly>

```

Рис. 11.9 Последовательность сборки сфальцованных листов

В заключение приведем пример ресурса – *сборка Assambly-Ressource*, который на рис.11.9 представлен упрощенно. JDF - дерево показано на рис. 11.10. В левом столбце представлены основные составляющие допечатных процессов: *проверка* данных на запуск производственного процесса (*Preflight*), *конвертация* данных Postscript в PDF (*PSToPDFConversion*), *трансформация* цветового пространства (*ColorSpaceConversion*), *наложения* краски (*Trepping*) и *изготовление* монтажного листа (*Stripping*). Вначале делают записи этих процессов через *Grayboxen допечатной подготовки* (*PrepressPreparation*) и подготовки *схемы спуска полос* (*ImpositionPreparation*). Средние два столбца на рис.11.10 показывают разные процессы и, соответственно, комбинированные процессы для частичных продуктов «обложка» и «содержание – блок». Два верхних комбинированных процесса содержат название процесса *предварительная генерация* (*PrievierGeneration*), но их функции различные.

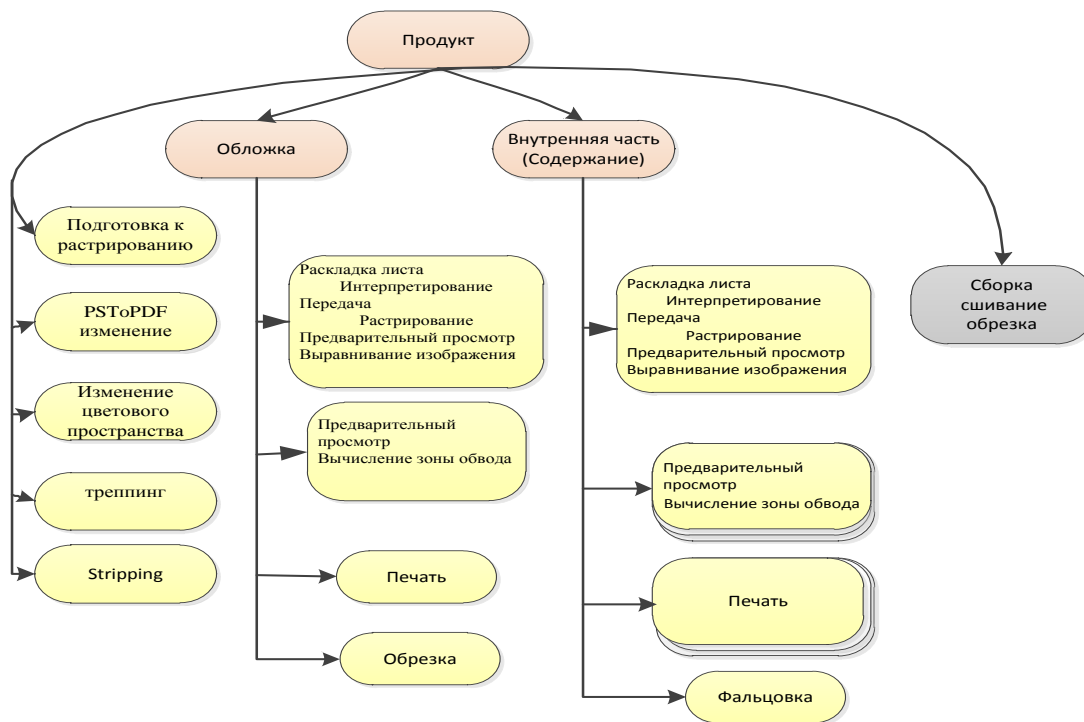


Рис. 11.10 JDF – дерево процесса

В то время как в комбинированном процессе при *введении установок изображения (Imposition ImageSetting)* определяется заставка, которую позже можно увидеть на мониторе пульта управления печатной машины, то в другом комбинированном процессе (*PreviewGeneration InkCalculation*) ввод производится только для расчета *предварительной настройки красочных зон* (обычно отдельные картинки в 50.8 ppi). Так как на печатном листе находится несколько обложек, то после процесса печати следует процесс разрезки. Листы, собираемые в блок, фальцуются.

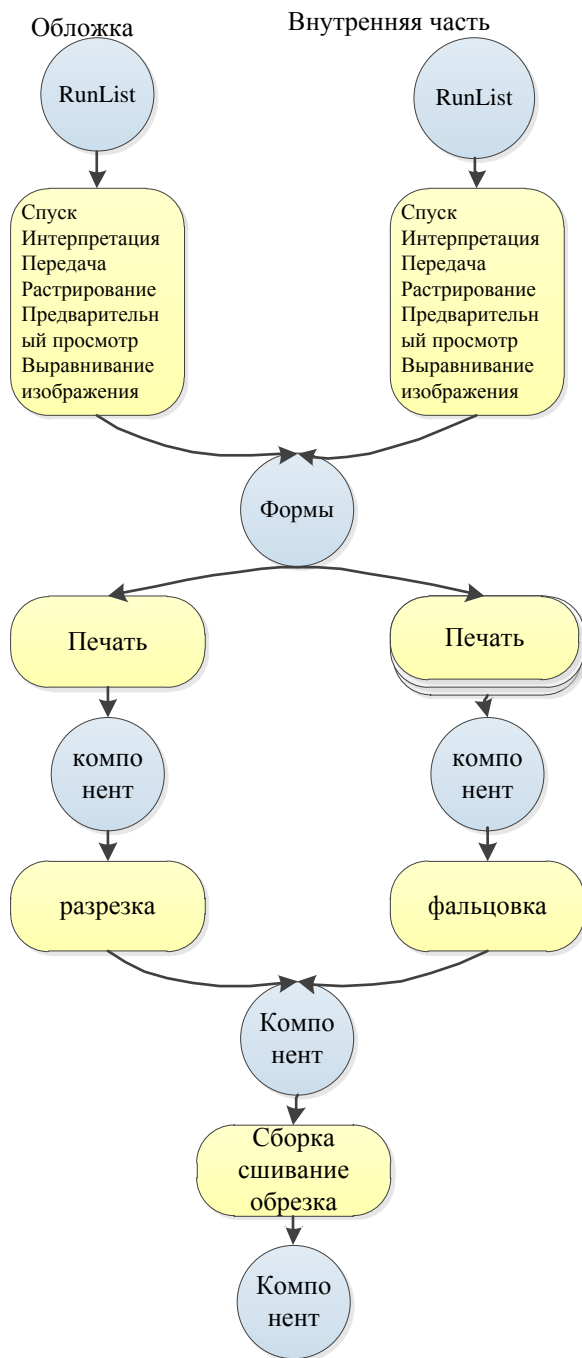


Рис.11.11 Производственный процесс

Рис. 11.11 изображает ещё одну часть производственного процесса. В нем называются только самые важные "передаваемые ресурсы" страниц (*RunList*) для *спуска полос (Imposition)*, *формы (ExposedMedia)*, как промежуточное звено между допечатным процессом и печатью и, наконец, разные ресурсы *Компонента (Component-Ressource)* - оттиск, сфальцованный лист и готовый продукт.

В переводе названий процессов фигурирует около 50 послепечатных процессов, из которых в данной главе рассмотрено только пять: *резка, фальцовка, сборка, шитье, техсторонняя обрезка (Cutting, Folding, Colecting, Stitching, Trimming)*. В следующей главе рассмотрим еще один процесс – изготовление высечных форм для штанцевания (*DieMaking*). Другие остаются, к сожалению, вне данной книги.

Задания для самостоятельной работы:

- составьте аналогично рис.11.11 схему процесса для брошюры бесшвейно-клеевого скрепления, обложку которой тиснят золотой фольгой. Найдите при этом необходимые процессы из JDF - спецификации и руководствуйтесь также “ресурсами передачи”;
- создайте дополнительно к рис.11.10 возможное JDF - дерево для брошюры бесшвейно-клеевого скрепления.

Глава 12. Печать упаковки

С точки зрения JDF/JMF - модели нет принципиального различия между печатью упаковки, акцидентной печатью и другими областями печати. В печати упаковки естественно применяются многие запечатываемые материалы, такие как картон, пленки, гофрированный картон, фольга и др. Поэтому следует определить лишь несколько новых атрибутов и их значений для отдельных ресурсов по использованию JDF - формата при производстве упаковки. Так уже неоднократно использованный атрибут *тип носителя информации (MediaType)* дополняют такие значения, как *гофрированный картон (CorrugatedBoard)* или *фольга (Foil)*. Кроме того, вводятся атрибуты для описания таких материалов, как *гофр (Flute)*, для гофрированного картона (например, в ресурсах *носитель информации (Media)* и *предназначение носителя информации (Mediainten)*).

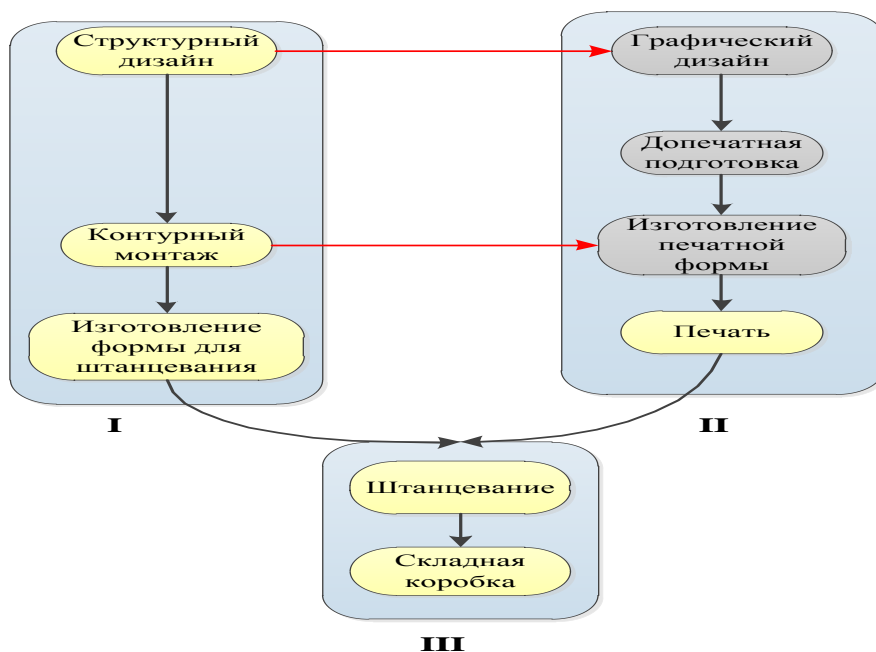


Рис. 12.1 Диаграмма действий для изготовления складной коробки

В JDF 1.4 имеется также дополнительная поддержка флексографской печати. Так, например, для атрибута *тип носителя информации (MediaType)* были введены такие

значения, как *мешок (Sleeve)* и *монтажная лента (MountingTape)*. Для печати с *фотополимерных форм* были обозначены элементы процесса *технологии* их изготовления (*PlateTechnology*) и *высоты печатающих элементов (ReliefThickness)*. Тем не менее, имеются процессы, которые применяются преимущественно только в области упаковки.

В этой главе будут представлены основные процессы для производства упаковки, разделенные на три следующих группы:

- a) конструирование складной коробки (*Strukturdesing*), подготовка, изготовление формы для штанцевания;
- b) штанцевание и склеивание складной коробки;
- c) штрих – код, включая компенсацию ширины штриха, прежде всего при флексографской печати.

На рис.12.1 показана диаграмма действий, причем процессы, представленные в примерах a) и b) можно увидеть еще раз в производственном контексте изготовления складной коробки. Пункт c) - это часть процесса действий по допечатной подготовке.

Действия, выделенные на рис.12.1 желтым цветом, соответствуют JDF - процессу. При создании *геометрической формы продукции (контура)* (*ShapeDefProduction*) с помощью системы автоматизированного проектирования - CAD обычно устанавливаются контуры, штриховые и перфорационные линии для каждого изделия.

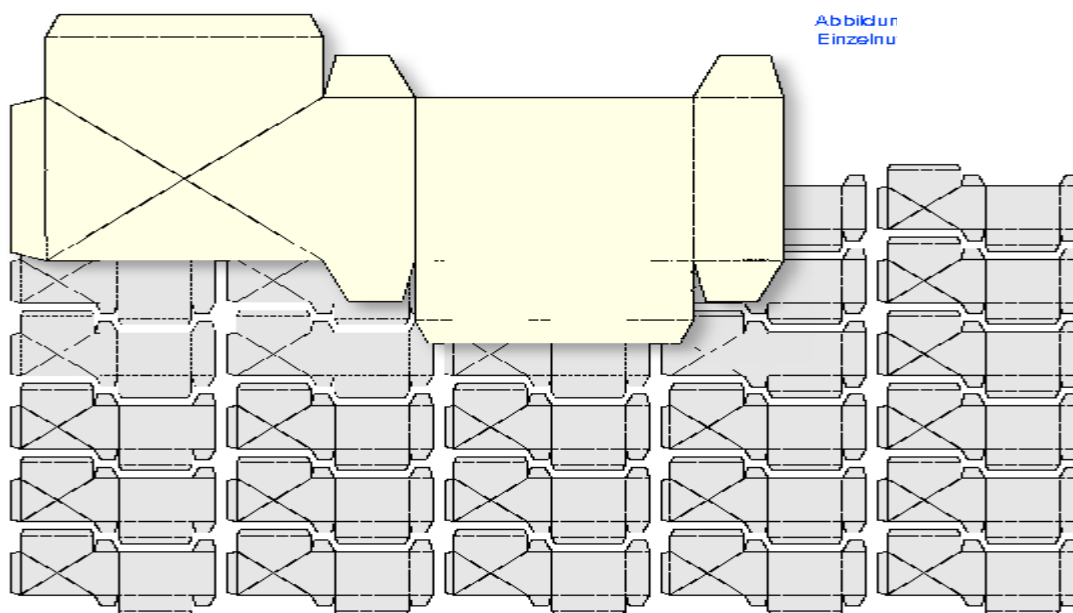


Рис. 12.2 Отдельные единицы изделия на оттиске

Назовем изделием различные виды упаковки. При изготовлении *графического макета (DieLayoutProduction)* одна или несколько единиц изделия располагаются таким образом на листе, чтобы эффективно использовать материал (рис. 12.2) и одновременно производство протекало надёжно и быстро.

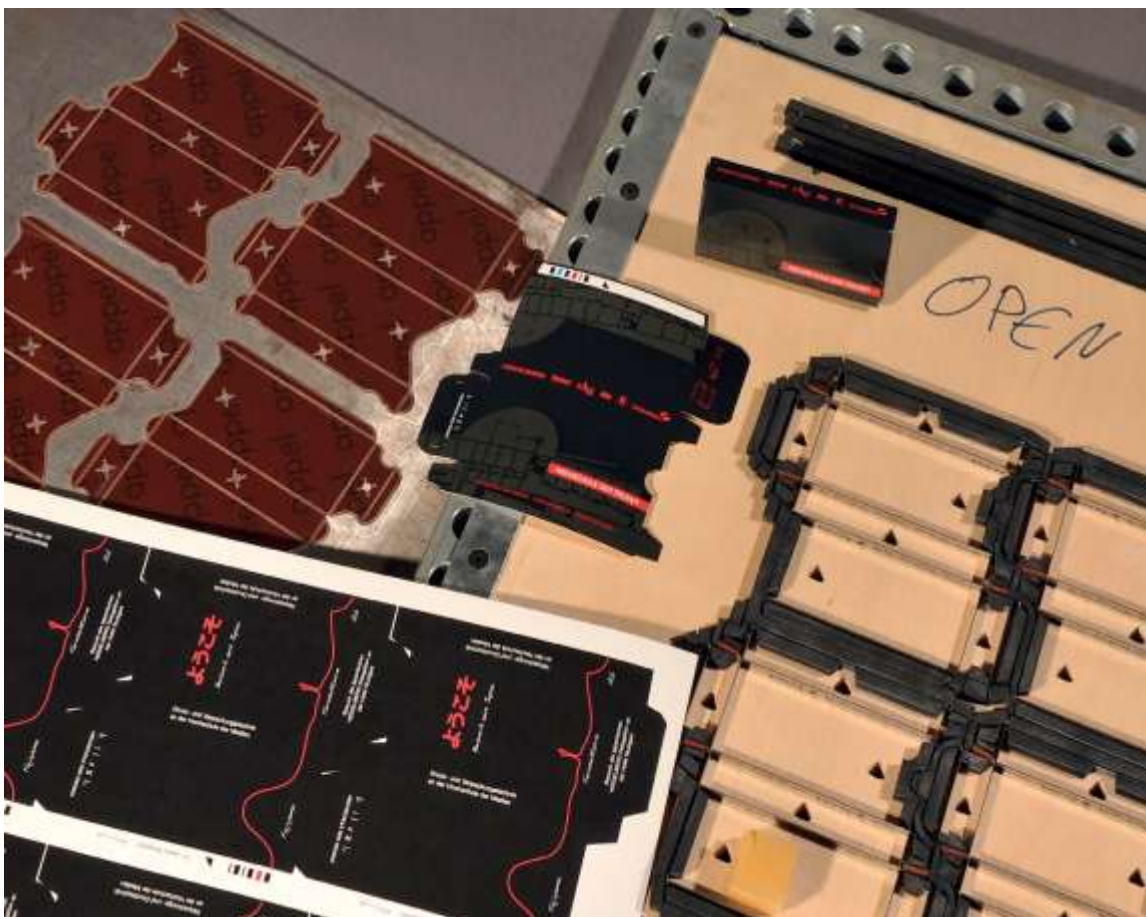
Этот процесс осуществляется, как правило, с использованием той же самой автоматизированной системы конструирования (CAD). Дополнительно указываются и другие элементы, которые важны при штанцевании, как, например, приводочные отверстия для пробивки позиционированным инструментом штанцевальной машины.

Данные CAD, при создании формы для штанцевания, вносятся в соответствующую производственную систему, предусматривающую применение инструментов для штанцевания, причём они могут перечисляться в виде определенного набора (рис. 12.3):

- форма для штанцевания (*CutDie*);
- контрформа/контротделка для штанцевания (*CounterDie*);
- инструменты для съема (*нижнее устройство, верхнее устройство для съёма – Lower Stripper, Upper Stripper*) обрезки (облоя);
- приспособление для хранения изделий (*Blanker*).

Контротделка необходима для биговки штрихов, а пористая резина для отделения отходов (облоя). *Изготовление формы для штанцевания (DieMaking)* включает один или несколько *инструментов (Tool)* для процесса *штанцевания (ShapeCutting)*. В штанцевальной машине часто не только высекается, но и отделяется выкройка. Однако ее отделение может происходить вне штанцевальной машины. Тогда листы с штанцованными выкройками складной коробки поступают в самонаклад машины для склеивания, в которой заготовки отделяются, складываются и закрываются. Этот процесс, таким образом, объединяется с процессом склеивания складной коробки (*BoxFolding*).

Красные линии на рис.12.1 показывают определенные потоки данных. Для дизайна складной коробки необходима информация о контурах или штрихах штанцевания. При этом часто берут соответствующий EPS – файл или PDF – запись из системы CAD, и преобразуют с помощью графического программного обеспечения для последующего использования.



.. Рис.12.3 Элементы операции штанцевания

Кроме того, при изготовлении штанцевальной формы необходимо так расположить изделия (вкопировывание и повторение), чтобы позднее штанцевание и штрихи на оттиске точно совпали с размером каждой единицы продукции. Так как позиции заранее устанавливаются системой CAD, а не при изготовлении печатной формы, то позиционные данные системы CAD представляются часто в форме CFF2-, DXF файлов или в файлах DDES. Они пересылаются как изготовителю формы для штанцевания, так и производителю печатной формы.

Причина того, почему осуществляется пересылка в два адреса на стадии до изготовления печатной формы (не так, как это происходит со страницами в акцидентной печати) состоит в том, что производство формы для штанцевания требует довольно много времени, тем более, что она готовится, как правило, не в типографии. Проектирование на основе применения системы CAD находится в начале всего процесса, в то время как данные на изготовление печатной формы подготавливаются незадолго до начала печати.

При производстве простых этикеток (*Labels*), которые ограничиваются прямоугольником (*Bounding Box*), информацию для вкопирования и повторения предоставляет иногда информационно – управляющая система (*MIS*) и передаёт ее как для производства формы для штанцевания, так и изготовления печатной формы. Для складных коробок, которые выкладываются комплексным способом на печатный лист, монтаж в каждом случае должна выполнять система CAD или другое программное обеспечение.

12.1 Дизайн формы для штанцевания, монтаж и изготовление формы для штанцевания.

На рис. 12.1 представлена схемы изготовления коробки: в варианте (I) дизайн и штанцевальная форма решены с использованием системы CAD, вариант (II) предусматривает операции изготовления продукции до печати и проведения последующей обработки (блок III). Процессы в (I) были выполнены с применением только JDF - спецификации 1.4 и являются в настоящее время еще недостаточно реализуемыми в JDF-рабочем потоке. Поэтому примеры основываются не на производственных данных. Процессы производства коробки (совместно II и III), представлены уже несколько лет в спецификации 1.4 и реализуются на рынке (хотя и немного) теми, кто интересуется использованием JDF - формата.

На рис. 12.4 процесс производства коробки по схеме рис. 12.1 (I) еще раз представлен как процесс-ресурс-модель в записи JDF - описания. Для процесса начала производства конструкции формы (*ShapeDefProduction*), как и CAD -конструирования складной коробки или другого вида упаковки, имеется два ресурса ввода. В опциональный элемент-макет(*LayoutElement*) обычно внесён эскиз, согласно которому должна быть выполнена упаковка. Записью может быть, к примеру, унифицированный указатель ресурсов (URL) EPS - файла, в котором сохранён эскиз. Сам эскиз или разработка задается в большинстве случаев заказчиком или рекламным агентством. В ресурсе *параметры производства конструкции формы (ShapeDefProduction-Params)* может быть записан унифицированный указатель ресурсов (URL) и описание 3-D модели упаковки. Кроме того, можно внести название стандарта упаковки, например, из каталога FEFCO (Европейская Федерация производителей гофрированного картона) или ЕСМА-каталога (Европейская ассоциация производителей картона).

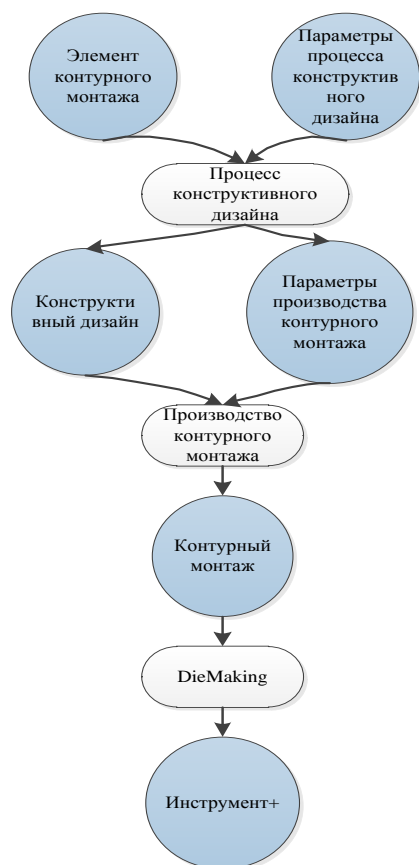


Рис. 12.4 Диаграмма действий по изготовлению формы для штамцевания

Выводом процесса *производства конструкции формы* (*ShapeDefProduction-Prozesses*) является описание геометрии складной коробки (*SapeDef*), причем может быть рекомендован либо внешний файл, который содержит контуры, либо сами контуры могут быть внесены в ресурс. В последнем случае прямые линии и кривые Безье указываются в качестве значений свойства *траектория реза* (*CutPath*). Кодирование прямых и кривых осуществляется как в PDF (параграф 4.4.1 [5]).

Рис. 12. 5 иллюстрирует еще раз две возможности. В первом случае это CFF2- файл, представленный как справка, во второй записи специфицируются как направление. Ресурс процесса *форма* (*Shape*) определен как тип *направление* в атрибуте *траектории* (*CutPath-Attribut*) в PDF-формате. С помощью оператора *m* курсор перемещается на определенную позицию, с помощью оператора *l* проводится прямая и, наконец, с помощью оператора *c* в конце прямой процесс переходит к кривой (*m* для *moveto*, *l* для *lineto*, *c* для *curveto*). Операнды могут быть при этом всегда заданы оператором в списке. Атрибут *DDESCut-Type* указывается в соответствии со стандартами ANSI DDES3 [7] как тип реза или перфорации.

Упаковка может состоять из нескольких частей как, например, коробки с крышкой. В этом отношении некоторые ресурсы *конструкции макета* (*ShapeDef-Ressourcen*) могут

быть вводом процесса *производства формы* (*DieLayoutProduction-Prozesses*). Кроме того, на листе могут быть макеты разных упаковок. В ресурсе *параметры производства формы* (*DiaLayoutProductionParams*) может быть дано несколько параметров вкопировывания и повторения, расположения, как единиц изделия, так и полей между ними.

```
<ShapeDef Class="Parameter" ID="_4711" Status="Available">
  <FileSpec URL="file://Fileserver1/CFE2/Faltschachtel.cf2"/>
</ShapeDef>
```

```
<ShapeDef>
  <Shape ShapeType="Path" DDESCutType="101"
  CutPath=" 28 28 m 10 72 l 20 140 144 150 144...">
  </ShapeDef>
```

Рис 12.5 Две различные возможности описания контуров складной коробки

Процесс создания *дизайна контура* (*DieLayoutProduction-Prozess*) определяет контур или несколько контуров и производит ресурс *макет* (*DieLayout*), в котором имеются данные CAD, необходимые для создания штанцевальной формы. При этом регистрируется, в основном, только ссылка на актив, где находится файл. В результате должна быть произведена физическая форма для штанцевания (*Tool*), что описывается в процессе *изготовления формы* (*DieMaking*). Как сказано выше, в производстве применяются не только формы для штанцевания, но и инструменты для контролделки, для разделения единиц изделия, которые не будем далее принимать во внимание.

В результате выполнения рабочих этапах технологии допечатного производства, представленных на рис.12.1(11), должен быть произведен макет со спуском полос и проведено экспонирование формных пластин. В какой-то мере процесс *изготовления формы для итанцевания* (*DieLaout*) аналогичен данному. В главе 8 и 9 был представлен необходимый для этого *процесс* (*Stripping-Prozess*) (рис. 8.7, 8.9 и 9.8). В акцидентной печати на печатном листе с помощью ресурсов *StrippingParams*, *RelativeBox*, *BinderySignature*, *ExposedMedia* и *других* задаются позиции одной или нескольких страниц, а в подэлементе дополнительно записывается схема спуска полос. При производстве упаковки вместо схемы для спуска полос используют либо макет формы для

штанцевания, либо образец вкопировывания и повторения для расположения заданных единиц изделия. К тому же эта информация может быть предоставлена определенными подсистемами MIS, которые со своей стороны получают данные от систем CAD (однако, до сих пор еще без использования JDF).

Рис. 12.6 иллюстрирует пример записи JDF-кода, в котором в ресурсе *отделочного производства* (*BinderySignature-Ressource*) дана ссылка на внешние данные CAD. В параметре (*StrippingParams*}, приведенных в записи, указаны позиции контуров штанцевания (как в примере на рис. 8.8, где на одном листе размещено несколько страниц).

```
<StrippingParams Class="Parameter" ID="_001" Status="Available"
WorkStyle="Simplex">
<Position MarginBottom="36.0" MarginLeft="36.0"
RelativeBox="0.00000 0.00000 1.00000 1.00000" />
<BinderySignatureRef rRef="_4711" />
<StripCellParams Mask="DieCut" TrimSize="744.12 752.04" />
...
</StrippingParams>
<BinderySignature BinderySignatureType="Die" Class="Parameter" ID="_4711"
Status="Available">
<DieLayout>
<FileSpec URL=" file://Fileserver1/DDES3/6-Faltschachteln.dd3" />
<Station StationAmount="6" StationName="DES1" />
</DieLayout>
</BinderySignature>
```

Рис. 12.6 Контур, описываемый данными CAD

Как и ранее, пример содержит подэлемент *параметры поля ячейки* (*StripCellParams*}. Он определяет конечный формат рамки, а также имеет данные о шаблоне Clp-Pfad (*Mask*), который идентичен контуру высебки на форме для штанцевания. Ресурс *отделочное производство* (*BinderySignature-Ressource*) может быть либо непосредственным результатом ресурса *параметры высебки* (*StrippingParams-Ressource*}, (как, например, в 8.9), либо может быть определён альтернативно вне него. В последнем случае параметр *StrippingParams*, естественно, должны содержать ссылку на *параметры отделочного производства* (*BinderyParams*}, как можно видеть на рис. 12.6,

Отметим три элемента *отделочного производства*:

- схема фальцовки (*Fold*);
- вкопировывание и повторение (*Grid*);
- контур для штанцевания (*Die*).

Отметим сразу, что *схема фальцовки* (*Fold*) установлена как невыполненное значение, поскольку это свойство не было представлено в предыдущих главах, оно не будет

представлено и далее. Если есть ресурс типа *контур (Die)*, то он должен содержать ссылку на элемент *контур макета (DieLayout)*. В нем представлен URL файла („6- складная коробка.dd3"), который служит для описания контура для штанцевания. Кроме того, в примере дано количество *секций (Stations)*, что соответствует такому же количеству единиц изделия на печатном листе.

```
<BinderySignature ID="_4712" Status="Available" BinderySignatureType="Grid"
NumberUp="2 3"...>
<SignatureCell FrontPages="0 0 0 0 0 0 " Orientation="Left" />
</BinderySignature>
```

Рис. 12.7 Вкопировывание и повторение на примере этикетки

Рис. 12.7 показывает пример ресурса *отделочное производство (BinderySignature-Ressource)*, в котором расположение единиц изделия определяется через матрицу *вкопировывание и повторение*. Вследствие этого, *тип отделочного производства (BinderySignatureType)* в результате *Grid*, что на немецком языке "решетка". С помощью атрибута *перед страницей (FrontPages)* можно установить количество столбцов и строк матрицы. При этом каждая матричная запись соответствует изображению этикетки. Это же самое свойство можно применить и для акцидентной печати, если есть возможность определить образец спуска полос. Тогда каждая матричная запись соответствует странице (см. параграф 8.2, рис. 8.11). С помощью атрибута *перед страницей (FrontPages)* величины - 0 представляется единица изделия одинакового размера и содержания. Таким образом, оттиск содержит только один вид этикетки (в противном случае нужно было бы определить несколько разновидностей отделочного производства).

12.2 Штанцевание и придание формы складным коробкам.

При штанцевании посредством штанцевального автомата из запечатанного листа высекаются фасонные детали - так называемые выкройки. В зависимости от конфигурации автомата, на соответствующих секциях, посредством инструмента выполняется первоначальная операция высечки, а затем производятся последующие процессы отделения заготовок и придания им определенной геометрической формы.

Упомянутые процессы объединяются в JDF под процессом *раскрой по форме (ShapeCutting)*. Самыми важными ресурсами ввода служат, с одной стороны, названные в последнем разделе *инструменты (Tool)*, а также *компонент (оттиск) (Component)*. Рис.

12.8 показывает модель процесса. Естественно, могут присутствовать и такие вводные ресурсы, которые допустимы для каждого узлового процесса, как, например, *узел* (*Nodeinfo*) или *устройство* (*Device*). Следует указать еще на то, что на штанцевальном автомате можно производить тиснение и даже встраивать в него модули для тиснения горячим штампом с применением фольги. В этих случаях весь процесс может быть описан комбинированным процессом, который охватывает *тиснение* (*Embossing*) и *раскрой по форме* (*ShapeCutting*).

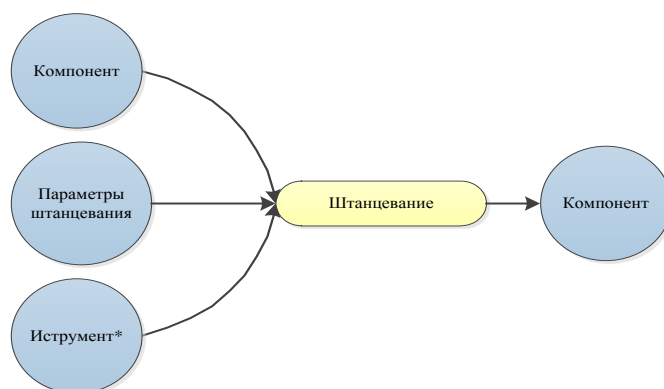


Рис.12.8 JDF - модель штанцевания

Пример процесса раскроя по форме, который частично описан на рис. 12.9, построен проще, чем модель на рис. 12.8. Он не содержит ресурс ввода инструмента (*Tool*), потому что, как уже было отмечено в начале параграфа 12.1, в настоящее время ни одна производственная система не предоставляет эту информацию.

```
<BoxFoldingParams BlankDimensionsX="43 388 502 848 960"
BlankDimensionsY="43 78 156 465 543 578 621" BoxFoldingType="Type01"
Class="Parameter" ID="_771" PartIDKeys="SignatureName SheetName
BlockName"
Status="Available">
<BoxFoldingParams SignatureName="SIG1">
<BoxFoldingParams SheetName="Faltschachtel-Druckbogen">
<BoxFoldingParams BlockName="Block1" />
<BoxFoldAction Action="LongPreFoldLeftToRight"
FoldIndex="0 -1" />
<BoxFoldAction Action="LongPreFoldRightToLeft"
FoldIndex="2 -1" />
<BoxFoldAction Action="LongFoldLeftToRight" FoldIndex="1 -1" />
<BoxFoldAction Action="LongFoldRightToLeft" FoldIndex="3 -1" />
</BoxFoldingParams>
</BoxFoldingParams>
</BoxFoldingParams>
```

Рис. 12.9 Описание процесса штанцевания

Имя блока (*Blockname*) идентифицируется из акцидентной печати - блок резки (*Cutblock*) для процесса обрезки (*Cutting-Prozess*) (рис. 11.3). Для упаковки название блока

отражает единицу изделия, которая высекается. Процесс печати (*ConventionalPrinting*) производит в качестве вывода компонент (*Component*), т.е. стопу оттисков с именем лист: *складная коробка - отпечатанный лист(SheetName)*, из которых потом высекается шесть заготовок коробок - изделий.

Выкройки в производстве помещают в приемное устройство следующей в технологической цепочке машины. В ней по штрихованным линиям заготовки формируются, вновь раскладываются и раскрытыми передаются далее по технологической цепочке. Первичное формирование служит для того, чтобы складные коробки легче принимали необходимую форму в фасовочных машинах.

Часто машины для производства складных коробок комплектуют секциями или модулями для отдельных заказчиков индивидуально. Так, например, имеются специальные модули для складных коробок, которые в определенном смысле особенны. Например, имеются модули для коробок конической формы или коробок разнообразных конфигураций для шоколадных конфет. При этом в одном комплексе оборудования могут устанавливаться компоненты разных производителей. Иногда машины конструируют специально только для одного вида упаковки, т. к. переоборудование машины для производства отдельных видов складных коробок, как правило, связано с заменой многих механических элементов. В случае серийного производства машин, установочные данные в системах управления просто перезаписываются для каждой из них. Однако уже имеются машины, которые настраиваются на определенное изделие автоматически с помощью программного обеспечения системы управления, а не ручным способом.

Наладочные работы для типичной машины по изготовлению складных коробок проводят, по большому счёту, в определенной последовательности. В зависимости от формата и толщины обрабатываемых выкроек производится предварительная настройка оборудования в соответствии с типом и толщиной заготовок. Наряду с выполнением уже отмеченных операций, задаются команды для клеевого аппарата в соответствии с шириной и длиной склеиваемых полос, а также определяется количество клея для выполнения заказа. Естественно, определяются позиции клеевого аппарата. Последовательность операций при этом соответствует процессу производства складной коробки.

JDF/JMF – интерфейс машины для изготовления складных коробок, а также других машин технологической цепи изготовления упаковки, позволяет:

- вводить данные заказа непосредственно в устройство;

- производственные данные (BDE) сохраняются для последующих аналогичных заказов;

- данные предварительной установки сокращают время запуска оборудования.

Однако, третий пункт в реальном производстве играет ещё незначительную роль, так как выпускается мало машин с возможностью автоматической наладки. С перспективами на будущее необходимо подробнее описать возможности использования формата JDF.

Процесс *складная коробка (BoxFolding)* представляет работы, проводимые на специализированной машине. Ресурсы ввода - это, в основном, *компоненты-выкройки (Components)*, которые получились как результат процесса штанцевания, а *параметры коробки (BoxFoldingParams)* – это параметры, которые на машине необходимо установить для выполнения работы. На рис. 12.10 можно увидеть пример, в котором описываются необходимые действия применительно к некой стандартной конфигурации.

```
<BoxFoldingParams BlankDimensionsX="43 388 502 848 960"  
BlankDimensionsY="43 78 156 465 543 578 621" BoxFoldingType="Type01"  
Class="Parameter" ID="_771" PartIDKeys="SignatureName SheetName BlockName"  
Status="Available">  
<BoxFoldingParams SignatureName="SIG1">  
<BoxFoldingParams SheetName="Faltschachtel-Druckbogen">  
<BoxFoldingParams BlockName="Block1" />  
<BoxFoldAction Action="LongPreFoldLeftToRight"  
FoldIndex="0 -1" />  
<BoxFoldAction Action="LongPreFoldRightToLeft"  
FoldIndex="2 -1" />  
<BoxFoldAction Action="LongFoldLeftToRight" FoldIndex="1 -1" />  
<BoxFoldAction Action="LongFoldRightToLeft" FoldIndex="3 -1" />  
</BoxFoldingParams>  
</BoxFoldingParams>
```

Рис. 12.10 Предварительная установка данных для машины по производству складной коробки

В JDF – спецификации определено десять стандартных наборов параметров коробок, кроме того, могут быть заданы и индивидуальные параметры. *Тип складной коробки (BoxFoldingType)* имеет здесь значение тип 01 (*Type01*), что соответствует конструкции складной коробки вида, представленного на рис. 12.11.

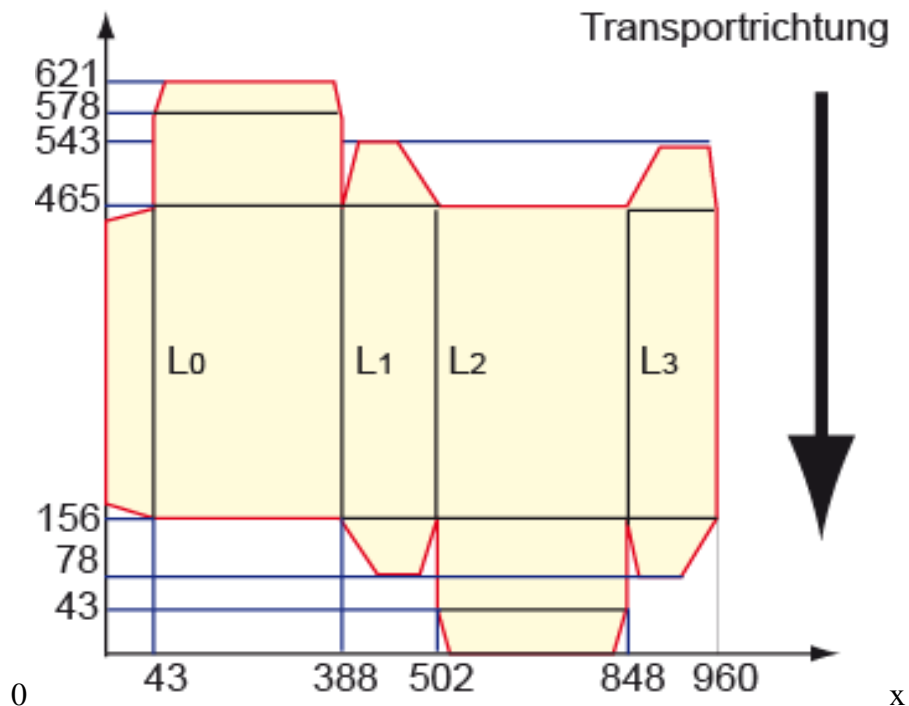


Рис. 12.11 Стандартная складная коробка типа 01

Значения атрибутов *размеров по оси X* (*BlankDimensionsX*) и *размеров по оси Y* (*BlankDimensionsY*) также изображены на рис. 12.11. Нулевой атрибут указан внизу слева, а другие значения указаны в пунктах DTP. Естественно, эти значения могут варьироваться, и описывают здесь только размеры конкретной складной коробки этого типа. Тип коробки определяет операции фальцовки, причем нужно обращать внимание на то, чтобы выкройки вкладывались в приемное устройство запечатанной стороной вниз:

- на линии L_0 необходимо фальцевать предварительно слева направо;
- на линии L_2 необходимо фальцевать предварительно справа налево.

И после нанесения клея снизу на левой клейкой планке необходимо складывать:

- на линии L_1 слева направо;
- на линии L_3 справа налево.

Полоска клея (*GlueLine*) в этом примере не определена, но *элементы процессов* (*BoxFoldAction-Elemente*) точно соответствуют последовательности операций фальцовки. В *индексе* (*Foldindex*) решающим является только первое значение, значение „- 1“

свидетельствует только о том, что складывание должно происходить в направлении подачи по длине.

12.3 Штриховой код.

На первый взгляд кажется, что штриховой код это просто графический элемент на упаковке, который нужно учитывать и выполнять при разработке ее дизайна. И, в действительности, с помощью стандартной программы создания макета можно создавать и размещать штриховой код. Создание штрихового кода поддерживается непосредственно программой конструирования макета, для этой цели имеется программное расширение (плагин). Альтернативно можно применять автономные программы, в которые вводятся цифровые данные для штрихового кода и затем они преобразуются в соответствующее его изображение. Такое изображение полностью соответствует задаваемому образцу штрихового кода и его можно использовать при создании макета упаковки. Так как в печатной индустрии все изображения в настоящее время преобразовываются в цифровые файлы, то и штриховой код передается одновременно производителю печатных услуг. Тем самым штриховой код на упаковке можно создавать уже при разработке ее дизайна, а не передавать его типографиям отдельно от заказа.

Но этот способ имеет недостаток. Так в флексографии при печати с фотополимерных форм происходит растискивание штриха, которое может привести к тому, что код не будет правильно считываться электронным устройством. Это может привести к большим неприятностям, так как не только упаковка, но и продукт могут стать непригодными для реализации. Если ошибка будет замечена после упаковывания, то заказчик понесет большие убытки. Таким образом, необходимо заранее предусмотреть допустимое растискивание ширины штриха при печати. Это зависит от различных факторов (особенности настроек процесса печати, материал для печати, печатная краска и др.), так что только специалисты допечатной подготовки и печати могут заложить необходимые данные.

Штриховой код, как правило, задается окончательно не на стадии дизайна упаковки, а позже корректируется производителем печатных форм с помощью специальных программ. Тем самым штриховой код представляет собой информацию, которая должна особо тщательно передаваться от заказчика до изготовителя. Эта связь может быть облегчена с помощью данных в формате JDF. Типография принимает от заказчика код EAN как последовательность цифр и вносит их в информационно – управляющую систему (MIS). Из нее информация передаётся в формате JDF в программное обеспечение

допечатной подготовки упаковки, что спасает от риска ошибочного ввода этой информации оператором.

С соответствующей записью можно ознакомиться на рис. 12.12.

```
<LayoutElementProductionParams Class="Parameter" ID="_300" Status="Available">
<LayoutElementPart>
<BarcodeProductionParams>
<IdentificationField Encoding="BarCode" EncodingDetails="EAN_13"
Value="0123456789128" />
</BarcodeProductionParams>
</LayoutElementPart>
</LayoutElementProductionParams>
```

Рис. 12.12 Определение штрихового кода

Параметры элементов создания макета (*LayoutElementProductionParams*) образуют ресурс ввода процесс создания макета (*LayoutElementProduction*). В нем описываются компоненты макета, такие как изображения, графики, текст и др., либо полный макет. С другой стороны, MIS не задает детально процессы, как это делает *Graybox*, которая предписывает допечатную подготовку (*PrePressPreparation*) или как (*ContentCreation*).



Рис. 12.13 «Ресурс - Родословное дерево» для процесса создания элемента макета

В нем описываются компоненты макета, такие как изображения, графики, текст и др., либо полный макет. С другой стороны, MIS не задает детально процессы, как это делает *Graybox*, которая предписывает *допечатную подготовку (PrePressPreparation)* или как *(ContentCreation)*. В ее подэлементе генерации - находятся *параметры создания штрихового кода (BarcodeProductionParams)*, которые содержат описание параметров для его воспроизведения на упаковке. Каждому штриховому коду в подэлементе отведено идентификационное поле. В примере указан тип штрихового кода (*EncodingDetails*) и способ задания (*Value*). Этот ресурс может дополнительно содержать другие подресурсы или устанавливать ссылки на другие ресурсы, как можно увидеть на рисунке 12.13. Например, в ресурсе *параметры копии штрихового кода (BarcodeReproParams)*, может быть внесена информация о размерах штрихового кода, а в *параметрах штрихового кода (BarcodeCompParams)* - компенсации ширины штриха, если только они известны.

Как видим, преимуществом автоматизированных методов нанесения штрихового кода на упаковку состоит в возможности обеспечения высокого качества его штрихов. Кроме того, вносимая в систему запись, позволяет сравнивать код на упаковке продукта с записью в банке данных, чтобы проверить, что штриховой код действительно соответствует упакованному продукту.

Глава 13. Проекты по внедрению JDF/JMF

На протяжении двенадцати глав мы опирались на достаточно надежные факты, теперь же разговор нужно вести с некоторой осторожностью. Проекты на основе JDF разнообразны, и методы управления проектами также различаются. Довольно сложно выработать общие правила или советы по внедрению или расширению рабочего потока на базе формата JDF. Мы не хотели бы излишне оптимистично или же негативно отзываться о внедрении JDF. Для облегчения задачи разделим возможные проекты на три типа:

- создание рабочего потока с использованием модулей одного производителя;
- создание рабочего потока с использованием модулей нескольких производителей;
- создание рабочего потока с использованием модулей нескольких производителей программного обеспечения.

Перед обсуждением перечисленных выше случаев хотелось бы высказать общие замечания, даже если читатель отнесется к ним как к воскресной проповеди.

В первую очередь важно перед началом разработки проекта снизить психологический уровень ожидания эффекта от рабочего потока на основе JDF. Что можно и нельзя ожидать от рабочего потока, следует рассматривать индивидуально и в контексте реального предприятия. Чем ниже уровень знаний по этой теме, тем становится сложнее как верно оценить ситуацию, поскольку сложности, как известно, возникают в мелочах. Зачастую ожидаются чудеса, как, например, внезапное превращение плохой организации производства в идеальное производство. Или же печатники бывают разочарованы тем, что от широко обсуждаемого внедрения рабочего потока на основе JDF они не увидели ожидаемого результата. Конечно, бывают случаи, когда все новшества касаются лишь пары предварительных настроек машины. Следует признать, что множество улучшений, которые достигаются сегодня с помощью JDF/JMF, были уже возможны ранее при использовании различных решений от разработчиков полиграфического оборудования. Ранее уже существовали системы сбора и учета производственных данных, межцеховой обмен информацией о заказе и предварительные настройки оборудования. Тем не менее, по нашему мнению, концепция систем на базе JDF/JMF перспективна, она значительно шире и универсальнее, чем предлагавшиеся до сегодняшнего дня решения от различных производителей. Ее потенциал еще далеко не исчерпан.

Иными словами, понять уровень ожиданий означает сформулировать аргументы в пользу создания проекта по внедрению системы на базе JDF/JMF. Такая система повысит прозрачность затрат, упростит внутрифирменные и внешние коммуникации, снизит уровень брака, улучшит или сделает оптимальнее производственные операции, оптимизирует планирование и снизит издержки? Или все одновременно? Эти вопросы задаются многими потребителями систем. Часто проблема заключается в том, что этими громкими вопросами дело и ограничивается, многие из них не вдаются в детали, где и как конкретно должны быть улучшены коммуникация и так далее. Если ожидания слишком абстрактны, то туманные надежды никогда не оправдаются.

Итак, приступая к разработке и реализации проекта, необходимо четко определить его цели, как с технической, так и с экономической точки зрения. Основой проекта является анализ фактического состояния предприятия. Изучение производственного потока представляет в любом случае самостоятельную ценность, даже если результатом будет рекомендация о нецелесообразности внедрения системы на базе JDF. При анализе возможно определение недостатков с последующей выработкой путей их устранения, даже применяя совсем другие методы и средства. В целом анализ позволяет сформулировать реальные цели для JDF - проекта. Кратко: типография должна выработать техническое задание для планируемых JDF-интерфейсов в системе и обеспечить подтверждение их решения от потенциальных поставщиков оборудования и программного обеспечения.

JDF/JMF - проект необходимо создавать с перспективой (ключевое слово «масштабируемость»). Это потребует увеличения объема работы, однако принесет также и большую пользу. Поэтому все согласны с тем, что проект должен выполняться поэтапно, и для каждой ступени должны быть сформулированы промежуточные цели. Так, приблизительная формулировка этапов может выглядеть следующим образом:

- объединение информационно – управляющей системы (MIS) и подсистемы допечатной подготовки;
- создание необходимого интерфейса между отделом допечатной подготовки и печатным цехом;
- интеграция в общую сеть отдельных машин послепечатной обработки;
- обеспечение бесперебойной связи с заказчиками.

Как правило, информационно – управляющая система является первым кирпичиком создания рабочего потока на базе JDF - сети. Как будет производиться расширение системы управления рабочим потоком, в значительной степени зависит от имеющегося

оснащения. Например, вряд ли предприятие будет приобретать новую фальцевальную машину только по причине того, что она идеально впишется в систему. Экономически это было бы неверно. Вместо этого следовало бы обеспечить ее постоянную загрузку и, возможно, дооснастить программным модулем, для которого доступен JDF - интерфейс, а при необходимой покупке нового оборудования учитывать его JDF - совместимость.

В представленных выше случаях реализации плана требуется активное участие сотрудников, причем степень активности возрастает к концу списка. Прочитируем Джеймса Харви, исполнительного директора организации CIP4 [21]:

«Внедрение автоматизации процессов, даже с помощью JDF, требует активного участия персонала типографии. Разные производители имеют различные концепции автоматизации процессов, и типографии играют роль менеджеров проектов, которым необходимо обойти сложности, вероятные при интеграции нескольких систем».

Участие в общем деле внедрения JDF – рабочего потока не должно инициироваться только лишь сверху, необходимо поддерживать командный дух и интерес к внедрению инноваций. Системы управления рабочим потоком (с JDF и без JDF) изменяют и стирают границы ответственности сотрудников и подразделений. Так, в типографии с JDF – рабочим потоком, как правило, происходит смещение основных компетенций из отдела дпечатной подготовки в отдел приема заказов, и сотрудники часто вынуждены менять рабочее место. Увеличение прозрачности имеет своим следствием также более объективную оценку производительности труда. Сотрудники не поддержат проект, если производственный климат пропитан недоверием и нездоровой конкуренцией. Даже анализ фактического состояния производства на отдельных участках уже может вызвать возражения у работников. Необходимо также при обоюдном согласии руководства и персонала определить право доступа отдельных категорий работающих к производственной информации. Рассмотрим далее вопросы создания систем рабочего потока с использованием модулей одного или нескольких производителей, а также в случае разработки индивидуальных решений.

13.1 Создание рабочего потока с использованием модулей одного производителя.

Конечно, усилия и затраты при создании JDF - рабочего потока из модулей одного производителя ниже, чем при использовании различных систем. Тем не менее, усилия эти все-таки значительны. Всегда имеется необходимость адаптации технологических схем и оборудования предприятия под модули одного разработчика, а это часто затруднительно сделать только лишь силами одного поставщика.

Например, администратор типографии должен иметь возможность включать новые устройства в рабочий поток, устанавливать права пользователей и задавать данные по умолчанию для производства. Поскольку различные модули JDF, как правило, устанавливаются распределенно на различных серверах, то за их работу и функционирование сети должен отвечать ответственный сотрудник (или несколько), чтобы вовремя реагировать на сбои в работе. Одновременно требуется и определенный уровень квалификации сотрудников. На практике часто случается, что этому вопросу уделяется мало внимания. Лишь в том случае, когда сотрудники знают, как необходимо реагировать на сбои, можно ожидать эффекта от рабочего потока на базе JDF. Итак, практика показывает, что для стабильной эксплуатации требуются IT - администратор и администратор рабочего потока. Лучше, если эти функции совмещает в себе один человек. Важность этого сложно недооценить, поэтому необходимо дублирование вторым сотрудником функций администратора. В зависимости от охвата сетью подразделений это может стать действительно сложной задачей. Охват на уровне администратора разветвленной сети – нелегкая задача. Даже у производителей полиграфического оборудования и программного обеспечения чаще встречается узкий “специалист”, чем технически опытный универсал.

При использовании модулей одного производителя для внедрения рабочего потока, не отменяется возможное изменение JDF - сети во времени и необходимости реализации всего проекта поэтапно, как было упомянуто выше. Системы рабочих потоков с использованием формата JDF находятся в состоянии постоянного изменения, адаптации и обновления программного обеспечения. Процесс регулярного обновления и тестирования программного обеспечения должен быть непрерывным. Следует также посоветовать сохранять заменяемое программное обеспечение для простого возвращения к предыдущей версии в случае сбоев и некорректных новых версий.

При создании на предприятии рабочего потока на основе формата JDF необходимо заранее обсудить и проверить конфигурацию сети совместно с разработчиком системы в рабочих условиях типографии. Автоматизация может прекрасно обеспечивать работу для некоторого оборудования и программного обеспечения и совершенно не работать для другого. Причины этого могут быть различны. Например, в системе информационного менеджмента предполагается использование только собственного для типографии каталога схем фальцовки, а создаваемая система использует стандартные спецификации. Определенные тонкости описания продукции могут передаваться не в полном объеме, ошибочно, или же вообще не передаются. Матрицы, которые показывают совместимость

различных систем друг с другом, часто позволяют осуществлять только приблизительную оценку. Поэтому необходимо проверять совместимость детально. Некоторые автоматизированные процедуры в реальном производстве (например, создание макета) можно использовать только для части заказов, а для других придется, как и прежде, выполнять множество ручных операций.

Внедрение рабочего потока на базе JDF требует, как неоднократно было сказано, заранее установленных производственных правил. Они могут запрещать, например, быстрое внесение изменений в находящийся в производстве заказ. Между тем изменения, вносимые в заказ, в современных условиях являются часто необходимостью. Усилия и затраты в этом случае могут быть выше как раз при использовании систем на базе JDF, чем при традиционной обработке с бумажной технологической картой.

Для большого производства и сложного рабочего потока на базе JDF задействуется одновременно много компьютеров в сети и различных программных продуктов. При этом, конечно же, возникают сбои, и определение и устранение ошибок является непростой задачей. Аппаратный сбой в сетевом устройстве может, к примеру, привести к ошибкам в обработке информации и появлению соответствующего брака. Поиск и устранение ошибок может занимать длительное время. В этом случае для сохранения работоспособности производства полезно иметь частично дублирующий («запасной») рабочий поток. Звучит как само собой разумеющийся факт, но таковым не является, поскольку связи между всеми операциями дважды установить слишком затратно. Например, если положения приводочных меток не будут передаваться в файле JDF, то, возможно, не будет работать автоматизированная система приводки в печатной машине и так далее. Включенная в JDF – рабочий поток цифровая печатная машина должна иметь возможность принимать данные и иным путем в случае проблем с передачей JDF-файлов.

В реальном производстве нет необходимости автоматизировать все операции, что предлагается производителями систем. Особенно это касается аспектов экономической эффективности. С технической точки зрения также существуют ограничения. Если старт последующих операций делается зависимым от многих подписей клиента, ответственных за производство или от определенных событий, то это может излишне затруднить производственный процесс. Так, например, подготовка к печати заказа на пульте печатной машины делается зависимой от готовности печатных форм, что означает, что JDF-файлы с параметрами печати загружаются в управляющий компьютер или копируются в “горячую папку” только тогда, когда информационно – управляющая система или система рабочего потока получила сообщение о готовности соответствующих печатных форм. Это

полностью разумное условие создает проблемы, например, когда сообщение от СтР по какой-то причине отсутствует. В таком случае в наличии есть комплект готовых печатных форм и все данные для предварительной настройки машины, но процесс тормозится, поскольку система или печатная машина не получили необходимой информации.

13.2 Создание рабочего потока с использованием модулей нескольких производителей.

Все сказанное в последнем абзаце особенно относится к случаю, когда в одном JDF - рабочем потоке встречаются модули разных производителей. Дополнительно к перечисленным выше случаям добавляется еще несколько аспектов.

В цифровой технике широко известна проблема интерфейсов: если один программный модуль создает файл в формате для другого модуля, то могут возникнуть проблемы отсутствия коммуникации и взаимочтения файлов. Это применимо и к JDF - рабочему потоку. Поскольку JDF и JMF являются относительно новыми форматами (например, по сравнению с PDF), то при их использовании велика вероятность столкнуться с несовместимостью. Рекомендуется при принятии решений провести основательное исследование, прежде всего для нового оборудования и обновлений программных средств. В значительной степени это необходимо интерфейсам обмена данными в формате JDF, особенно в тех случаях, если JDF - файлы передается через “горячую папку”, а не упаковывается в MIME - пакет и пересылается по протоколу HTTP. Возможно перехватывание пакетов в системе. То же может происходить и с файлами JMF. Часто модулями системы предлагается выбор средств коммуникаций – через файлы или HTTP. При проведении исследования интерфейсов необходимо выбрать правильное решение.

Если есть необходимость проверить коммуникации на базе JDF с еще не установленным оборудованием, то следует попросить производителя одного устройства записать JDF - файлы и попросить производителя следующего в технологической цепочке устройства ввести эти файлы для контроля. Иной метод заключается в непосредственном исследовании интерфейса. В обоих случаях при проблемах с совместимостью стараются проанализировать JDF - файлы. Многие производители составляют также т.н. “Белые книги” (White Papers) – документацию, описывающую требования к JDF - файлам для успешного их прочтения и обработки соответствующим программным обеспечением. Если такой документации нет, то можно запросить у создателя программного обеспечения примеры файлов, которые успешно прочитываются.

Проверку JDF - файла нужно начинать с проверки на достоверность (см. раздел 5.2). Для этой цели на сайте CIP4 в разделе «Technical Resources» необходимо выбрать пункт меню «CheckJDFServer» и открыть, таким образом, JDF - файл с помощью сервиса «CheckJDF». Результаты проверки сразу укажут на ошибки, как, например, отсутствие необходимых атрибутов, или наличия невыполнимых ссылок. Можно также использовать службу «FixJDF», исправив с ее помощью JDF - файл и обработав его до соответствия определенной версии. Автоматическое восстановление имеет свои пределы.

Члены организации CIP4 могут осуществлять необходимые действия с помощью ресурса «Technical Resources» и его подраздела «Downloads», «Internal Source» программу «CheckJDF.exe» и дополнительную библиотеку dll. С помощью программы «JDFEditor», бесплатно загружаемой с сайта CIP4, можно также проверить JDF - файлы (рис. 13.1).

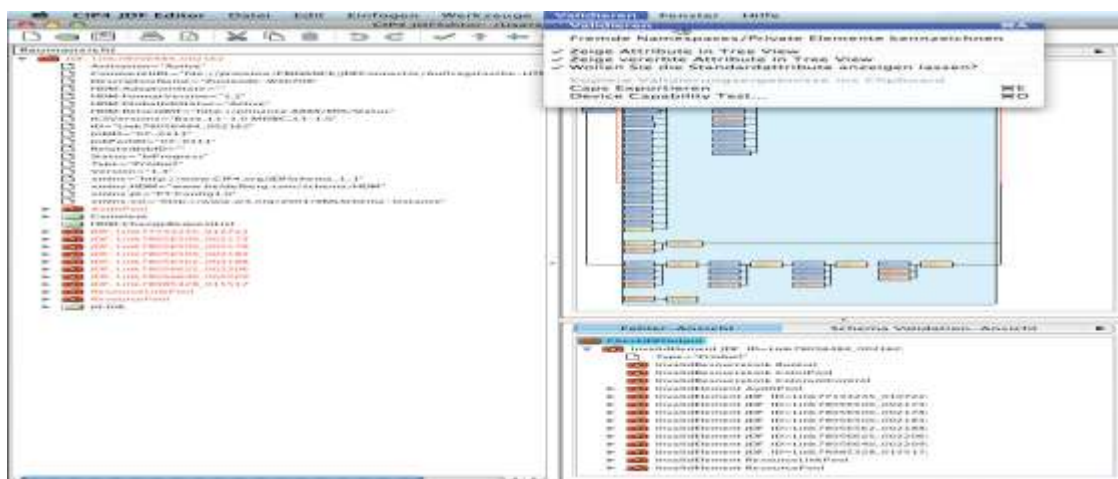


Рис. 13.1 Функция проверки в программе CIP4-JDF-Editor

Полезным может быть просмотр значений атрибутов ICSVersions и анализ ICS - версий JDF - программ. То же можно проделать и для версий JDF, которые обозначены в атрибуте Version. Тем не менее, опыт (или здоровый скептицизм) учит нас, что указанные версии не всегда соответствуют реальности.

Если уже во время проверки и чтения JDF - файла ошибка вызывает аварийное завершение программы, то этому могут быть различные причины. Возможно, какой-то ресурс располагается не непосредственно у JDF - узла, который к нему обращается, а у корневого элемента и не может быть найден (см. рисунок 6.5 в разделе 6.1). Подобные проблемы сложно анализировать без *дебаггинга* кода импортирующего программного обеспечения. В данном случае читателям, интересующимся этими вопросами, как и работникам полиграфических предприятий, можно посоветовать начать тестирование очень простых JDF - файлов, а затем брать все более сложные. Задача значительно

облегчается, если только отдельные значения, которые должны передаваться, не понимаются читающей JDF программой. В этом случае легко посмотреть, стоит ли соответствующий элемент/атрибут в коде JDF, и выяснить, кто из двух разработчиков в этом виноват

JDF - файлы могут иметь значительный объем. Примеры, приведенные в данной книге, являются лишь фрагментарными. В действительности файлы могут занимать не одну страницу и содержать несколько десятков JDF - узлов, что усложняет их анализ.

Авторы хотят еще раз подчеркнуть: несмотря на то, что JDF является промышленным стандартом, интерфейсы, к сожалению, работают небезупречно. Причиной могут являться частные данные конкретных фирм (см. раздел «Стандартизация» в главе 2). Однако в действительности виноваты чаще структуры, отвечающие за применение JDF, и отсутствующая или неверная информация в JDF - файлах. JDF - интерфейсы между модулями различных производителей не работают сами по себе, они должны быть взаимно увязаны. Конечно же, они тестируются производителями перед установкой, но на практике это не приводит к гарантированному исключению ошибок, особенно при сложных заказах. Эти ошибки часто остаются долгое время неясными для пользователей, поскольку они не располагают ни временем, ни инструментами, ни, возможно, знаниями для удовлетворительного анализа проблемы.

13.3 Программирование JDF/JMF.

Данный раздел обращен к тем, кто начинает внедрение на предприятии программного обеспечения и имеет некоторый опыт программирования на JAVA или C++, но еще не сталкивался с использованием JDF. Он не будет интересен профессиональным разработчикам программного обеспечения и экспертам, которые хотели бы увидеть специальные советы и хитрости.

Конечно же, импортирующие или экспортирующие JDF - файлы программы можно написать на любом языке программирования. Например, с помощью Visual Basic for Applications можно легко создать интерфейс, представленный на рис. 13.1, и написать небольшую программу, которая будет сохранять данные, введенные в диалоговом окне, и создавать JDF - файл с информацией о заказчике (*CustomerInfo*) в качестве ресурса (в соответствии с примером на рис.6.8). Однако в данном случае не будет возможности позднейшего обращения к библиотеке программ, создание кода трудоемко и подвержено ошибкам.

На рис. 13.3 представлен фрагмент подобной программы. Данная программа не создает корректного JDF - кода, поскольку, к примеру, оба обязательных атрибута Class и Status отсутствуют. Этим недостатки подобной программы не ограничиваются.



Рис. 13.2 Графический интерфейс пользователя простого JDF - приложения

Написание программы, читающей подобные файлы JDF, еще сложнее, поскольку нет соответствующей библиотеки. В целом, не стоит рекомендовать подобный образ действий при разработке программного обеспечения.

```
Private Sub schreibe_CustomerInfo()
CustomerInfoID = 11101
Print #1, "<ResourcePool>"
Print #1, "<CustomerInfo CustomerID=" & """" & CustomerID & """" & "
CustomerJobName=" & """" & CustomerJobName & """"
Print #1, " CustomerInfoID=" & """" & CustomerInfoID & """" & "
CustomerOrderID = " & """" & CustomerOrderID & """" & ">"
Print #1, "<Comment>" & Comment & "</Comment>"
Print #1, "<Contact Class=" & """" & "Parameter" & """" & "
ContactTypes=" & """" & "Customers" & """"
Print #1, " ID=" & """" & ContactRef1 & """" & "
Status=" & """" & "Available" & """" & ">"
Print #1, "<Person FirstName=" & """" & FirstName & """" & "
FamilyName=" & """" & FamilyName & """" & "
NamePrefix=" & """" & NamePrefix & """" & ">"
Print #1, "<ComChannel ChannelType=" & """" & "Phone" & """" & "
Locator=" & """" & Telefon & """" & "
Status=" & """" & "Available" & """" & "/>"
Print #1, "<ComChannel ChannelType=" & """" & "WWW" & """" & "
Locator=" & """" & Website & """" & "
Status=" & """" & "Available" & """" & "/>"
Print #1, "</Person>"
Print #1, "<Address City=" & """"; City & """" & "
Street=" & """" & Street & """" & " Country=" & """" & Country & """" & "
PostalCode=" & """"; PostalCode & """" & "/>"
Print #1, "</Contact>"
Print #1, "</CustomerInfo>"
Print #1, "</ResourcePool>"
```

Рис. 13.3 Неоптимальная программа создания JDF - файлов

К счастью, организация СІР4 как раз создает подобные библиотеки, но только для JAVA и С++. Поэтому лучше разрабатывать программное обеспечение для JDF - системы на одном из этих языков. Далее мы будем говорить только о JAVA-библиотеке. Сначала необходимо выбрать (интегрированную) среду программирования. Список возможностей велик: приведенный нами пример выполнен с помощью среды *Eclipse* [17], но работают также и в *NetBeans IDE*.

В Eclipse сначала требуется определить новый «проект», при этом библиотеки программ могут связываться в форме jar-файлов. Для разработки JDF - приложений требуются следующие библиотеки, которые можно загрузить из Интернета:

XercesImpl.jar (наша версия: 2.9.0);

Commons-lang.jar (наша версия: 2.3);

JDFLibJ.jar (наша версия: 2.1.3.4, см. www.cip4.org).

В нашем небольшом наглядном примере (рис.13.4) мы создали схему для выполнения следующих функций:

- JDF - файлы считываются и отображаются из выбираемой “горячей папки”;
- по двойному щелчку по обозначению заказа из JDF - файла извлекаются и выводятся на экран некоторые его детали (номер, имя, формат листа, тираж);
- в диалоговом окне отображаются четыре кнопки для отдачи распоряжений: начало работы, макулатурные листы, тиражные листы, конец работ;
- после нажатия кнопки «Начало работы» с помощью фотоэлемента подсчитываются все проходящие через машину, не обладающую JDF - интерфейсом, листы; при выборе «макулатуры» или «тиражных листов» инициируются соответствующие счетчики;
- при нажатии кнопки «Конец работы» программа посылает JMF - сообщение на выбираемый IP - адрес и TCP - порт.

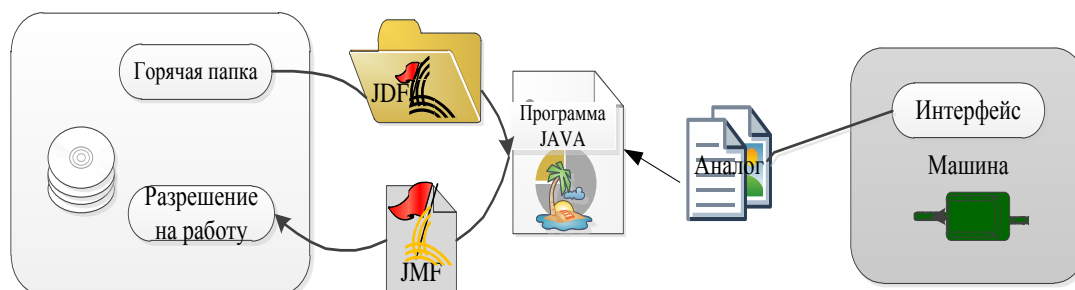


Рис.13.4 Проект по включению машины в сеть JDF/JMF

Конечно, мы приводим здесь не весь ход составления кода программы, а только интересные фрагменты. На рис. 13.5 виден *Thread* (поток команд) с именем *ListWin*, регулярно считывающий имена файлов из “горячей папки” и затем «засыпающий». Он запускается из основного окна программы. Когда он активируется, то сначала удаляет записи из списка выполненных работ, затем заносит имена файлов горячей папки в список и затем отображает результат в основном окне.

```
public Hauptfenster () { // Konstruktor
...
ListWin = new Thread(this);
ListWin.start();
...
}
public void run(){
while (true){
try{
jobliste.removeAll();
dateien = ordner.list();
for (int i =0; i< dateien.length; i++){
jobliste.add(dateien[i]);
}
this.add(jobliste);
repaint();
Thread.sleep(10000);
}
catch (InterruptedException e){
e.printStackTrace();
}
```

Рис. 13.5 Пример записей в «горячей» папке

Записи и величины значений атрибутов представлены на рис.13.6. Сначала представлен JDF – документ *JDFDoc*, и корневой элемент обозначается именем *Root*. Таким образом, легко отделяются атрибуты корневого элемента, как, например, *JobPartID*. *Ressource Pool*, который мы назвали RSP, а затем отдельные ресурсы (например, *CustomerInfo*) могут быть определены с помощью *Root*. Легко представить себе простоту всей системы, несмотря на то, что в целях наглядности мы не стали реализовывать в программе несколько контрольных запросов. Так, например, мы исходили из того, что существует ресурс *CustomerInfo*, который содержит атрибут *CustomerJobName*.

```

LJDFDoc

JDFDocument = JDFDoc.parseFile(pfad + dateiname);

final JDFNode Root = JDFDocument.getJDFRoot();

...

String JobPartID = Root.getJobID(true);

...

JDFResourcePool RSP = Root.getResourcePool();

final JDFResource CustomerInfo = RSP.getResource("CustomerInfo",0,"");

CustomerJobName = CustomerInfo.getAttribute("CustomerJobName");

```

Рис. 13.6 Записи значений атрибутов

В завершение на рис. 13.7 представлен исходный код для отправки сообщения JMF.

```

JMFMessage(String JobID,String JobPartID, Long goodCount){
String DeviceStatus = "Running";
//Construct a JMF-Message, i.e. the content of the http-package
JDFJMF JMF = JDFJMF.createJMF(EnumFamily.Signal,EnumType.Status);
KElement DI = JMF.appendElement("DeviceInfo");
DI.setAttribute("DeviceStatus", DeviceStatus);
KElement JP = DI.appendElement("JobPhase");
JP.setAttribute("TotalAmout", goodCount.toString());
JP.setAttribute("JobID", JobID);
JP.setAttribute("JobPartID", JobPartID);
String content = JMF.toXML();
//construct a http-header
long length = content.length();
String header = "HTTP/1.1 200 OK" + "\nContent-Length: " + length +
"\nContent-Type: text/html";
...
}

```

Рис. 13.7 Исходный код для отправки JMF - сигнала

Для этого JMF - сообщение и HTTP - заголовок соединяются. Создаются и заполняются атрибутами узловые элементы *DeviceInfo* и *JobPhase*. Под конец все описывается в строке символов *Content*. Здесь уже не видно, как с помощью адресов IP и порта создается новый аутентификатор, в который записываются JMF сообщение и HTTP – заголовок. Результатом работы этой маленькой программы является JMF - сигнал, аналогичный на рис. 13.8.


```
HTTP/1.1 200 OK
Content-Length: 509
Content-Type: text/html
<?xml version="1.0" encoding="UTF-8"?>
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1"
TimeStamp="2008-12-02T18:32:24+01:00" Version="1.3">
<!--Generated by the CIP4 Java open source JDF Library version : CIP4 JDF
Writer Java 1.3 BLD 40-->
<Signal ID="m081202_063224609_000000" Type="Status"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="SignalStatus" />
<DeviceInfo DeviceStatus="Running">
<JobPhase JobID="_08-0157" JobPartID="_08-0157" TotalAmout="300" />
</DeviceInfo>
</JMF>
```

Рис. 13.8 JMF - сигнал с HTTP - заголовком

В некоторых случаях нет необходимости писать целое приложение JDF/JMF, а имеет смысл превратить XML - интерфейс программного обеспечения в JDF -интерфейс, или же слегка модифицировать JDF - файлы. Для этого необходимо перевести XML - файл в другую форму XML. Эту задачу можно выполнить с помощью *Extensible Stylesheet Language Transformation (XSLT)*. Для этого необходимо написать XSLT - документ, который будет управлять трансформацией документов. Этой теме посвящено много книг (например, [9]), поэтому мы не будем ее касаться.

Некоторые термины и определения (гlossарий)

Акцидентная печать.

Продукция для бизнеса и частных заказчиков, например, визитные карточки, фирменные бланки, рекламные проспекты и каталоги, заказы на которые поступают нерегулярно. Периодические издания (газеты, журналы), книги и упаковка не относятся к акцидентной продукции.

АМ/ЧМ-растр.

Амплитудно-модулированный (АМ) и частотно-модулированный растр (ЧМ), также называется периодическим и непериодическим растром. Для периодического растра расстояния между центрами растровых точек для одного цветоделенного изображения одинаковы, в то время как для непериодического растра они различны.

ASCII85.

Кодирование поточных двоичных данных при помощи 85 различных символов – набора символов ASCII, которые выражают все печатаемые символы.

Система управления заказом.

Называется также системой информационного менеджмента (MIS) и «отраслевым программным обеспечением» (ПО). ПО для расчета и управления заказами на полиграфическом предприятии.

Спуск полос.

Сведение отдельных полос, заданных с помощью языка описания страниц, и меток макета в единую структуру данных, содержащую всю информацию об одном или нескольких печатных листах или только об отдельных двух сторонах листа (лицо/оборот).

Bitmap, битовая карта.

Структура данных для изображений с глубиной цвета 1 бит (черный/белый).

Макет листа.

См. Макет.

Byte-map, байтовая карта.

Структура данных для изображений с глубиной цвета (минимум) 8 бит (265 возможных оттенков цвета отдельных участков изображения).

Наладка.

Операция подготовки машины к работе.

Управление цветом.

Метод и программные инструменты для наиболее близкого к оригиналу воспроизведения цвета на всех устройствах вывода. (Color Management).

Глубина цвета.

Число битов, которые используются для записи одного пикселя или ступени серого.

Предварительная настройка зональной подачи краски.

Процесс предварительной настройки количества подаваемой печатной краски по зонам в соответствующем красочном аппарате офсетной печатной машины.

Плоттерная распечатка.

Распечатка печатного листа на широкоформатном плоттере для проверки позиций всех элементов печатной формы.

Печать тиража.

Выполнение печати заказа после выполнения приладки и получения подписного листа.

Клапан.

Край запечатываемого материала, который не может быть запечатан, поскольку за него в печатной машине осуществляется удерживание и транспортировка листа захватами печатной машины.

Горячая папка.

Независимый от направления передачи данных интерфейс между двумя программными модулями, который реализован в форме папки. Одна программа сохраняет файлы в эту папку, в то время как другая регулярно проверяет поступление в папку файлов.

Отделка в линию.

Агрегаты по послепечатной отделке и обработке продукции, установленные в печатной машине или соединенные с ней в поточную технологическую линию с единым тактом работы.

Интерпретация.

Трансформация языка описания страниц в растровом процессоре (RIP) во внутренний формат данных, называемый display-list.

Кодирование длин серий или кодирование повторов.

Алгоритм сжатия без потерь, оперирующий последовательностями, в которых один и тот же символ встречается несколько раз подряд.

Направление отлива бумаги.

Направление расположения волокон в структуре бумаги.

Нормализация данных.

Конвертация данных, поступающих от заказчика, в единый PDF-формат.

Монтаж полос.

Сведение отдельных полос и контрольных элементов в оригинал, с которого может быть сделана печатная форма.

Повторяющийся элемент печатной формы.

Расположение нескольких одинаковых составных элементов, например, одинаковых полос или коробок, на одной печатной форме.

«Начало бумаги».

Расстояние от края изображения печатной формы до края бумаги в листовой печати.

Класс бумаги.

Классификация бумажных запечатываемых материалов для офсетной печати по ISO 12647-2.

Префлайт.

Программный анализ файлов на пригодность к обработке в типографии.

Линиатура растра.

Число растровых элементов (точек, линий на единицу длины (дюйм или сантиметр)).

Тип растривания.

Применяемая в конкретных случаях технология растривания. Различают АМ и ЧМ-растривание.

Рендеринг.

Перевод математических описаний объектов, заданных с помощью языка описания страниц (например, PDF), в полутоновые изображения.

Рипование.

Трансформация файлов, описанных языком описания страниц, в пиксельные файлы. Для рекордеров создаются данные с глубиной цвета 1 бит.

Растривание (в автотипных технологиях).

Разложение полутонового изображения (глубина цвета больше 1) на печатающие и пробельные элементы (глубина цвета 1).

Макет полосы.

Определяет положение и размер полос, а также меток.

TIFF-B.

Распространенный формат данных для сохранения битовых карт.

Кривая компенсации.

Кривая калибровки процесса. Кривая или таблица для изменения значений тона в растровом процессоре (RIP).

Треппинг.

Анализ граничащих между собой цветных штриховых объектов и возможное увеличение или уменьшение одного или обоих объектов для устранения последствий неприводки красок цветоделенного изображения при печати.

Trimbox.

Границы обрезного формата печатной продукции.

Изменение размера полей.

Изменение положения на макете внутренних и внешних страниц, например, при комплектовке вкладкой (накидкой) и скреплении проволокой.

Шлейф.

Припуск листов одной половины тетради – передней или задней со стороны бокового поля, необходимый для автоматического раскрытия тетрадей во вкладочно-швейно-резальном агрегате (ВШРА) и при шитье нитками тетрадей в книжном блоке.

Web2Print.

Создание печатной продукции в интернет-браузере, в котором используются стандартные схемы и макеты.

Список сокращений:

термины англоязычные и широко употребляемые в полиграфии, не переводятся, за исключением:

<i>BDE</i>	Betriebsdatenerfassung, система сбора и учета производственных данных.
<i>SOAP</i>	Первоначально Simple Object Access Protocol.

Список литературы:

[1] Adobe System:

Adobe PDF Print Engine 2, White Paper,

<http://www.adobe.com/de/products/>

pdfprintengine/ (2008)

[2] Adobe System Incorporated:

XMP Specification,

<http://www.adobe.com/devnet/xmp/> (2004)

[3] Adobe System Incorporated:

TIFF Revision 6.0,

<http://partners.adobe.com/public/developer/en/>

tiff/TIFF6.pdf (1992)

[4] Adobe System Incorporated:

PostScript Reference Language, 3. Ausgabe,

Addison Wesley Publishing Company,

ISBN 0-201-37922-8 (1999) oder

<http://www.adobe.com/products/postscript/pdfs/>

System Incorporated:

PDF Reference sixth edition, Adobe Portable

Document PLRM.pdf

[5] Adobe Format, Version 1.7,

http://www.adobe.com/devnet/pdf/pdf_reference_

archive.html (2006)

[6] Adobe System Incorporated:

Portable Job Ticket Format, Version 1.1,

Technical Notes # 5620 (1999)

[7] ANSI:

IT8.6-2002 (R2007) Grafic technology – Prepress

digital data exchange – Diecutting data (DDES3)

[8] Apple Inc.:

AppleScript Overview,

<http://developer.apple.com/documentation/>

AppleScript/Conceptual/AppleScriptX/

AppleScriptX.html (2007)

[9] Behme, Henning und Mintert, Stefan:

XML in der Praxis:professionelles Web-Publishing

mit der Extensible Markup Language,

Addison Wesley, ISBN 3-8273-1636-7 (2000)

[10] Bohan, Mark et al:

Automation and JDF Workflow, 2006 TAGA

Proceedings

[11] CGATS.20:

Variable printing data exchange using PPML and

PDF (PPML/VDX),

www.npes.org (2002)

[12] CIP4:

Interoperability Conformance Specifications,

Version 1.3 www.cip4.org/ (2007/2008)

[13] CIP4:

JDF-Specification Release 1.4,

www.cip4.org (2008)

[14] CIP4:

The JDF Marketplace;

<http://www.cip4.org/marketplace/>

[15] Commerce XML (cXML) User's Guide, Version

1.2.019, <http://www.cxml.org> (2008)

[16] Dolin, Penny Ann:

Exploring Digital Workflow,

Thompson Delmar Learning,

ISBN 1-4018-9654-5 (2006)

[17] Eclipse Foundation:

Eclipse IDE for Java Developers,

<http://www.eclipse.org/downloads>

[18] Freund, Jacob; Gützer, Klaus:

Vom Geschäftsprozess zum Workflow,

Carls Hanser Verlag München,

ISBN 978-3-446-41482-2 (2008)

[19] Ghent PDF Workgroup:

PDF/X Plus

www.gwg.org

[20] Hamilton, Eric:

JEPEG File Interchange Format, Version 1.02,

<http://www.jpeg.org/public/jfif.pdf> (1992)

[21] Harvey, James:

About the JDF User Group

<http://www.jdfusergroup.org/>

unter: Grafic Arts Information Network

<http://www.gain.net> [Zugriff April 2009]

[22] Hoffmann-Walbeck, Thomas:

Lehrbuch Digitale Druckformherstellung,

dpunkt Verlag, ISBN 3-89864-182-1 (2004)

[23] International Organization for Standardisation

(ISO) 15930: PDF/X Teil 1 bis Teil 8,

[24] International Organization for Standardisation

(ISO Beuth Verlag (2001 – 2008)

) 12647: Teil 1 bis Teil 7,

Beuth Verlag (2001 – 2007)

Literaturverzeichnis 215

[25] International Organization for Standardisation
(ISO) 16612-2: Grafic technology – Variable data
exchange – Part 2: Using PDF/X-4 and PDF/X-5
(PDF/VT-1 and PDF/VT-2),
under development (2008)

[26] International Press Telecommunication Council
(IPTC): <http://www.iptc.org>

[27] Japan Electronics and Information Technology
Industries Accociation(JEITA):
<http://www.jeita.or.jp/english/>

[28] JEITA:
Exchangeable image file format for digital still
cameras: Exif Version 2.2,
JEITA CP-3451 (2002)

[29] Kipphan, Helmut:
Handbuch der Printmedien,
Springer Verlag Berlin, Heidelberg, New York,
ISBN 3-540-66941-8 (2000)

[30] Kodak Grafic Communications:
“Prinergy 4 Software and Rules-Based Automation”,
White Paper, [http://graphics1.kodak.com/
documents/Release%20date.doc](http://graphics1.kodak.com/documents/Release%20date.doc) (2007)

[31] Koster, Kai:
Informations- und Kommunikationstechnologien
für Unternehmen,
Carl Hanser Verlag, ISBN: 3-446-2118-3 (1999)

[32] Kuhn, Wolfgang; Grell, Martin:

JDF: Prozessintegration, Technologie,
Produktdarstellung,
Springer Verlag Berlin Heidelberg,
ISBN 3-540-20893-3 (2004)

[33] Kular, Christopher:

The Job Definition Format and Print Media,
International Journal Of The Book,
ISBN: 1447-9516 (2007)

[34] Microsoft: Windows Script Host,
<http://www.microsoft.com/downloads>

[35] Mittelhaus, Michael:

Automatisierung in Druckunternehmen,
Bundesverband Druck und Medien e.V (bvdm),
Art-Nr. 83107 (2005)

[36] PODi - the Digital Printing Initiative:
Personalized Print Markup Language – Functional
Specification, Version 2.2 (2006)

[37] PODi - the Digital Printing Initiative:
Personalized Print Markup Language – Imposition
Specification, Version 2.2 (2006)

[38] PODi - the Digital Printing Initiative:
Personalized Print Markup Language – Grafical Art
Conformance Specification Specification,
Version 2.2 (2006)

[39] PODi - the Digital Printing Initiative:
Digital Print Ticket – Printing with PPML and JDF,
Version 2.0 (2005)

[40] PrintTalk Version 1.3, <http://www.cip4.org> (2007)

[41] Schwabe, Gerhard et al (Hrsg.):

CSCW-Kompendium,
Springer Verlag, ISBN 3-540-67552-3 (2001)

[42] St. Laurent, Simon; Fitzgerald, Michael:

XML, O'Reilley Verlag Кцлн,

ISBN: 13 978-3-89721-516-0 (2006)

[43] Workflow Management Coalition, www.wfmc.org

[44] W3C (Dave Beckett):

Resource Description Framework (RDF),

<http://www.w3.org/RDF/>

[45] W3C:

Web Services Description Language (WSDL)

Version 2.0,

Перевод книги осуществлен преподавателями и сотрудниками Национального технического университета Украины, Харьковского национального университета радиоэлектроники, Омского государственного технического университета, Санкт-Петербургского государственного университета технологии и дизайна, Московского государственного университета печати имени Ивана Федорова, Рыбинского полиграфического колледжа.

Координация работы и подготовка книги к изданию на русском языке осуществлена сотрудниками Центра принтмедиаиндустрии Академии медиаиндустрии Федерального агентства по печати и массовым коммуникациям.

Редакция профессора С.И. Стефанова.

Федеральное государственное бюджетное учреждение дополнительного профессионального образования «Академия медиаиндустрии», Москва, 2012г.

Тираж 100экз.

Авторы книги:



Профессор Томас Хоффман-Вальбек
Специалист в области математики,
программных средств, проектного
менеджмента. С 1998 года профессор
Высшей школы Медиа.г.Штуттгарт -
отделение печати и медиатехнологий.
Преподает в области Препресс,
Прикладной информатики, JDF
применения.



Себастьян Ригель-дипломированный
инженер. Окончил Высшую школу
Медиа в г.Штуттгарт. С 2003 года
научный сотрудник отделения
печатные и медиатехнологии.