Job Ticket
Application Programming Interface
(JTAPI)

Version 0.50
August 19, 2003

jt-api-spec--V0050-081903.sxw

Authors (alphabetical)

Authors	Company	Email
Claudia Alimpich	International Business Machines	alimpich@us.ibm.com
Tom Hastings	Xerox	hastings@cp10.es.xerox.com
Till Kamppeter	Mandrake	till.kamppeter@gmx.net
Ira McDonald	High North	imcdonald@sharplabs.com
Glen Petrie (Editor)	EPSON	glen.petrie@eitc.epson.com

Abstract

The Free Standards Group (FSG) Open Printing Job Ticket Application Programming Interface (JTAPI) provides an abstract interface for applications to read, edit, and write industry standard document processing job tickets (defined outside this specification). This specification defines an abstract model of objects (jobs, documents, etc.) and their operations and attributes for document processing (e.g. scanning, printing, copying, etc.). This specification also defines C language and Java bindings of this abstract object model (including standard header files).

Copyright Notice

Table of Contents

1. Introduction

This specification describes and defines the Free Standards Group (FSG) Open Printing Job Ticket Application Programming Interface (JTAPI) for the Free Standards Group.   The JTAPI defines the abstract interface and does define a specific job ticket, job ticket file format or job ticket syntax . An implementation of the JTAPI produces and/or consumes one or more standard or vendor specific job tickets formats.

This specification is intended for software developers, designers and architects that need to read and/or write job tickets.

The purpose of this specification is to define an open standard, the Job Ticket Application Programming, for open printing on Linux/Unix/Posix/Windows/Macintosh/Embedded platforms. A JTAPI implementation produces and consumes job tickets.   The JTAPI is job ticket syntax neutral. The JTAPI isolates an application from the syntax of a job ticket to hind details and the structural complexity of specific job ticket along with interoperability between different job ticket file formats. The JTAPI abstract model is programming language neutral.

2. Background

The International Cooperation for the Integration of Process in Prepress, Press and Postpress (CIP4) is a joint initiative of vendors for the graphical arts industry [cip4].   CIP4 has published a Job Definition Format (JDF) specification.   JDF is a comprehensive XML-based file format proposed industry standard for end-to-end job ticket specifications combined with a message description standard and message interchange protocol to cover all aspects of the commercial printing workflows.

JDF / 1.0 was published April of 2001.   JDF / 1.1 was published in August 2002 [jdf].   JDF / 1.2 is due to be published the end of 2003.   Yearly updates are foreseen as the needs of the printing industry evolve.

The Printer Working Group (PWG) is a joint initiative of printer vendors and print system providers to develop printing protocol standards for use on the Internet and within enterprises on their intra nets [pwg].   The PWG has published the Internet Printing Protocol (IPP) in September 2000 [rfc2910, rfc2911]

The PWG is in the process of publishing the PWG Semantic Model which summarizes the printing semantics common to a number of printing protocols, centered around IPP semantics [pwg-sm]. The PWG Semantic Model includes an XML Schema definition.   Therefore, an XML Job Ticket

using the semantics of the PWG Semantic Model is possible.

3. Terminology and Acronyms <all>

3.1 Conformance Terminology < All >

In this document, the uppercase terms "MUST", "MUST NOT", "REQUIRED", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are intended to be interpreted as described in [RFC2119]

In this document, each conformance statement uses one of the terms:

Term	Meaning
MUST	Implementation support is REQUIRED for conformance to this specification.
SHOULD	Implementation support is RECOMMENDED for conformance to this specification.
MAY	Implementation support is OPTIONAL for conformance to this specification.

In this document, each operation, object, or attribute is defined as:

Term	Meaning
REQUIRED	Each implementation MUST support object operations or attributes.
RECOMMENDED	Each implementation SHOULD support object operations for interoperability
OPTIONAL	Each implementation MAY support object operations or attributes .

[[Editor note: Should we include SHOULD and RECOMMENDED]]

In this document, the term "support" is defined as:

Term	Meaning
"support an operation"	An implementation MUST accept a syntactically correct instance of the operation (includes all REQUIRED parameters) and MUST return one of the defined results for the invoked operation.

Term	*Meaning*
"support an object"	An implementation MUST accept a syntactically correct instance of the object (includes all REQUIRED attributes), MUST locally instantiate the object and MUST return one of the defined results for the invoked operation.
"support an attribute"	An implementation MUST accept a syntactically correct instance of the attribute (includes a valid value) and MUST locally instantiate the attribute. Further, an implementation MUST accept at least one value (other than the empty value NOT_SET) defined for the given attribute. [[Ira will create sentence]]

[[Editor Note: Are all required attribute required to be support for closed set?]]

[[Editor Note: We need to go through all object, attributes to determine which REQUIRED, RECOMMENDED, OPTIONAL]]

Element	Object operation or attribute.

3.2 Other Terminology <all>

Term	*Meaning*
Job Ticket	A set of one or more job processing activities (e.g. stapling, binding, number of copies, insert sheet), contained within a file or stream, serialized into some file format (for example, the XML instance textual encoding of a CIP4 JDF Job Ticket).
Job	A set of one or more processing activities, contained within a "Job Ticket", described by "Job Description" attributes, and processed according to "Job Processing" attributes.
Document	A single document to be processed, contained within a "Job", described by "Document Description" attributes, and processed according to "Document Processing" attributes.

[[Editor Note: Need a better definition for document]]

Document Processing	
Page	A single logical page in a source document. Multiple source pages may be included in a single "Impression" on a "Side" of a "Sheet".
Sheet	A single physical piece of media.

Side	A single side (front or back) of "Sheet" of media.
Impression	A single image marked (by some means) on a "Side" of a "Sheet".

3.3 Acronyms < All >

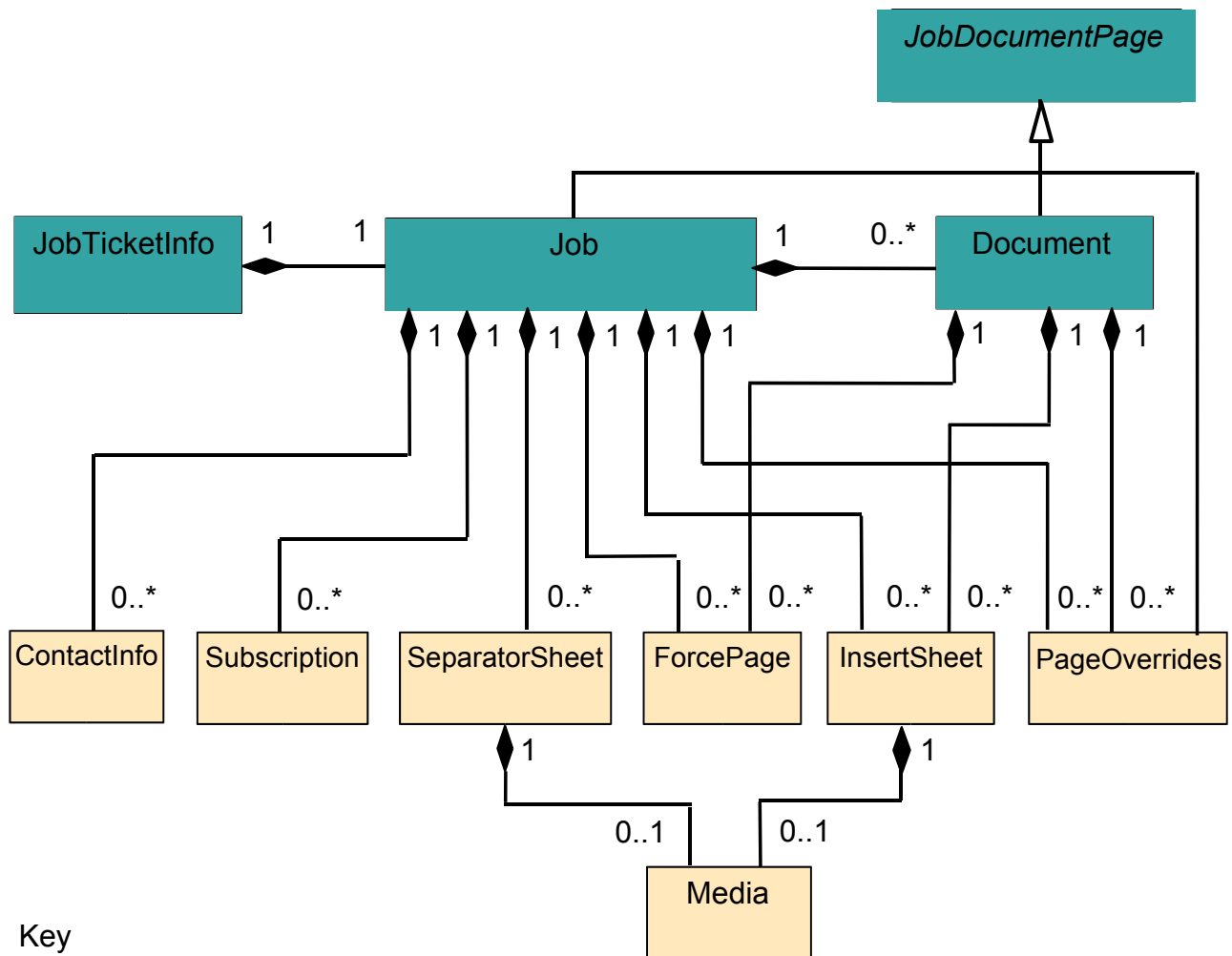| Acronyms	Meaning	Source
CIP4	International **C**ooperation for the **I**ntegration of **P**rocesses in **P**repress, **P**ress and **P**ostpress	http://www.cip4.org/
JDF	**J**ob **D**efinition **F**ormat	Version 1.1, August 2002. See documentation tab at http://www.cip4.org/
PWG	**P**rinter **W**orking **G**roup	http://www.pwg.org/
FSG	Free Standards Group	
FSG/OP	Free Standards Group – Open Printing	http://www.openprinting.org/
JTAPI	**J**ob **T**icket **A**pplication **P**rogramming **I**nterface	

4. High Level Model

The JTAPI is object oriented where the objects represent well-known printing terminology and functionality.

4.1 Diagram

The following two diagrams describe the JTAPI high level model. The diagrams show the objects that are defined in the JTAPI and the relationships between the objects.

A job ticket information object (JobTicketInfo) contains one job object (Job) where the job contains zero or more document objects (Document). The JobDocumentPage object is abstract and contains functionality that is common to jobs, documents, and specific pages (PageOverrides) in a job or document. Each of the other objects in the diagrams represent functionality that can be specified for the Job, Document, and PageOverrides objects . For example, the Media object represents the media that the job, document, or specific pages in the job or document is to be imaged/printed on. The section following the diagrams contain more detailed descriptions of each of the objects in the JTAPI model.

JobDocumentPage

| JobTicketInfo | 1 | 1 | Job | 1 | 0..* | Document |

1 1 1 1 1 1

1 1 1

| 0..* | 0..* | 0..* | 0..* 0..* | 0..* 0..* | 0..* 0..* |

| ContactInfo | Subscription | SeparatorSheet | ForcePage | InsertSheet | PageOverrides |

1   1

0..1   0..1

Media

Key
 italics   abstract class/operation
 1   exactly 1 occurrence
 0..1   0 or 1 occurrence
 0..*   0 or more occurrences
 1..*   1 or more occurrences
 ▨   Alpha implementation
 ▢   1.0 implementation
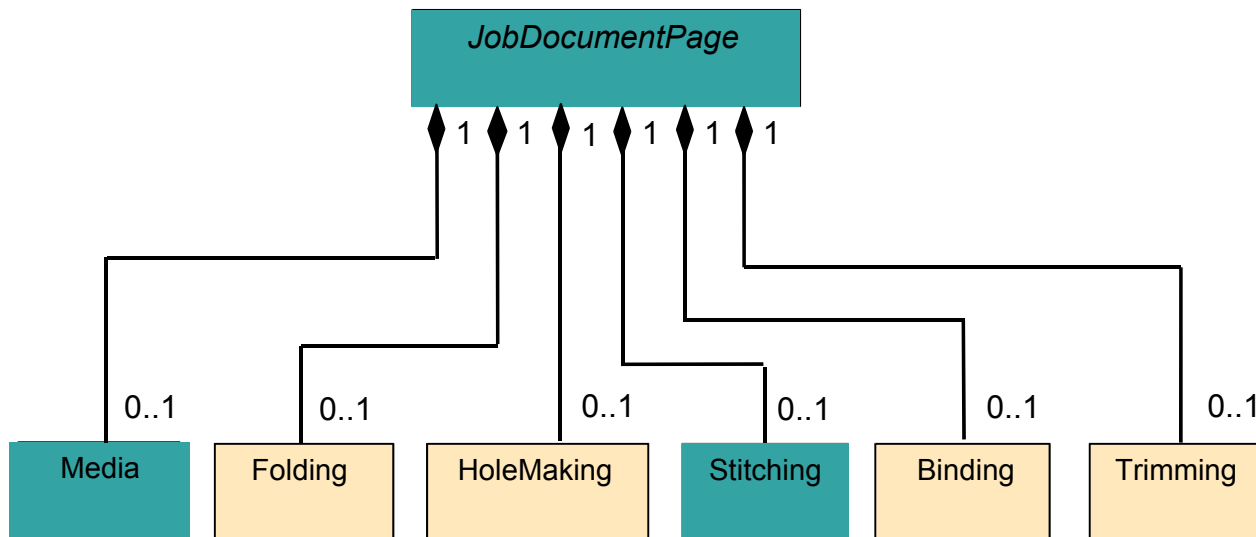
4.2 Object Descriptions

- **Binding**

  A Binding object specifies how a job, document, or range of pages in a job or document are to be bound. It contains attributes that specify the type of binding to perform (like tape or soft cover, which is also known as perfect), and the edge of the sheet to bind.

- **ContactInfo**

  A ContactInfo object contains information about the person who is to be contacted regarding the job. It contains attributes that specify the name, address, company, URIs, and role of the person who is to be contacted. The role of the person identifies why the person is interested in the job. Some typical roles are Administrator, Customer, Delivery, and Approver.

- **Document**

  A Document object represents one or more URIs, where each URI specifies the location of a file that is to be processed. It contains attributes that describe the document and how it is to be processed. A Job object contains zero or more documents.

  Note: For the Alpha implementation, only one URI per Document object is supported.

- **Folding**

  A Folding object contains information that specifies how a job, document, or range of pages in a job or document is to be folded. It contains attributes that specify the type of folding (like z-fold and saddle fold) and the reference edge from which to perform the folding operation.

- **ForcePage**

  A ForcePage object is used to specify that a single page, identified by a number, is to be imaged on a specific side (front or back) of the sheet. Forcing a page is typically performed when the first page of each chapter is to be imaged on the front side of a sheet, also known as "chapter starts". It can also be used to only image on the back side of a back cover. ForcePage is only valid when two-sided printing is specified.

ForcePage will force a page to be imaged on the specified side even if the page would normally fall on the other side. For example, if the second page would normally be imaged on the back side of the sheet and it is to be forced to the front side, then the back side of the current sheet is left blank and the second page is imaged on the next sheet.

A Job or Document object can contain zero or more ForcePage objects.

- **HoleMaking**

  A HoleMaking object specifies how a job, document, or range of pages in a job or document is to be punched. It contains attributes that specify the number of holes to punch and the edge of the sheet to punch.

- **InsertSheet**

  An InsertSheet object encapsulates information that describes an insert sheet that is to be inserted in the job or document. An insert sheet will not be imaged/printed with content that is contained in the data/content file.

  An InsertSheet object contains attributes that specify where the insert sheet is to be placed (before or after specific page numbers), the media that is to be used for the insert sheet, the number of insert sheets to insert, and other insert sheet information. A Job or Document object can contain zero or more InsertSheet objects.

- **Job**

  A Job object represents a job and contains attributes that describes the job and how it is to be processed. A JobTickeInfo object contains one and only one Job object and a Job object contains zero or more documents.

- **JobDocumentPage**

  JobDocumentPage is an abstract object that contains attributes that describe functionality that can be specified for the Job, Document, or PageOverrides objects. For example one media can be specifed for the entire job, which can be overridden by another media for a specific document in the job, which can be overridden by another media for specific pages in the job or document.

- **JobTicketInfo**

  A JobTicketInfo object contains information about the job ticket, such as a comment describing the job ticket, the version of the JTAPI, the type or syntax (JDF, PWG, etc.) of job ticket, and other job ticket information. It does not contain job processsing

information. A JobTicketInfo object contains one and only one Job object.

- **Media**

    A Media object encapsulates information that describes the media that is to be used when printing the job, document, or a range of pages in the job or document. It contains attributes that specify a name that describes the media dimensions (like letter), the color of the media, the actual media dimensions, and other media information.

- **PageOverrides**

    A PageOverrides object contains an attribute that specifies a single page number or a range of consecutive page numbers, and the attributes that are to override the same attributes that are specified for the Job or Document object that contains the PageOverrides object.

    A Job or Document object will contain zero PageOverrides objects if the attributes specified in the Job or Document object are to be used for the entire job or document. A Job or Document object can contain one or more PageOverrides objects where the PageOverrides object's attributes will override the attributes specified for the Job or Document object.

- **SeparatorSheet**

    A SeparatorSheet object encapsulates information that describes separator sheets. A separator sheet will not be imaged/printed with content that is contained in the data/content file. A separator sheet can be placed before the job, after the job, between copies of the job, before each document in the job, after each document in the job, or between copies of the documents in the job.

    A SeparatorSheet object contains attributes that specify where the separator sheets are to be placed, the media that is to be used for the separator sheet, and a message that is to be printed on the separator sheet. A Job or Document object can contain zero or more SeparatorSheet objects.

- **Stitching**

    A Stitching object specifies how a job, document, or range of pages in a job or document are to be stapled/stitched. It contains attributes that specify the number of stitches, the type of stitch (like corner, edge, or saddle), and the edge of the sheet to stitch.

- **Subscription**

A Subscription object is used to subscribe for notification events that are to be sent to a specific URI. It contains attributes that specify the events to be sent, the character set, the language, and the comment text to be included in the event.

- **Trimming**

  A Trimming object specifies how a job, document, or range of pages in a job or document are to be trimmed. It contains an attribute that specifies the type of trimming to perform (like default trimming, face, gutter, or tab).

5. C Language Implementation Framework < who >

5.1 Header Files Relationships < who >

5.2 Header File Naming Convention < who >

5.3 List of Header Files < who >

6. Java Implementation Framework

To be determined.

7. Conformance

[[Note: Should there be statement of the client/applications conformance using the API ?]]

A conforming library implementation of the Job Ticket API v1.0:

(1) MUST support the abstract object model defined in Section 3 of this specification;

(2) MUST publish one or more sets of standard header files verbatim from Section 4 or Section 5 of this specification;

(3) MAY publish one or more sets of vendor extension header files for each supported language binding defined in this specification;

(4) MUST support every REQUIRED object and attribute (with at least one supported value) defined in this specification;

(5) MAY support any OPTIONAL object or attribute (with at least one supported value) defined in this specification;

(6) SHOULD support read/write access to one or more industry standard job ticket formats (defined outside this specification);

(7) MAY support read/write access to one or more vendor-specific job ticket formats (defined outside this specification).

(8) MUST identify supported job ticket formats in any claim of conformance to this specification.

8. Internationalization

Each library implementation of the Job Ticket API MUST accept a charset tag [RFC2978] to specify the character set and encoding for all text strings. Each library implementation of the Job Ticket API MUST default to the UTF-8 [RFC2279] transform of [ISO10646] for text strings.

ISSUE: The Job Ticket API does NOT support the transfer of a language tag [RFC3066] to specify the natural language of text strings, although all XML-based job ticket formats support a language tag. Therefore the Job Ticket API only partially conforms to the IETF Policy on Character Sets and Languages [RFC2277].

9. Security

The Job Ticket API does NOT support the transfer of any user security credentials. Each implementation of the Job Ticket API is a library that may be statically or dynamically linked with an application program. The application program itself may be authenticated and authorized by the native host operating system (by means outside the scope of this specification) for read and/or write access to job tickets stored on local or network file systems. A conforming implementation MUST not store any sensitive information (password, private keys, etc). The JTAPI does not define or support encryption of a job ticket.

10. NORMATIVE REFERENCES

[ISO639]   multi-part International Standard, presently consisting of [ISO639-1] and [ISO639-2].

[ISO639-1] Codes for the Representation of Names of Languages -- Part 1: Alpha-2 Code, ISO/IEC 639-1, 2000.

[ISO639-2] Codes for the Representation of Names of Languages -- Part 2: Alpha-3 Code, ISO/IEC 639-2, 1998.

[ISO3166] multi-part International Standard, presently consisting of [ISO3166-1] and [ISO3166-2].

[ISO3166-1] Codes for the Representation of Names of Countries and their Subdivisions, Part 1: Country Codes, ISO/IEC 3166-1, 1997.

[ISO3166-2] Codes for the Representation of Names of Countries and their Subdivisions, Part 2: Country Subdivision Codes, ISO/IEC 3166-2, 1998.

[ISO10646] multi-part International Standard, presently consisting of [ISO10646-1] and [ISO10646-2].

[ISO10646-1] Information Technology - Universal Multiple-Octet Code Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane, ISO/IEC 10646-1, September 2000.

[ISO10646-2] Information Technology - Universal Multiple-Octet Code Character Set (UCS) - Part 2: Supplemental Planes, ISO/IEC 10646-2, January 2001.

[RFC2119] Bradner. Key words for use in RFCs to Indicate Requirement Levels, RFC 2119, March 1997.

[RFC2396] Berners-Lee, Fielding, Masinter. URI Generic Syntax, RFC 2396, August 1998.

[RFC2910] Herriot, R., Butler, S., Moore, P., Turner, R., "Internet Printing Protocol/1.1: Encoding and Transport", RFC 2910, September 2000.

[RFC2911] R. deBry, T. Hastings, R. Herriot, S. Isaacson, P. Powell, " Internet Printing Protocol/1.10: Model and Semantics", RFC 2911, September 2000.

[RFC2978] Freed, Postel. IANA Charset Registration Procedures, RFC 2978, October 2000.

[RFC3066] Alvestrand. Tags for the Identification of Languages, RFC 3066, January 2001.

11. INFORMATIVE REFERENCES

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

[jdf]   The Job Definition Format (JDF), version 1.1, August 2002. Sett the Document tab at: http://www.cip4.org

[pwg]   The Printer Working Group located at http://www.pwg.org/

[pwg-sm]   Zehler, P., Hastings, T., and Albright, S., Printer Working Group (PWG): Semantic Model, March 26, 2003, work in progress at ftp://ftp.pwg.org/pub/Semantic-Model/wd-sm010-20030326.pdf

[IANA-CHAR]   IANA Registry of Character Sets ftp://ftp.iana.org/assignments/charset-reg/...

[IANA-MIME]   IANA Registry of MIME Media Types ftp://ftp.iana.org/assignments/media-types/...

[RFC2277] Alvestrand. IETF Policy on Character Sets and Languages, RFC 2277, January 1998.

[RFC2279] Yergeau. UTF-8, a Transformation Format of ISO 10646, RFC 2279, January 1998.

APPENDIX: X CHANGES < Editor >

Date	Affected Version	Author	Change
08.12.03	0.30	G. Petrie	Added comments from T. Hastings in the introduction and reference sections.
08.12.03	0.30	G. Petrie	Added comments from I. McDonald in the reference section. Added comments and changes based on discussion from 08.12.03 weekly job-ticket meeting. Many section affected.

End of Document