INTERNATIONAL STANDARD

ISO/IEC 8831

Second edition
1992-03-01

Information technology — Open Systems Interconnection — Job transfer and manipulation concepts and services

Technologies de l'information — Interconnexion de systèmes ouverts — Concepts et services pour le transfert et manipulation de travaux

Reference number
ISO/IEC 8831:1992(E)

Contents

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 8831 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

This second edition cancels and replaces the first edition (ISO 8831:1989). The text of this second edition of ISO/IEC 8831 includes changes resulting from alignment with ISO/IEC 9804 (CCR) and the full protocol amendment to ISO/IEC 8832.

Annexes A and B form an integral part of this International Standard. Annexes C and D are for information only.

Introduction

The purpose of the job transfer and manipulation (JTM) standard is to provide a set of communication-related services which can be used to perform work in a network of interconnected open systems. This work can include both the running of traditional background jobs and other forms of information processing.

Background jobs have, in the past, been submitted either directly on the host system where they ran, or else at a remote job entry station connected to that system. Data files, program and "JCL" would be already available on the host, or would form part of the submitted "job deck". Output would be delivered on the host system, or alternatively on a printer attached to the remote job entry station. In a network of open systems, such jobs can be submitted at any open system supporting JTM to be run on another open system, using files collected from any other open systems, with output directed to peripherals or files held on any other open systems.

The JTM protocol covers not only the movement of job-related data (input and output) between open systems, but also provides for the movement of data concerned with monitoring job-related activity, and for controlling and manipulating the progress of this activity.

This International Standard does not specify individual implementations or products, nor does it constrain the implementation of entities and interfaces within a computer system. There is, therefore, no conformance to this International Standard.

Annex A is part of this International Standard, and describes the notation used for JTM service definition. This differs from that used by lower layers of OSI only because JTM is concerned with concurrent and related activity on more than two open systems. It is a superset of the notation used in the lower layers.

Annex B is part of this International Standard, and defines the requirements of JTM for a Document Type Registration Authority. These requirements of JTM apply to private, enterprise-specific, Registration Authorities; they are, however, also expected to be satisfied by any International Standard covering this area.

Annex C is tutorial material, and is not part of this International Standard. It gives a broad introduction to JTM, and should be read first by readers unfamiliar with the JTM work.

Annex D is not part of this International Standard, and contains an alphabetic glossary of the definitions appearing in the body of this International Standard.

Information technology – Open Systems Interconnection – Job transfer and manipulation concepts and services

Section 1 : General

1.1 Scope

This International Standard is an application layer standard within the Open Systems Interconnection framework set up by ISO 7498.

It defines the concepts and services for job transfer and manipulation.

This International Standard requires that the user of JTM

- specifies the open systems where work is to be done;
- knows the local functions and facilities of the open systems where work is to be done;
- knows the control languages used to specify local work on the open systems where work is to be done.

This International Standard provides the means to

- to specify work to be done on one or more open systems. The work done at one open system can result in new work to be done at other open systems;
- monitor the execution of work previously specified;
- modify work previously specified.

This International Standard does not address the standardisation of control languages, but is also applicable to the use of a standardised control language. This International Standard does not address the standardisation of user interfaces.

1.2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO 646:1991, *Information technology – ISO 7-bit coded character set for information interchange.*

ISO 2022:1986, *Information processing – ISO 7-bit and 8-bit coded character sets - Code extension techniques.*

ISO 2375:1985, *Data processing – Procedure for registration of escape sequences.*

ISO 7498:1984, *Information processing systems – Open Systems Interconnection – Basic Reference Model.*

ISO/TR 8509:1987, *Information processing systems – Open Systems Interconnection – Service Conventions.*

ISO 8571-3:1988, *Information processing systems – Open Systems Interconnection – File Transfer, Access and Management – Part 3 : File Service definition.*

ISO 8649:1988, *Information processing systems – Open Systems Interconnection – Service definition for the Association Control Service Element.*

ISO 8822:1988, *Information processing systems - Open Systems Interconnection - Connection oriented Presentation Service Definition.*

ISO/IEC 8824:1990, *Information technology – Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1).*

ISO/IEC 8832:1992, *Information technology – Open Systems Interconnection – Specification of the Basic Class and Full Protocol for Job Transfer and Manipulation.*

ISO/IEC 9804:1990, *Information technology – Open Systems Interconnection – Service definition for the Commitment , Concurrency and Recovery service element.*

1.3 Definitions

1.3.1 CCR service definitions

This International Standard makes use of the following terms defined in ISO/IEC 9804:

a) atomic action

b) master

c) superior

d) subordinate

e) commitment

f) rollback

1.3.2 JTM service definitions

The definitions are grouped into major categories, corresponding to the subclauses of clause 2.1.

For the purposes of this International Standard, the following definitions apply.

1.3.2.1 General

1.3.2.1.1 agency: An abstract description of those functions of a real open system which are needed to support the JTM service.

1.3.2.1.2 work specification: A conceptual data structure within the JTM service provider which specifies in a defined way the work which is to be done.

1.3.2.1.3 document: A collection of data which forms part of a work specification, and which forms a unit of interaction between the JTM service provider and an agency.

1.3.2.1.4 initiation agency: That agency which causes a work specification to be created.

1.3.2.2 Proformas and spawning

1.3.2.2.1 proforma: Part of a work specification which specifies further work and is used to form a new work specification as part of the processing of earlier work.

1.3.2.2.2 spawning: The process of taking the data from a proforma and using it to produce a new work specification.

1.3.2.2.3 spawning control data: Data contained in a proforma which controls the circumstances in which spawning takes place from that proforma.

1.3.2.2.4 top level proforma: A proforma which is not contained within any other proforma.

NOTE — A proforma which is not a top-level proforma can become a top-level proforma as a result of spawning from its parent.

1.3.2.3 Source, sink and execution agencies

1.3.2.3.1 source agency: Any part of an open system which can provide documents for inclusion in a work specification when required by the JTM service provider as a result of processing the work specification.

1.3.2.3.2 sink agency: Any part of an open system to which documents can be passed by the JTM service provider as a result of processing a work specification.

NOTE — Source and sink agencies can obtain and dispose of documents locally, or by use of non-standard protocols, or by use of FTAM.

1.3.2.3.3 execution agency: Any part of an open system which initially acts as a sink for documents, but which subsequently acts as a source of related documents produced as a result of processing the earlier documents.

1.3.2.3.4 activity (in an agency): Work performed by an agency, initiated by a service primitive issued to the agency by the JTM service provider; the completion of the activity is indicated by a service primitive issued to the JTM service provider by the agency.

1.3.2.4 OSI jobs

1.3.2.4.1 initial work specification: A work specification created as a result of the issue of an initiation service primitive by an initiation agency.

1.3.2.4.2 OSI job: The total work on all open systems arising directly or indirectly from an initial work specification.

1.3.2.4.3 OSI subjob (subjob): The total work arising from the processing of a single work specification, including the spawning of further work specifications, but excluding work arising from the processing of these further work specifications.

1.3.2.4.4 OSI job submission: The use of the initiation service primitive by an initiation agency for the creation of an initial work specification.

1.3.2.4.5 OSI job submission system: The open system on which OSI job submission occurs.

1.3.2.5 Reporting and the monitor function

1.3.2.5.1 JTM report: Encoded information recording the progress or failure of an OSI job, generated by the JTM service provider, possibly as the result of interaction with an agency.

1.3.2.5.2 OSI job monitors: Open systems to which JTM reports about a particular OSI job are sent.

1.3.2.5.3 report work specification: The type of work specification created by the JTM service provider to move JTM reports; the target open system for these work specifications is one of the OSI job monitors.

1.3.2.6 Commitment, concurrency and recovery

1.3.2.6.1 level of commitment: A parameter which determines whether operations requested in an atomic action are completed at the time of the atomic action, or are noted (as secure data) for later performance.

1.3.2.6.2 warning diagnostic: Information carried by the CCR service on an offer of commitment which reports (usually for a human being) any variations on the expected action or unexpected consequences of the action.

1.3.2.6.3 retry-later diagnostic: Information carried by the CCR service on a rollback when an action cannot be completed for reasons which can be transient.

1.3.2.6.4 no-retry diagnostic: Information carried by the CCR service on a rollback when an action cannot be completed, and a later retry is not proposed.

1.3.2.7 Transfer control

1.3.2.7.1 work specification transfer: An atomic action by which a work specification is created at the receiving open system and destroyed at the sending open system.

1.3.2.7.2 transfer control record: A conceptual data structure held by an open system to control the transfer of work specifications and the issue of service primitives.

1.3.2.8 Report manipulation

1.3.2.8.1 report manipulation operations: Operations requiring deletion or display of reports held by an open system nominated as a monitor point by some work specification.

1.3.2.8.2 report manipulation work specification: A work specification containing report manipulation operations.

1.3.2.9 Work manipulation

1.3.2.9.1 work manipulation operations: Operations which select one or more work specifications or proformas and request displaying, killing, stopping or modification.

1.3.2.9.2 work manipulation work specification: A work specification containing work manipulation operations.

1.3.2.9.3 selector: Data which is used to select zero, one, or more work specifications.

1.3.2.9.4 update: Data which is used to modify a selected work specification or proforma.

1.3.2.10 Transfer manipulation

1.3.2.10.1 transfer manipulation operations: Operations requiring setting, displaying or checking transfer control records.

1.3.2.10.2 transfer manipulation work specification: A manipulation work specification containing transfer manipulation operations.

1.3.2.11 Authorisation and accounting

1.3.2.11.1 identification authority: A naming authority which issues identifications; these identifications can be used to determine the capabilities to be made available to a particular authenticated identification (authorisation), or can be used to levy charges (accounting), or both.

1.3.2.11.2 authenticated identification: Data which is known to correctly identify the user or management who requested the work to be performed, either by the use of a password check, or by some other checking mechanism.

1.3.2.11.3 user identification: Data which can be used in a particular context to identify the user on whose behalf work is being requested.

1.3.2.11.4 account identification: Data which can be used in a particular context to identify the account to be debited with any charges which are levied.

NOTE — User and account identifications consist of the name of an identification authority together with one of the identifications it issues.

1.3.2.11.5 open system management identification: The name of an open system which, when authenticated, authorises JTM activities or charging related to the management of that open system.

NOTE — Examples of such activities are the holding of work specifications about to be transferred to it, or the setting of transfer control records.

1.3.2.11.6 identification authority management identification: The name of an identification authority which, when authenticated, authorises JTM activities related to control of activity initiated by user identifications issued by that authority.

1.3.2.12 Work specification identification

1.3.2.12.1 OSI job local reference: A reference for an OSI job which is unambiguous within the OSI job submission system, assigned by that open system.

1.3.2.12.2 initiating identification: An identification provided by the JTM user at submission time to identify the initiator of the OSI job.

1.3.2.12.3 OSI job name: A string provided by an initiation agency when submitting an OSI job.

1.3.2.12.4 work specification identifier: A unique reference for a work specification which includes the name of the OSI job submission system, the identification of the initiating user, the OSI job local reference and the OSI job name; where a work specification was created by spawning, the identifier also contains one or more proforma names.

1.4 Abbreviations

OSI	open systems interconnection
JTM	job transfer and manipulation
FTAM	*file transfer, access and management*
CCR	commitment, concurrency and recovery

1.5 Conventions

The conventions used in this service definition are contained in annex A.

Section 2 : Overview

2.1 Overview and general description

In order to define JTM services, a model of the elements involved in the services is required. This JTM model is derived from the definitions given in ISO 7498 by adding further detail to encompass the services.

2.1.1 Overview

The JTM model recognizes the existence of a number of independent application entities on different open systems which co-operate to provide the JTM service and together form the JTM service provider. The JTM model also recognizes the existence of a number of agencies which are the users of the JTM service. The conceptual interactions between the JTM service provider and an agency are defined by service primitives. The JTM service provider receives from an initiation agency enough information to create a work specification.

The work which is requested is performed by

a) standardized functions of the JTM service provider;

b) functions of the local system environment accessed by the movement of documents between the JTM service provider and agencies.

2.1.2 Work specification contents

A work specification contains fields which provide data for

a) identification of the work;

b) authorisation of the work;

c) defining where reports on the work are to be sent;

d) selecting the reports which are required;

e) identifying the type of the work specification;

f) identifying the open systems which are to perform the work;

g) specifying the urgency of the work;

h) holding parts of the work until specified events occur;

i) specifying the actions to be performed by the JTM service provider to carry out the initial work;

j) specifying further work to be carried out when the initial work is completed.

A work specification refers to documents in the local system, or to documents which are obtained by the use of ISO 8571-3; these documents are subsequently passed by the JTM service provider to the same or to some other open system for storage locally, or for disposal using ISO 8571-3. When an open system has completed the work specified by a work specification (possibly including the generation of one or more new work specifications), the work specification ceases to exist. The JTM protocol transfers work specifications between open systems in order to progress the work.

The contents of a work specification relate to the following features supported by the JTM service, and are completely defined in clause 3.4.

The **identification** field of a work specification provides a universal and unique name by which the work can be referenced for subsequent reporting and manipulation. The work specification identifier is allocated when the work specification is created following submission by the initiation agency, or when it is created as the result of processing an earlier work specification. In the latter case, the identifier will reflect the parentage of the work specification.

The **initiating identification** field, and the **time of submission** field are generated when a work specification is created as a result of submission by the initiation agency. These fields are copied when new work specifications are created by JTM as a result of processing earlier work specifications.

The **authorisation** and **accounting** fields provide data which enables open systems to allow the performance of the requested work. When a work specification references files for obtaining or disposing of documents, additional authorisations and accounts can be included for the access to the files.

OSI job monitors can be specified; these are open systems to which any selected reports are to be sent.

The **report selector** field determines (for each OSI job monitor) which categories of event are to be reported to that monitor.

Different **types** of work specification are defined, corresponding to the different types of initial work to be performed. The most important type is the *document movement work specification which provides for the movement of user documents* between open systems. Other types provide for the movement of reports and manipulations: they are introduced later.

The **target** open system is that open system which carries out the final processing of the initial work, and which can generate new work specifications for further work. **Relays** can be specified for use as store-and-forward sites on the way to the target.

The **urgency** for the performance of the desired work and the **holding** of it can be specified.

JTM **actions** specify – in relation to the various types of work specification – what specific actions the JTM service provider has to perform. They determine the interactions between the JTM service provider and its local system environment, and determine the actions to be taken by it on its own data, or with regard to the reporting of events.

The **further work** field provides data in a form which allows the creation of new work specifications by a target.

2.1.3 Proformas and spawning

A proforma can contain further proformas nested to any depth. The JTM service provider spawns work specifications using top level proformas and the spawning control data in the proformas. The new work specifications are formed using both data in the proformas, and also data in other fields of the original work specification. An important feature of spawning is the addition to the new work specifications of documents which have become available as a result of earlier activity. The spawning process is initiated by an execution agency (see 2.1.4), or as the result of completion of activity on a sink or execution agency (see 1.3.2.3.4).

2.1.4 Source, sink and execution agencies

The model recognizes that the JTM service provider, when processing a work specification, can – according to information in the work specification – *interact with source, sink and execution agencies.*

There are service primitives for interactions between a JTM service provider and agencies. Following these interactions with a source, the activity is complete. Following these interactions with a sink or execution agency the activity can either be complete or the agency can indicate only acceptance, that is, that the request for some activity has been secured. In the latter case, completion of the activity will be signalled by the agency at some later time, using another set of service primitive interactions.

An execution agency can, on completion of an activity, indicate that a number of documents are available for collection by a work specification resulting from spawning. It can also, at any time prior to completion, and if required by the activity, use service primitives to demand spawning using specified proformas; the resulting work specifications can collect documents which have become available before the end of the activity.

At any time prior to the completion of an activity in a sink or execution agency, the following events can occur:

a) the JTM service provider can, as a result of processing a request for work manipulation (see 2.1.11), require that the activity be killed or stopped or held or released;

b) *the JTM service provider can ask for status information about the activity;*

c) the agency, if required by the activity, can cause the JTM service provider to generate a report about some significant event in the life of the activity;

d) the agency, if required by the activity, can ask the JTM service provider to spawn from specified proformas.

All service primitives defined for interaction with an agency relate to a single work specification, and are independent of primitives used in relation to other work specifications.

An agency which cannot support concurrent activity on behalf of several work specifications can reject an attempt by the JTM service provider to initiate new activity with the response "retry later".

Primitives used to obtain or dispose of documents using source and sink agencies carry information relating to the activity being attempted. The information is in one of two forms:

a) in a JTM-specific form suited to simple local access, or to remote access in a non-standardised manner; or

b) in a form specified by ISO 8571-3 (FTAM), suitable for local or remote access where FTAM is supported.

2.1.5 OSI jobs

A complete OSI job can involve access to source agencies on the OSI job submission system, access to source agencies, sink agencies and execution agencies on other open systems, the spawning of further work specifications and their processing. The complete OSI job comprises a set of activities performed either consecutively or simultaneously, potentially involving many open systems.

2.1.6 Processing of work specifications

2.1.6.1 Normal processing

After OSI job submission, an initial work specification can cause accesses by the JTM service provider to source agencies at the OSI job submission system, together with accesses to agencies at other open systems. The processing of a work specification typically comprises the following steps, performed repeatedly if necessary:

a) source agencies are accessed and the documents delivered by them are (conceptually) embedded in the work specification;

b) the work specification (including any embedded documents) is transferred to another open system, using the JTM protocol;

c) documents within the work specification are passed to one or more sink or execution agencies as specified by the work specification;

d) new work specifications are spawned using top level proformas contained within the original work specification, either on acceptance of documents by agencies, or on demand from an activity in an agency, or following completion of the sink or execution agency activity; documents which have been made available as a result of activity in an execution agency or which are specified to be obtained from other source agencies can be included in the new work specifications;

e) the original work specification is destroyed when all the activity resulting from the above processing is completed.

If new work specifications have been produced by the processing of a work specification then these can cause further accesses to source agencies, followed by processing, as described above.

An illustration of the processing of a work specification, the activities of the JTM service provider related to an OSI job, including reporting (see 2.1.7) and manipulation (see 2.1.11), and the relationship between JTM model elements is given in annex C.

2.1.6.2 Errors in OSI job processing

Where errors are detected in a work specification (being received by an open system) before it has taken responsibility for it (offered commitment – see 2.1.8), such errors are reported to the sending open system using facilities present in the JTM protocol. This is seen by the sending system as a failure to transfer the work specification.

Errors which are detected by an open system (with responsibility for a work specification) while processing it are treated as described below.

The normal processing of a work specification, as described in 2.1.6.1, involves obtaining documents from source agencies and disposing of documents to execution agencies and sink agencies. Errors can arise which prevent one or more of these document exchanges being completed successfully. For example, the work specification might contain the name of a document which does not exist in the filestore accessed by the specified source agency, or might contain the name of a sink or execution agency which does not exist at the specified open system. Such errors are events for which JTM reports can be generated (see 2.1.7).

An error which prevents a document being obtained does not necessarily prevent further progress of the OSI job, as further useful processing can still be possible on other documents which have been obtained successfully. The work specification can indicate that an error diagnostic is to be included in the work specification in place of the document which could not be obtained; this error diagnostic is passed to the intended sink or execution agency instead of the document. In this case, all specified interactions with agencies still take place, and the sink or execution agency decides what action to take in the event of an error diagnostic being present.

An error detected while disposing of a document to a sink or execution agency, or while attempting a JTM transfer, prevents further processing of the OSI job. In this situation, the JTM service provider can be required to retain the work specification for a certain period of time to allow the error to be corrected by a subsequent work specification manipulation (see 2.1.11). (The work specification can also specify this form of error handling when documents are being obtained from a source.) For example, an erroneous sink or source name can be changed to a correct sink or source name or an erroneous target name can be changed to a correct target name. If the error has not been corrected when the time period expires, then the JTM service provider deletes the work specification, generating an abnormal termination report (see 2.1.7). This ultimate action can be specified in the work specification as the immediate action on errors, and is the only error handling provided in the Basic Class subset.

Error handling by the open system responsible for a work specification can be summarised as follows:

a) accessing a source agency:

   1) embed a diagnostic and continue;

   2) abnormal termination;

   3) suspend to allow user correction and a later attempt;

b) accessing a sink agency;

   1) abnormal termination;

   2) suspend to allow user correction and a later attempt;

c) attempting to transfer a work specification:

   1) abnormal termination;

   2) suspend to allow user correction and a later attempt;

In the Basic Class, only option (i) is available in each of the above cases.

2.1.7 Reporting and the monitor function

After the JTM user submits an OSI job (through an initiation agency), he can be decoupled from it. Since, in general, it can take a long time for the complete work to be performed, the JTM user has to be able to find out about the progress of the OSI job and any related work specifications. Since there might be no direct contact with the JTM user, he cannot be directly informed of the progress or failure of the OSI job. The concept of OSI job monitors is therefore introduced.

These are open systems to which report work specifications can be sent regarding the progress of the OSI job: they are specified at OSI job submission time. When a significant event occurs in the life of an OSI job, the JTM service provider can create a report and deliver it to one or more of the OSI job monitors. The originating user can either enquire from an OSI job monitor in order to discover the status of the OSI job, or can arrange for all reports to be delivered by the OSI job monitor to sink agencies of his choice. The presentation of a report in a suitable (human-readable) format is the responsibility of the sink agency, and is not standardized.

The data needed to create a report work specification, and to select events for reporting, is included in fields of the work specification being reported on.

Two forms of OSI job monitor are supported. The **primary monitor** (and the report categories to be sent to it) is determined by the JTM service provider at the OSI job submission system. This data can only be changed by the "management" of that open system. The **secondary monitors** (and the report categories to be sent to each one) are determined by the user through parameters on the initiation service primitive. The user can change this data as the work proceeds (see 2.1.11).

The responsibility for generating reports is described in 2.1.15, but the OSI job submission system is responsible for ensuring that the monitor specification and the authorisation data allow any requested reports to be delivered to the primary monitor. (The user is responsible for such actions in relation to secondary monitors; the service provider cannot guarantee delivery of reports to secondary monitors if this data is in error or becomes out-of-date.)

Report work specifications differ from other work specifications in the following respects:

a) report work specifications do not contain any user documents; the report information is contained in them in encoded form;

b) a report work specification is not created by spawning from a proforma; it is created from a monitor specification contained in the initial work specification and copied into all work specifications produced by spawning;

c) the work specification identification of a report work specification is the same as that of the work specification being reported on;

d) the processing of report work specifications never causes further reports to be generated; errors detected during their processing or transfer cause them to be discarded;

e) a report work specification contains all the permissions (see 2.1.13) which were present in the work specification being reported on; these are, however, ignored if any attempt is made to modify the report work specification; they are present to provide information to the OSI job monitor for handling subsequent requests for report manipulation (see 2.1.10);

f) transfer of report work specifications (and manipulations – see 2.1.11) has priority over other JTM traffic.

The following categories of event can be selected for reporting (these categories are defined in clause 3.3):

a) the creation of an OSI job;

b) the transfer of (responsibility for) a work specification from one open system to another open system;

c) the creation of a new work specification by spawning;

d) the acceptance of documents by sink and execution agencies;

e) the normal termination of a work specification following completion of the subjob;

f) the premature ending of a subjob due to a manipulation request (STOP);

g) the deletion of a work specification and termination of the subjob due to a manipulation request (KILL);

h) the modification of a subjob by a manipulation request;

i) the detection of errors in a work specification which result in error diagnostics being embedded in it;

j) the holding of a work specification for the correction of errors;

k) abnormal termination of the work specification due to errors;

l) the generation of accounting information related to the subjob from the local systems environment;

m) the generation of information by activity in an execution agency (user messages);

n) abnormal termination of the work specification due to a request for facilities which are not supported;

o) an attempted modification of the work specification by a subjob lacking the necessary authorisation;

p) warnings associated with successful completion of the JTM activity, but alerting the (human) user to possible effects or variations.

A report work specification carries one or more reports. Each report has the following parameters:

a) the name of the open system generating the report (see 2.1.14);

b) the identification of the subjob being reported on, and of its initiator;

c) the type of event being reported;

d) the date and time when the report was generated (optional);

e) a non-standardised text message.

Reports for certain categories of event also carry the name of the target, the number of work specifications which were spawned, a diagnostic message, the current hold/released status, and identification of another subjob (violation attempts). The reports which carry these parameters are defined in clause 3.3.

2.1.8 Commitment, concurrency and recovery

2.1.8.1 Overview

In order to ensure the correct operation of the JTM service in the presence of application or communication failures (recovery), freedom from interference by other activities (concurrency control), and the ability to perform the OSI job as one or more atomic actions (commitment), the Commitment, Concurrency and Recovery (CCR) procedures specified in ISO/IEC 9804 are used.

Each JTM transfer between open systems, and each interaction with an agency, uses service primitives to which the CCR semantics are applied. The procedures specified in ISO/IEC 9804 are carried out by each open system, and are conceptually carried out by each agency.

Annex C of ISO/IEC 9804 describes the nature of CCR, for those unfamiliar with this work. An understanding of CCR is essential to an understanding of JTM.

The CCR primitives are used to initiate an atomic action, to indicate readiness to commit to it, to order commitment, to roll it back and to recover following failure.

The CCR service provides a user data parameter on all the primtives.

This user data is used to

a) indicate the minimum level of commitment which is required, and the level actually achieved; and

b) indicate to a subordinate the character sets which the master prefers for diagnostic messages; and

c) indicate, when a subordinate rolls back, whether a subsequent attempt could succeed (retry-later) or will fail for the same reason (no-retry); and

d) carry optional warning messages on C-READY, optional timer values when a later retry is proposed, and a diagnostic message when no retry is proposed.

2.1.8.2 Levels of commitment for JTM

In order to perform an OSI job, the JTM service uses one or more atomic actions, according to the level of commitment requested by the initiation agency. The initial atomic action starts at the point when the initiation agency submits the OSI job to the JTM service provider. (The initiation agency is the commitment master for this initial atomic action).

The JTM service defines three levels of commitment which can be requested by the initiation agency to control the scope of this initial atomic action. The agency specifies the minimum commitment level which it requires, and is informed of the (equal or higher) commitment level actually achieved.

2.1.8.2.1 Completion commitment (level 3)

The highest level of commitment is COMPLETION (level 3). In this case, the entire OSI job is completed as part of the initial atomic action.

When commitment takes place, all the work specified has been completed, and all work specifications created as part of the processing of this OSI job have been fully processed, and have ceased to exist.

This level of commitment is sometimes described as "on-line" processing.

2.1.8.2.2 Agency acceptance commitment (level 2)

The next lower level of commitment is AGENCY-ACCEPTANCE (level 2). In this case, commitment to the initial atomic action means that the initial OSI subjob has progressed at least to the point at which all specified source agencies have been accessed, and all interactions with sink or execution agencies have resulted in the associated documents being secured by the agency, for later completion of the OSI subjob as part of subsequent atomic actions.

2.1.8.2.3 Provider acceptance commitment (level 1)

The lowest level of commitment is PROVIDER-ACCEPTANCE (level 1). (This level is also applied to all atomic actions initiated after the initial atomic action.) At this level, the initial atomic action results in the securing of the created work specification by the JTM service provider, and in it becoming visible to other open systems. There is no guarantee that the specified source documents have been accessed, nor that any material has been passed to sink or execution agencies. This commitment level is always available to a user, even if the local system is currently unable to establish any form of communication with other open systems. It can be described as "off-line" processing.

2.1.8.2.4 Definition of agency actions

A source agency treats all commitment levels as requiring it to deliver the requested material (or to refuse commitment).

A sink or execution agency treats levels 1 and 2 as requiring it to at least secure the document and carry out the action later (or to refuse commitment). It treats level 3 as requiring it to complete the activity (or to refuse commitment).

2.1.8.3 Timeouts

No time-outs are specified for the provision of required responses. An inability to carry out the required actions because of temporary lack of resources (congestion or concurrency control) results in local retry at that open system. The commitment master (for example, the initiation agency) can set a time-out on a local basis and issue a rollback service primitive if the timer expires.

2.1.8.4 User data when starting an atomic action

This user data carries a commitment level parameter and a diagnostic code indicator parameter.

2.1.8.4.1 Commitment Level

The commitment level parameter is an integer. (See 2.1.8.2.) The lowest level of commitment is 1. The highest is 3.

If the commitment level parameter has a maximum value, all results of the action are completed before the subordinate offers commitment. If it has the minimum value, the subordinate need only note the actions to be taken (as secure data), and to perform them some time later.

NOTE — A subordinate can be able to attempt the action at a higher level of commitment than that requested by its superior, possibly involving several new subordinates. If this attempt fails (RETRY-LATER), it can be possible to rollback all the new subordinates and perform the action at the lower commitment level on a purely local basis.

This parameter specifies the minimum commitment level required by the superior. The equivalent parameter on an offer of commitment specifies the commitment level actually achieved, and equals or exceeds that of the parameter when starting the action.

An atomic action started by a subordinate carries an equal or greater level of commitment than that on the start of the action which it received from its superior.

2.1.8.4.2 Diagnostic code indicator

The diagnostic code indicator parameter is a list of zero one or more code specifiers. Each code specifier consists of one or more integers, or the value "ALL". Each integer is a Register Number from the ISO Register of Character Sets (see ISO 2375).

Each code specifier references the one or more graphics character sets in the Register entries with the given numbers. Each subsequent diagnostic message generated by JTM contains only the character space, together with characters from the character sets referenced by one of the code specifiers.

NOTE — Different diagnostic mesages can use character sets referenced by different code specifiers.

The code specifiers appear in the diagnostic code indicator in the order of preference of the CCR master. A code specifier with the single integer 2, specifying the graphics characters of the International Reference Version of ISO 646, is always implicitly present as the least preferred option. An empty diagnostic code indicator specifies ISO 646 only.

The start of an action issued by a subordinate carries a diagnostic code indicator which is equal to that received in the start of the action from its superior.

2.1.8.5 Use of CCR user data when offering commitment

This carries a commitment level parameter, an optional warning diagnostic, and optional accounting information.

2.1.8.5.1 Commitment level

The form of the commitment level parameter is described in 2.1.8.4.1. The value on the offer of commitment is greater than or equal to the value on the start of the atomic action.

NOTE — Where a superior uses subordinates to perform part of an atomic action, the commitment level offered to its superior (if any) does not exceed that offered by any of its subordinates.

2.1.8.5.2 Diagnostic parameter

The diagnostic parameter is optional. If present, it is a warning diagnostic, consists of one or more diagnostic messages, each consisting of

a) the name of the generator of the diagnostic message; and

b) a JTM diagnostic code; and

c) human-readable message text or an FTAM datastructure carrying warnings.

The name of the generator of the diagnostic message is an application-title. Its form is outside the scope of this International Standard. It unambiguously identifies the generator of the message.

The JTM diagnostic code values are listed in the JTM Protocol Specifications.

The human readable message text consists of one or more strings of graphic characters (plus space). Each character is taken from any of the graphics sets referenced by one of the code specifiers in the diagnostic code indicator parameter of the start of the atomic action.

Preference is given to the code specifiers which appear earlier in the diagnostic code indicator. Each string is a line of text, and does not exceed 40 characters in length. It contains zero or more characters.

NOTES

1 The number of strings of characters is not restricted.

2 The human language used in the strings is chosen by the generator of the diagnostic message; the requested character sets can be used as an indication of the preferred language.

2.1.8.5.3 Accounting parameter

This parameter is optional, and provides charging information relating to the atomic action for which commitment is offered. If present, it can cause reports to be generated as described in 2.1.7.

Its form is a list of quadruples, consisting of

a) an (globally unambiguous) account identification (see 2.1.13);

b) a character string resource identifier;

c) a character string charging unit;

d) an integer charge.

The values of b) and c) are assigned by the identification authority (see 2.1.13) associated with the account identification.

2.1.8.6 Use of CCR user data on rollback

The user data is used when the subordinate does not offer commitment and rolls back the atomic action branch. The user data then carries a diagnostic parameter and an optional accounting parameter.

Note that the CCR facilities do not guarantee the transmission of user data on the rollback primitive.

2.1.8.6.1 Diagnostic parameter

The diagnostic parameter is either a retry-later diagnostic or a no-retry diagnostic.

If it is a no-retry diagnostic, it consists of one or more diagnostic messages, each consisting of

a) the name of the generator of the diagnostic message; and

b) a JTM diagnostic code; and

c) human-readable message text or an FTAM datastructure carrying "no-retry" diagnostics.

If it is a retry-later diagnostic, it consists of an optional retry timer and an optional diagnostic message, possibly containing an FTAM datastructure with "retry-later" diagnostics.

The retry timer, if present, indicates that the superior should wait a specified time before attempting the atomic action again. It can be ignored by the superior.

When a superior has multiple subordinates and they each produce diagnostic parameters, the superior discards those of lower severity. The order of severity is defined as

a) warning (lowest);

b) retry-later;

c) no-retry (highest).

2.1.8.6.2 Accounting parameter

This has the form described in 2.1.8.5.3

Note that although the atomic action is in this case rolled back, charges are still levied, and a report of accounting charges can be generated.

Where multiple attempts are made to complete the action, there can be multiple charges.

2.1.9 Transfer control

A transfer control record is used to control which work specifications are eligible for transfer to another (specified) open system, and the number of such transfers which can be in progress simultaneously.

A transfer control record has no effect on the transfer of transfer manipulation work specifications (see 2.1.12). It can control all other types of work specification, depending on the value of the selectors.

An open system with work specifications to transfer to another open system, in the absence of a transfer control record for the destination system, attempts the transfers in an order and with a degree of concurrency decided locally.

A transfer control record, if present, causes the implementation to restrict its transfer attempts to those needed to progress work specifications with specified characteristics.

The presence of transfer control records can prevent commitment to atomic actions. Such refusals are always flagged as "retry later".

2.1.10 Report manipulation

A report manipulation work specification contains one or more report manipulation operations. There are two types of operations defined for report manipulation.

The first is the DELETE operation which causes deletion of all received reports about a specified OSI job or tree of subjobs. Its only parameter is a work specification identification which is used to refer to reports on the specified work specification or on any created from it.

The second is the DISPLAY operation. This has two parameters. The first is the same as above, a work specification identification used to refer to part or all of an OSI job. The second is a document name which is referenced by a proforma contained in the manipulation work specification. This proforma defines an arbitrarily complex subjob for disposal of the named document.

The action by the JTM service provider at the open system is to take each report which was received concerning the specified work specification(s), in the order in which they have been received, and to generate a document containing the reports. If two display operations specify the same document name, both displays are placed in the same document. The end of the activity produces spawning of new work specifications from proformas in the usual way: these collect the display document.

NOTE — Where a monitor point is requested to pass reports directly to a sink agency, storage for display or deletion by report manipulation does not occur.

Authorisation for deletion of reports requires the report manipulation work specification to contain an authority corresponding to the permission information in the report work specification which delivered the report. It is a local option whether authorisation for display is also restricted in such a way. (See 2.1.13.)

2.1.11 Work manipulation

A work manipulation work specification contains one or more JTM work manipulation operations. These provide for the operations of selection, modification, killing, stopping and displaying work specifications and associated subjobs. (The effect of these operations is described below.)

The first operation is a selection; the work specifications or proformas selected are the ones to which all operations preceding the next selection apply. The form of a work specification selector is described in 2.1.11.1.

The modification operation changes certain fields in a work specification or proforma. It can cause an interaction with source, sink or execution agencies associated with the work specification in order to effect a hold or release on activity associated with the work specification.

The kill and stop operations are only applied to a work specification, and can also cause interactions with one or more agencies in order to stop or to kill associated activity. If an uncommitted transfer is in progress, they affect the transfer.

Killing causes the JTM service provider to delete the work specification to which it is applied, and to inform any associated agency that a kill is required. This causes the agency to cease its activity, with rollback if possible. Any uncommitted transfer is rolled back.

Stopping causes the JTM service provider to inform any associated agency that the activity is to be stopped. Normal spawning will occur, and the work specification is not affected. Any uncommitted transfer is rolled back, with retry at a later time.

The display operation causes a copy of parts of the selected work specifications or proformas to be placed in a specified document. The document is collected by normal spawning as for the report manipulation display. The display also contains local status information concerning any source, sink, execution or transfer activity associated with the work specification.

Authorisation for killing, stopping, and modification requires the work manipulation work specification to contain an authority corresponding to a permission (see 2.1.13) in the selected work specification, or in the work specification containing a selected proforma. When an attempt is made to kill, stop, or modify one or more work specifications (selected by a selector), then the manipulation is applied only to those work specifications which contain the necessary permission, and the other work specifications are unchanged. When an attempt is made to display one or more work specifications, it is a local option whether the work specifications which do not contain the necessary permission are displayed or not.

When an unauthorised modification attempt is made by subjob A (say) on a work specification B (say), this results in the VIOLATION-ATTEMPT event for B, which can be reported to one or more monitor points of B.

2.1.11.1 Selectors

A selector specifies a series of tests to be applied to certain fields of a work specification.

The basic tests are of the form

field-name operator value

where the range of permitted field names and of operations is defined in section 3.

For the purposes of selection, the important fields of a work specification are regarded as forming a somewhat simpler data-structure called a **header-list**. The selector selects on the basis of a general logical expression using tests for equality or inequality of these fields against specified values.

Work specifications can also be selected according to the fields of the proformas they contain.

2.1.11.2 Updates

An update is of the form

field-name operator value

and is used in the modification operation to specify the changes to be made. It is specified in section 3. Certain field names which are available for selection cannot be modified.

The header-list conceptual data structure is again used for specifying the update, but changes to its fields produce corresponding changes in the associated work specification.

2.1.12 Transfer manipulation

A transfer manipulation work specification contains one or more transfer manipulation operations.

There are JTM transfer manipulation operations for setting, displaying and checking transfer control records.

The setting operation has a parameter which is a transfer control record.

The display operation has a parameter which identifies a transfer control record, and a second parameter which is a document name; it displays the current transfer control record in the specified document for collection by spawning.

The checking operation contains a copy of a transfer control record which the sending open system is holding in respect of transfers to the receiving open system or to an agency. The receiving open system generates a new transfer manipulation work specification containing a setting operation if this is no longer an appropriate transfer control record for the sender to be using. An open system normally issues checking operations following any period in which it is disconnected from the open systems environment.

A transfer manipulation work specification needs an authorisation element for the management of any open system referenced in the transfer control record of a setting or displaying operation, and also for the management of an open system performing a checking operation. (See 2.1.13.)

2.1.13 Authorisation and accounting

A work specification contains authorisation data. This provides a list of **authorisation elements**. The authorisation element provides an open system with an identification issued by an identification authority whose name it recognises, and a **validation field**. The validation field is either secret data (a password or key) which serves to authenticate the identification, or is a statement that the identification has already been authenticated by an open system earlier in the history of the OSI job.

In order to accept a validation field as having been "already checked", the authenticated identification of the open system which performed the check has to be available. This is obtained in a step-wise manner. The lower layer services provide to the JTM service provider the authenticated name of a calling open system. This data is recorded in the work specification, so that it contains the name of each open system through which the work specification (or one of its ancestors) has passed. Thus an open system is able to determine, by inspection of this data, whether to accept authorisation elements claiming that an identification has already been authenticated. (The detailed operation of this procedure is specified in ISO/IEC 8832.) A further description (and examples), of this mechanism appears in annex C.

An authorisation element can identify a user, or the management of an identification authority, or the management of an open system.

The authenticated identification is used to authorize any activity which is requested, and can, in the absence of a suitable account element (see below), also be used to levy charges.

A work specification can also contain account data. This provides a list of **account elements**. An account element provides an open system with an account identifier issued by an identification authority whose name it recognises, and a validation field.

The authorisation data also names the user identifications which are given permission to make modifications to this work specification. Implicit permission is given to validated identifications representing the identification authority management issuing these identifications, and to those representing the open system management for open systems involved in the OSI job.

An open system's local management decides whether to allow queries and status displays about a work specification without the existence of a permission, or whether to treat a work specification as invisible to anyone other than those with permission to modify it.

Permission to modify a transfer control record is implicitly given to authenticated identifications representing the open system managements involved.

Arrangements between open systems for use of a common identification authority are not excluded, but are not required.

2.1.14 Work specification identification

The work specification identifier provides an unambiguous name by which the work specification can be referenced by a report or by a manipulation work specification.

The first part of a work specification identifier is the name of the OSI job submission system followed by the identification of the initiating user, the OSI job local reference and the OSI job name. The second part of the identifier is a list of (proforma name, qualifying integer) pairs. An element is added to the list whenever a new work specification is created by spawning. The proforma name is the name of the proforma used to spawn the work specification, and the qualifying integer is used to distinguish between work specifications spawned from the same proforma.

In summary, all work specifications have an identification known as the work specification identifier. A complete OSI job, that is the set of all its work specifications, can be referenced by quoting the name of the OSI job submission system and either

a) the initiating user identification and OSI job-name; or

b) the OSI job local reference.

An individual subjob can be referenced by quoting a complete work specification identifier. A group of subjobs with a common origin can be referenced by quoting the work specification identifier of their origin.

2.1.15 Reporting responsibility

Reports are generated either as part of the CCR atomic action progressing the work (and with the same commitment level) – integral reporting, or as part of a separate atomic action – separate reporting. In the former case they are committed if and only if the atomic action proceeds to commitment. When generated by a commitment master, they are either not generated before commitment is offered by subordinates, or are rolled-back if the main action is rolled back (as an implementors option).

In the latter case, they are generated as a result of progressing the main action, and are not affected by rollback of that action.

Generation of reports with a commitment level of one, or generated by separate reporting, is never subject to failure. Subsequent processing of the report can result in failures causing it to be discarded. These events are not reported. At levels two and three, failures to deliver reports generated by integral reporting to agencies causes the entire atomic action to fail.

2.1.15.1 Separate reporting

Reports issued as separate atomic actions are

a) reports of accounting information;

b) reports of attempted security violations;

c) reports of the failure of an atomic action, resulting in no progress of a work specification, or in its abnormal termination.

2.1.15.2 Integral reporting

Reports which are committed only if the main action commits are

a) reports that J-INITIATE has created a new work specification;

b) reports that spawning of a new work specification has occurred;

c) reports that another open system has taken responsibility for progressing a work specification in a subsequent atomic action;

d) reports that AGENCY ACCEPTANCE for the subjob has occurred;

e) reports that an error diagnostic has been included in a work specification (for delivery to a sink agency) in place of a document;

f) reports carrying messages produced during processing activity (and passed to JTM by a J-MESSAGE request from an execution agency);

g) reports of normal termination of a subjob;

h) reports of termination of a subjob, or of modification as a result of manipulation by another work specification (see section 2.1.11);

i) warning reports.

2.1.15.3 Responsibility for separate reporting

Responsibility for issuing reports which are issued as separate atomic actions is

a) for accounting information, that open system containing the master of the atomic action; (note that where the master is a JTM agency, the report is generated by the JTM service provider accessing the agency); the accounting information is carried to the master by the accounting parameter in the user data of the CCR primitives; the number and frequency of such reports is not Standardised;

b) for attempted security violations, that open system with responsibility for the work specification being manipulated;

c) for reports of failure of an atomic action, that open system containing the master of the atomic action; diagnostic messages are carried to the master by the diagnostic parameter in the user data of the CCR primitives.

2.1.15.4 Responsibility for integral reporting

Responsibility for integral reporting rests with the open system most closely involved, thus:

a) for creation, the open system where the J-INITIATE request is issued;

b) for spawning, the open system perfoming the spawning;

c) for transfer of responsibility, the open system taking responsibility;

d) for AGENCY ACCEPTANCE, the open system where the J-DISPOSE indication service primitives are issued;

e) for error diagnostic embedding, the open system performing the embedding;

f) for user messages, the open system where the J-MESSAGE service primitive was issued;

g) for normal termination, the open system which completes the subjob;

h) for manipulation termination, the open system performing the manipulation (and responsible for the work specification being manipulated).

2.1.15.5 Job tracing

The reporting categories of JTM have been designed to permit a sufficiently sophisticated monitor point implementation to maintain a "picture" of the progress of an OSI job, providing it receives all reports in certain categories.

Such job tracing is not required of an implementation by ISO/IEC 8832, and can be performed on a purely manual basis. On the other hand, highly sophisticated JTM implementations are not excluded.

The minimum relevant report categories are

a) creation reports, indicating the creation of a new OSI job, with its identification, the open system responsible for it, its target, and any relays to be used on the way to the target;

b) spawning reports, indicating the activation of a new subjob, with its identification, and the open system responsible for it;

c) transfer of responsibility, indicating a change in the open system responsible for progressing a subjob;

d) normal, abnormal, and manipulation termination, indicating the end of a named subjob.

It can happen that, due to congestion or long-term network failures, some reports are delayed. In general, reports can arrive out of order. The correct order of the above categories can be deduced from the identifications of the subjobs.

Ambiguity arises only when all subjobs appear to have ended, but reports of prior spawning have been delayed. The JTM service enables this ambiguity to be resolved by including with all termination reports a count of the number of subjobs which were spawned.

The JTM service does not provide any mechanism for deducing, on receipt of a subjob termination report, that all reports of accounting information, attempted security violations, error diagnostic insertion, and user messages have been received.

The above list of reporting categories is the minimum needed to determine the current location and states of the parts of an OSI job. Useful additional information is provided by:

a) reports of AGENCY ACCEPTANCE;

b) reports of holding and release of a work specification by modification or of "no-progress".

2.1.16 Documents

JTM defines three document types which carry JTM information. These are JTM-report-display documents (defined in 3.5.1), JTM transfer-display documents (defined in 3.5.3), and JTM work-display documents (defined in 3.5.2).

The semantics of documents of these types are fully defined in this International Standard, and a transfer syntax for such documents is fully defined by the JTM protocol.

In addition to these document types (which are generated by the JTM service provider), the JTM service is also capable of carrying any form of user document for which a transfer syntax has been defined. The definition of the syntax and semantics of other document types is outside the scope of this International Standard, but is essential to its use.

JTM recognises three sorts of document type. These are

a) the JTM-generated document types; these are the three types listed above; and

b) standardised document types; these are document types whose semantics and transfer syntax have been defined elsewhere, and for which there is an entry in **the ISO Register of Document Types**; and

c) enterprise-specific document types; these are document types whose semantics and transfer syntax are determined on an enterprise-specific or bilateral basis.

The JTM protocol static conformance clause recommends that certain JTM implementations provide support for three simple forms of document, designed to carry simple lines of text, text with carriage controls, and an arbitrary length string of bits.

These document types are defined in ISO/IEC 8832. They can all be carried by FTAM where necessary.

There are no JTM conformance requirements in relation to support for other document types such as those supporting graphical transfers, office documents, or data descriptive files. Support by JTM implementations for such document types is optional.

NOTE — Annex B defines the information required in a Document Type Register Entry for the registration or distribution of definitions of enterprise-specific document types.

2.1.17 JTM relays

Each JTM work specification has a **target** open system. This is present in the original work specification when the OSI job is created. It is obtained from proformas when new work specifications are spawned, and from monitor data when reports are generated.

Each of these sources of information can additionally specify one or more **relays** to be used in transferring the work specification to the target.

NOTE — These are JTM application-layer relays, they are not communications relays.

The work specification is transferred to the first relay, then to the second, and so on, until the final relay transfers it to its target. If reporting of the TRANSFER event is selected, reports are generated for each of these transfers.

Each relay performs normal initial processing to resolve any source references for which it is the action open system, then queues the work specification for onward transfer.

It is visible for manipulation and subject to transfer control in exactly the same way as work specifications which are generated by OSI job creation or spawning, and are queued for transfer (or are being transferred).

Examples of the use of relays are given in annex C.

2.2 Overview of the service

The JTM service primitives defined in section 3 of this International Standard are specified in terms of the model elements described in 2.1. These JTM service primitives provide for performing all job-related activities between open systems.

They provide for

a) the submission of an OSI job to the JTM service provider from an initiation agency; (OSI jobs can be submitted to move documents between source, sink and execution agencies; OSI jobs can also be submitted for display or manipulation of any previously submitted OSI jobs, that is, to display and modify the actions of the JTM service provider or the current activities of agencies);

b) the delivery of documents to sink agencies, together with an interaction about the end of the activity caused by the delivery;

c) the collection of documents from source agencies;

NOTE — The use of FTAM by source or sink agencies is fully supported.

d) the delivery of documents to execution agencies for processing, together with an interaction about the end of the activity of the agency caused by the delivery.

2.3 Basic and extended implementations

NOTE — A complete definition of Basic Class JTM services is given in section 4.

A Basic Class implementation and an extended implementation are recognised. A Basic Class implementation supports only a subset of the possible range of work specifications, and supports only a subset of the range of parameters on JTM service primitives.

Interworking between extended and Basic Class implementations is possible. In particular it is possible for work specifications generated on Basic Class implementations to be transferred to and processed by extended implementations; the converse is also possible if the work specification is within the range covered by Basic Class (see section 4 and clause C.17).

The following are the main restrictions and characteristics relating to Basic Class implementations:

a) a work specification carries at most one document;

b) a work specification carries at most one top-level proforma; any embedded proformas are wholly transparent to a Basic Class implementation;

c) errors detected when collecting a document from an execution agency result in embedded diagnostics; other errors result in abnormal termination;

d) the only reporting which can be selected is

1) abnormal termination;

2) normal termination;

3) manipulation termination;

4) user messages.

e) accounting is not supported;

f) commitment level 3 (COMPLETION) can not be requested;

g) transfer control records are not supported;

h) report manipulation is not supported;

i) only selection (by "subjob type", "OSI jobname", and "OSI job submission system"), "brief display", "stop", and "kill" operations are supported;

j) use of FTAM for obtaining or disposing of documents is not supported;

k) there are no secondary monitors;

l) the "hold" concept is not supported;

m) the following primitives are not supported;

1) J-INITIATE-TCR-MAN

2) J-INITIATE-REPORT-MAN

3) J-ENQUIRE

4) J-HOLD

5) J-RELEASE

6) J-SPAWN.

Another important characteristic of a JTM implementation is the types of agency which it supports, and the range of real devices or processing functions that these are mapped onto.

A JTM implementation might support only an initiation agency or only an execution agency; it might support only a sink agency mapped to a single one of its many printers. On the other hand, it might support the mapping of all its devices and functions onto JTM agencies, and support a full range of agencies.

This agency support dimension is separate from the Basic Class and extended distinction.

2.4 Model for the service specification

JTM services are provided by JTM Specific Application Service Elements, the service elements for Association Control and for Commitment, Concurrency and Recovery and lower layer entities. These together constitute the JTM service provider. The JTM services are defined in relation to and in terms of the JTM model elements which have been described earlier.

The service primitives represent conceptual interactions between the JTM service provider and the different agencies in the model (i.e. initiation, source, sink and execution agencies, constituting the users of the service).

According to further specific JTM mechanisms, such as spawning of new work specifications from proformas, a service request from one user can give rise to a set of interactions between the JTM service provider and agencies. This constitutes a one-to-many correspondence between JTM service users. (See figure 1.)

An initiation agency interacts with the JTM service provider to provide all the information necessary to define an initial work specification with its proformas (that is, the whole work to be performed), and there are service primitives related to this.

The main unit of interaction between the JTM service provider and sink, source and execution agencies is the document, and a set of service primitives is provided to support these interactions for any type of document.

Another set of service primitives relate to the manipulation of work specifications.

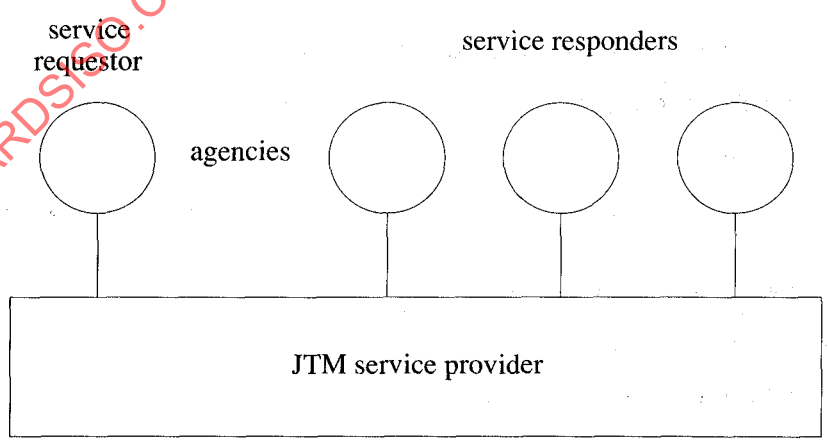

Figure 1 — JTM users

Finally, there are a set of service primitives related to the monitoring and status concepts in the JTM model.

All these service primitives are defined in section 3. A summary is given at the end of that section. The name of each service primitive has been chosen to be in line with the concepts of the JTM model, and relates to a specific interaction between a specific agency and the service provider. Several of these service primitives are needed to describe the complete processing of a single OSI job.

Section 3 : Definition of primitives

3.1 JTM service primitives

An agency provides an abstract representation of part of a real open system.

Each conceptual interaction between the JTM service provider and an agency is defined by the issue of a sequence of service primitives (called a **JTM service primitive group** or a **commitment group**).

This International Standard associates Commitment, Concurrency, and Recovery semantics (as defined in ISO/IEC 9804) with these service primitives. This provides a means of defining the points in the JTM protocol when externally visible events are to occur, and of stating the commitment, concurrency, and recovery capability which the total real system is to exhibit.

Provided the external behaviour corresponds with that of the model open system, no constraints are placed on the internal structure, interfaces, or procedures of the real system.

3.1.1 Service primitive groups

A JTM service primitive group is a sequence of service primitives, some of which are mapped onto the CCR service primitives of ISO/IEC 9804, and some of which have a purely JTM function.

3.1.1.1 Creation of new work specifications

There are four groups of primitives (initiated by an initiation agency) which each define the creation of a new work specification in an open system. These are

J-INITIATE-WORK — this group is used by an initiation agency to create a work specification for document movement;

J-INITIATE-WORK-MAN — this group is used by an initiation agency to create a work specification for the manipulation of existing work specifications;

J-INITIATE-REPORT-MAN — this group is used by an initiation agency to create a work specification for the manipulation of reports received by an OSI job monitor;

J-INITIATE-TCR-MAN — this group is used by an initiation agency to create a work specification for manipulation of JTM transfer control records;

Note that where a statement applies to all the above primitive groups, the term "J-INITIATE" is used. In particular, the only difference between the parameters of these primitives is in the nature of the JTM action parameters outside of proformas, and the Basic Class restrictions.

3.1.1.2 Document movement

There are three groups of primitives initiated by the JTM service provider as the result of processing a document movement work specification, that is, as the result, directly or indirectly, of a J-INITIATE group of primitives. The first group is issued to a sink or execution agency, and the last two groups to an execution agency or source. These groups are

J-DISPOSE — this group is used by the JTM service provider to pass a document to a sink or execution agency, creating a new activity in that agency;

J-GIVE — this group is used by the JTM service provider to request a document from a source or execution agency;

J-ENQUIRE — this group is used by the JTM service provider to obtain from a source or execution agency a list of document names for satisfaction of references.

3.1.1.3 Other agency-initiated groups

There are three groups of primitives which are initiated by a sink or execution agency in relation to a specific activity which is in progress in the agency. These are

J-MESSAGE	(execution agency only) this group is used by the agency to cause the JTM service provider to generate a USER-MESSAGE report work specification;
J-SPAWN	(execution agency only) this group is used by the agency to specify that demand spawning from a specified proforma is to occur;
J-END-SIGNAL	this group is used by a sink or execution agency to signal the end of an activity which is in progress following an earlier ACCEPTANCE commitment.

3.1.1.4 Other provider-initiated groups

There are five groups of primitives which are initiated by the JTM service provider to a sink or execution agency to control an activity which is in progress following earlier ACCEPTANCE commitment. These are

J-STATUS	this group is used by the JTM service provider to obtain information about the progress of an activity;
J-HOLD	this group is used by the JTM service provider to request temporary suspension of the progress of an activity (its only defined effect is to prevent the issue of the primitive groups of 3.1.1.3; other effects are not standardised);
J-RELEASE	this group is used by the JTM service provider to cancel an earlier J-HOLD;
J-KILL	this group is used by the JTM service provider to cause termination of the activity; this causes an immediate J-END-SIGNAL, and no documents are provided by an execution agency; it is desirable, but not required, for all effects of the activity being killed to be undone;
J-STOP	this group is used by the JTM service provider to cause termination of the activity; this causes an immediate J-END-SIGNAL, but documents can be supplied; it is recommended that any work already done should not be undone, but this is not required.

3.1.2 Components of service primitive groups

3.1.2.1 Groups initiated by an agency

These groups consist of the following primitives (in order):

a) a J-BEGIN request; followed by

b) one of

1) J-INITIATE-WORK request and confirm; or

2) J-INITIATE-WORK-MAN request and confirm; or

3) J-INITIATE-REPORT-MAN request and confirm; or

4) J-INITIATE-TCR-MAN request and confirm; or

5) J-MESSAGE request; or

6) J-SPAWN request; optionally followed by the results collection sequence specified in 3.1.2.3; or

7) J-END-SIGNAL request; optionally followed by the results collection sequence specified in 3.1.2.3; followed by

c) one of

1) a J-READY indication; or

2) a J-ROLLBACK indication and response; followed, in the case of a J-READY indication only, by

d) one of

1) J-COMMIT request and confirm ; or

2) J-ROLLBACK request and confirm.

The primitives in a), c), and d) carry only CCR parameters, and correspond exactly to CCR service primitives of the same name. Their sequence, semantics and parameters are completely defined by ISO/IEC 9804, except for the user data parameter (see 3.1.3). The agency is the commitment master (and issues the CCR atomic action identifier and defines the commitment level) in all cases except the J-MESSAGE and J-SPAWN group. Where these groups are issued following ACCEPTANCE commitment, the agency is the commitment master, otherwise it is a sub-master. Where the agency is a sub-master, it chooses a commitment level greater than or equal to that received. For a J-END-SIGNAL request, the agency always specifies PROVIDER-ACCEPTANCE (level 1), and does not refuse commitment if the JTM service provider offers commitment.

3.1.2.2 Groups initiated by the JTM service provider

These groups consist of the following primitives (in order):

a) a J-BEGIN indication; followed by

b) one of

1) J-DISPOSE indication and response; optionally followed by the results collection sequence specified in 3.1.2.3; or

2) J-GIVE indication and response; or

3) J-ENQUIRE indication and response; or

4) J-STATUS indication and response; or

5) J-HOLD indication; or

6) J-RELEASE indication; or

7) J-KILL indication; or

8) J-STOP indication; followed by

c) one of

1) a J-READY request; or

2) a J-ROLLBACK request and confirm ; followed, in the case of J-READY request only, by

d) one of

1) J-COMMIT indication and response ; or

2) J-ROLLBACK indication and response.

The primitives in a), c) and d) carry only CCR parameters, and correspond exactly to CCR service primitives of the same name. Their sequence, semantics and parameters are completely defined by ISO/IEC 9804, except for the user data parameter (see 3.1.3). The JTM service provider is the commitment master (and issues the CCR atomic action identifier) if and only if it has already given ACCEPTANCE commitment to the J-INITIATE, J-SPAWN or J-END-SIGNAL group which generated the work specification causing the primitives to be issued, otherwise it is a submaster. When the JTM service provider is the master it always requests commitment level 1.

3.1.2.3 Results collection sequence

This sequence consists of one or more repetitions of

a) J-ENQUIRE indication and response; or

b) J-GIVE indication and response

3.1.3 Parameters of CCR-related primitives

The parameters of the CCR-related primitives identified in 3.1.2.1 and 3.1.2.2 are specified in ISO/IEC 9804. The user-data parameter specified in ISO/IEC 9804 is used to carry JTM-specific CCR-related parameters as defined below.

3.1.3.1 J-BEGIN request and indication

These primitives carry

a) commitment level (see 2.1.8.4.1);

b) diagnostic code indicator (see 2.1.8.4.2).

3.1.3.2 J-BEGIN response and confirm

The user data parameter is not used.

3.1.3.3 J-PREPARE request and indication

The user data parameter is not used.

3.1.3.4 J-READY request and indication

These primitives carry

a) commitment level (see 2.1.8.5.1);

b) optional warnings (see 2.1.8.5.2);

c) optional accounting information (see 2.1.8.5.3).

3.1.3.5 J-COMMIT request and indication

The user data parameter is not used.

3.1.3.6 J-COMMIT response and confirm

The user data parameter is not used.

3.1.3.7 J-ROLLBACK request and indication

When issued from a subordinate to a superior, these primitives carry

a) a diagnostic (see 2.1.8.6.1);

b) optional accounting information (see 2.1.8.6.2).

3.1.3.8 J-ROLLBACK response and confirmation

The user data parameter is not used.

3.1.3.9 J-RECOVER request and indication

The user data parameter is not used.

3.1.3.10 J-RECOVER response and confirm

The user data parameter is not used.

3.1.4 Sequence of service primitive groups

The following groups are issued at any time:

J-INITIATE-WORK
J-INITIATE-WORK-MAN
J-INITIATE-REPORT-MAN
J-INITIATE-TCR-MAN
J-DISPOSE
J-GIVE
J-ENQUIRE

There is an additional restriction on the J-GIVE and J-ENQUIRE groups when issued to an execution agency. In this case they are only issued (and complete) between a J-END-SIGNAL or J-SPAWN request, and the corresponding J-READY or J-ROLLBACK indication or between a J-DISPOSE response where COMPLETION commitment is offered and the corresponding J-COMMIT or J-ROLLBACK indication.

The following groups are issued between a J-DISPOSE indication and the corresponding J-READY or J-ROLLBACK request:

J-MESSAGE
J-SPAWN

The following groups are issued between a J-COMMIT response in a J-DISPOSE group which gave ACCEPTANCE commitment, and the J-BEGIN request commencing the J-END-SIGNAL for the activity:

J-MESSAGE
J-SPAWN
J-END-SIGNAL

A J-END-SIGNAL group is not started if any other group is incomplete in relation to the same activity, nor is a J-READY request to a J-DISPOSE indication issued if a J-MESSAGE or J-SPAWN sequence is incomplete.

The following groups are started if and only if no other group is in progress in relation to the same activity:

J-STATUS
J-HOLD
J-RELEASE
J-KILL
J-STOP

In addition to these primitives, J-RECOVER is defined with the same semantics and parameters as C-RECOVER, and can occur as defined in ISO/IEC 9804.

3.1.5 Sequence of service primitives

An arbitrarily complex JTM activity involves many primitive groups. The possible sequences of all the primitives in these groups can be determined, for any specific JTM activity, by applying the following rules:

a) the sequence within a primitive group is determined by 3.1.2;

b) the sequence of primitive groups issued by or to a single JTM entity for a single JTM subjob is determined by 3.1.4;

c) the sequence of primitives with CCR semantics (at the same or at different open systems) is determined by the CCR Service definition;

d) where data provided by one primitive (e.g. a J-GIVE response) is needed for the issue of another primitive (e.g. a J-DISPOSE indication), then the sequence of these primitives is determined by this requirement;

e) there are no other constraints on the sequence of the service primitives.

An example applying these rules to specific JTM activity is provided in annex C.

NOTE — An implementation can support only a single agency (initiation, sink, source, or execution), or can support several agencies of the same or different types. Where a type of agency is not supported, service primitive groups interacting with that type of agency do not occur on that open system.

3.2 Notation for service primitive and datastructure definition

This clause specifies the notation used for defining the parameters of the JTM service primitives which are not CCR-related. (The parameters of the CCR-related primitives have a less complex structure, and are defined in 3.1.3.)

Each JTM service primitive parameter or field in a JTM conceptual datastructure is either a basic datatype or a structured datatype. Each structured datatype is constructed using one of the structuring mechanisms in 3.2.2 below applied to either basic datatypes or to further structured datatypes.

3.2.1 Basic datatypes

There are six basic datatypes used by JTM. These are

a) literals;

b) integer datatypes;

c) string datatypes;

d) name datatypes;

e) referenced datatypes;

f) document datatypes.

A literal is specified by giving the literal name in capitals. Each literal is an unambiguous identification whose semantics are entirely defined in this service definition. For example:

MOVE HIGH CREATION

All other basic datatypes are written with a lower case name enclosed in angle brackets, with an "=I", "=S" or "=OS", "=N" or "=NA" or "=ND", ="R", and "=D" included to denote the basic datatypes b) to f) above. For example:

integer	<qualifying integer = I>
string	<OSI job name =S> <source name = S>
name	<authority name = NA> <open system name = N> <document type = ND>
referenced	<security parameters = R>
document	<basic document = D>

The definition of these basic datatypes is given in 3.2.3 below.

3.2.2 Structuring mechanisms

There are four mechanisms (and corresponding notation) for forming a structured datatype. These are

a) an ordered sequence of datatypes; the datatypes are listed; for example, the datatype <OSI-subjob parameters> is defined as

<subjob name-list>
<target list>
<urgency>
<holds>
<subjob type>

each of which is another structured datatype;

b) zero one or more repetitions of a single datatype, where order is important and duplicates are allowed (a list); the single datatype is included in square brackets followed by "*L"; for example, the datatype <document list> is defined as

[<basic document = D>]*L

c) zero one or more repetitions of a single datatype, where order is not significant, and where duplicate values can not occur (a collection); the single datatype is included in square brackets followed by "*C"; for example, the datatype <permissions> is defined as

[<permission element>]*C

d) a choice from two or more alternatives, often (but not always) literals; the alternatives are separated by vertical bars and included in round brackets; for example, the datatype <urgency> is defined as

(HIGH | MEDIUM | LOW)

A structured datatype is defined by placing its name on the left of a "::=" symbol, with the structure notation on the right. For example

<permissions>::= [<permission element>]*C

3.2.3 Definition of basic datatypes

3.2.3.1 Literal and integer datatypes

The meaning of each literal is defined as it occurs.

An integer is any value from zero upwards, and is not bounded.

3.2.3.2 String datatypes

A string datatype is either a sequence of characters plus space taken from any character set registered for use as a G0, G1, G2, or G3 set in the ISO Register of Character Sets, or is an octet string. In the former case "=S" is used, in the latter case "=OS" is used.

The JTM protocol conformance requirements specify that an implementation

a) supports the specification of "=S" string fields using at least

1) the ISO 646 International Reference Version character set; and

2) hexadecimal notation for the encoding of the string in accordance with ISO 2022; and

b) supports the display of "=S" string fields which contain only ISO 646 characters using the character repertoire, and supports the display of other string fields using either the character repertoire or using hexadecimal notation for the encoding of the string in accordance with ISO 2022; and

c) supports the specification and display of "=OS" string fields using the hexadecimal notation.

Where a string is used for naming, the non-ambiguity requirements are met by this International Standard by associating with the string field another datatype "name" which limits the scope of the string (see 3.2.3.3). For example, the identification of a user is expressed as

<authority name = NA>
<user identification = S>

where <user identification> is required to be unambiguous within the scope of <authority name>.

The length of a string is unbounded.

3.2.3.3 Name datatypes

A **name** is a data-type which provides a globally unambiguous identification. The definition of its form is outside the scope of this International Standard. The means by which unambiguous names are assigned is outside the scope of this International Standard. There are only three sets of entities for which JTM requires unambiguous names. These are

a) application-entities containing a JTM Specific Application Service Element (a JTM application-entity);

b) user identification authorities;

c) document types.

A datatype which identifies a JTM application-entity is denoted by "=N". A datatype which identifies a user identification authority is denoted by "=NA". A datatype which identifies a document type is denoted by "=ND". It is possible, but not required, for the same name to identify both a JTM application-entity and a user identification authority.

Names are unambiguously assigned by ISO Registration Authorities, with escape mechanisms to support private registration authorities. Names assigned by a private registration authority are only unambiguous within the scope of a particular enterprise.

Names of objects of type a) and b) have the form and allocation mechanisms defined for OSI application-entity-titles. Names of type c) have the form and allocation mechanisms defined for the OBJECT IDENTIFIER datatype in ISO/IEC 8824.

3.2.3.4 Referenced datatypes

A referenced datatype is defined by some other International Standard. The accompanying text will identify the International Standard defining the datatype. The form of these datatypes is not constrained by this International Standard.

NOTE — The only use currently made of these datatypes is for the security parameters.

3.2.3.5 Document datatypes

A document is a datatype defined using any notation which provides (either directly or by using an abstract syntax notation):

a) a semantic definition; and

b) one or more transfer syntaxes; and

c) rules for concatenation of the document with other document types, or with other instances of the same type.

NOTE — This information is sufficient for JTM purposes. Other uses of Document Type Registration can require additional information.

Document datatypes are defined in this International Standard (see clause 3.5), in other International Standards or CCITT Recommendations, or directly in the ISO Register of Document Types. All documents carried by JTM have associated with them an unambiguous name (=ND) assigned when they are defined.

3.3 JTM events and report parameters

The categories of event which can cause the JTM service provider to generate reports are listed in 3.3.1. The parameters carried by a report of an event in each of the categories are listed in 3.3.2.

Event type	Target list	Number spawned	Text	Diag-nostic	hold	security state	stop data	accounting signal	accounting information
CREATION	●		●		●				
TRANSFER	●		●		●				
SPAWNING	●		●		●				
AGENCY-ACCEPTANCE			●						
ERROR-DIAGNOSTIC			●	●					
NO-PROGRESS			●	●					
USER-MESSAGE			●						
ACCOUNTING-DATA			●					●	
NORMAL-TERMINATION		●	●						
MANIPULATION-TERMINATION		●	●						
MODIFICATION	●		●		●		●		
NOT-SUPPORTED-TERMINATION		●	●						
ABNORMAL-TERMINATION		●	●	●					
VIOLATION-ATTEMPT			●			●			
WARNING-REPORT			●	●					

3.3.1 Event categories

The processing of a work specification involves one or more atomic actions during which responsibility for progressing the work can be transferred from one open system to another. Responsibility is transferred only when an atomic action commits. Reports can be generated by the master of the atomic action, or by the open system first detecting the occurrence of an event. This is defined for each event. Where the following definitions specify that a report is to be produced as part of some atomic action, the branch of the atomic action producing the report commits if and only if the main atomic action commits; the main atomic action fails if the branch of the action progressing the report fails. The commitment level for the report follows the normal rules. Where the following definitions specify that the report is issued as a separate atomic action, the commitment level is one, and success or failure of the two actions is independent.

The event categories are listed and defined below.

CREATION — A report is to be produced by the open system where J-INITIATE is issued, as part of that atomic action;

TRANSFER — The JTM protocol operates by transferring between open systems the conceptual data structures (work specifications) defining the remaining parts of the OSI job; this literal requires an open system taking responsibility for a work specification (that is, committing with PROVIDER-ACCEPTANCE or AGENCY-ACCEPTANCE commitment level) to generate a report as part of the atomic action transferring responsibility;

SPAWNING — A report is to be produced by an open system when it uses the data in a proforma to initiate a new subjob (spawning); this report is produced as part of the atomic action performing the spawning;

AGENCY-ACCEPTANCE — A report is to be produced by the open system which is the master of an atomic action processing a work specification for which it is responsible when an AGENCY-ACCEPTANCE commitment level (or greater) is achieved for all branches of the atomic action; this is produced as part of the atomic action;

NOTE — This is not produced if all branches of the action produced COMPLETION commitment level, as this generates the NORMAL-TERMINATION event.

NORMAL-TERMINATION — A report is to be produced by a target open system which is processing a work specification when all branches of the atomic action produce COMPLETION commitment; it is generated as part of the atomic action; it is also produced as part of the J-END-SIGNAL atomic action when a work specification is discarded following any necessary spawning; note that spawning reports can also be produced as part of this same atomic action, and that the requested minimum commitment level for J-END-SIGNAL is always PROVIDER ACCEPTANCE;

MANIPULATION-TERMINATION — A report is to be produced by the open system performing the manipulation if the work specification is cancelled by a KILL manipulation operation; this is generated as part of the manipulation atomic action;

MODIFICATION — A report is to be produced by the open system performing the manipulation whenever the work specification is modified by a manipulation operation, or stopped by a STOP operation; it is generated as part of the manipulation atomic action;

ERROR-DIAGNOSTIC — A report is to be produced by an open system whenever a document source reference (see 3.4.4.1.3) in a work specification is replaced by an error diagnostic; it is generated as part of the main atomic action;

NO-PROGRESS — A report is to be produced whenever an open system responsible for a work specification, and master of an atomic action trying to progress it, receives C-ROLLBACK or J-ROLLBACK indication with a NO-RETRY diagnostic and retains the work specification; it is also generated if continuous rollbacks with RETRY-LATER diagnostics recur for an implementation-dependent time; the CCR diagnostic is made available in the report, and the work specification is retained for correction by subsequent modification; this report is generated as a separate atomic action;

ACCOUNTING-DATA — A report is to be produced by an open system whenever charging information relating to this OSI job becomes available; the report is generated as a separate atomic action by the open system which is the master of an atomic action processing a work specification for which it is responsible;

USER-MESSAGE — A report is to be produced by an open system when the J-MESSAGE request is issued by an activity in an execution agency; it is generated as part of the J-MESSAGE atomic action;

NOT-SUPPORTED-TERMINATION — A report is to be produced by an open system responsible for a work specification and master of the atomic action if the complexity of the subjob exceeds the capabilities of the open system processing it; it is generated as a separate atomic action and the work specification is discarded;

NOTE — An open system checks a work specification received in a transfer element, and refuses the transfer if it is unacceptable; this normally produces a NO-PROGRESS report by the sender; the NOT-SUPPORTED-TERMINATION event only occurs when a proforma is used for spawning which exceeds the capabilities of the spawning system.

ABNORMAL-TERMINATION
A report is to be produced whenever an open system responsible for a work specification, and master of an atomic action trying to progress it, receives a J-ROLLBACK or C-ROLLBACK with a NO-RETRY diagnostic and discards the work specification; it is also generated if continuous rollbacks with RETRY-LATER diagnostics recur for an implementation-dependent time; the CCR diagnostic is made available in the report; this report is generated as a separate atomic action;

VIOLATION-ATTEMPT A report is to be produced if a manipulation work specifica tion attempts to modify, DISPLAY, STOP, or KILL this work specification, and does not contain an authorisation element permitting the operation; it is generated as a separate atomic action.

WARNING-REPORT A report is to be produced if, during successful JTM activity, warnings are produced relating to the results or effects of that activity.

3.3.2 Reports

A report consists of the following information:

<report>::=	<name of reporter = N> <time stamp> <subjob identification> <event identification> <event parameters>	(see 3.4.2.2)
<time stamp>::=	(NULL | <date-time = S>)	
<subjob identification>::=	<OSI job submission system = N> <initiating identification> <initiating time> <OSI job name = S> <OSI job local reference = S> <subjob name-list> <subjob type>	(see 3.4.3.3)
<initiating identification>::=	<identification>	
<identification>::=	(<open system management> | <authority management> | <user>)	
<open system management>::=	<open system name = N>	
<authority management>::=	<authority name = NA>	
<user>::=	<authority name = NA> <user identification = S>	
<initiating time>::=	<time stamp>	
<subjob name-list>::=	[<subjob name component>]*L	
<subjob name component>::=	<proforma name = S> <qualifying integer = I>	(see 3.4.5)
<event parameters>::=	(NULL | <target list>) (NULL | <number spawned = I>) (NULL | [<text = S>]*L) (NULL | [<diagnostic information>]*C) (NULL | <security data>) (NULL | <hold state>) (NULL | <stop signal>) (NULL | <accounting information>)	(see 3.4.3) (see 3.3.3) (see 3.3.3.1) (see 3.3.3.2)

where

<hold state>::= (HELD | NOT-HELD)

<stop signal>::= (STOPPED | MODIFIED)

The <name of the reporter> specifies the open system generating the <report>. The <time stamp> carries date and time of report generation, in the form specified for GeneralizedTime in ISO/IEC 8824, if this is available on an open system, otherwise it is NULL. Support for a time-stamp is optional in all JTM implementations.

The <subjob identification> identifies the subjob being reported on; its fields are copied from the work specification being reported on.

The <OSI job submission system> provides a globally unambiguous identification of the OSI job submission system. Other naming strings are unambiguous only within this scope.

The <initiating identification> is also globally unambiguous, and identifies the management of an open system, the management of a user identification authority, or a user.

The <initiating time> is inserted by the JTM service provider at the time of a J-INITIATE.

The <OSI jobname> is supplied by J-INITIATE and is used to identify the whole OSI job. There are no requirements that this should have different values on different OSI jobs.

The <OSI job local reference> is inserted by the JTM service provider (and returned in the J-INITIATE response). It unambiguously identifies the OSI job within the scope of the <OSI job submission system>.

The <subjob name-list> is empty for the first subjob of an OSI job. It is generated by the JTM service provider on spawning by appending a <subjob name component> to the <subjob name-list> of the parent subjob. The <subjob name component> contains the name of the proforma from which the subjob was spawned, and an integer which is the count of subjobs spawned from this proforma. (That is, the first or only subjob spawned from a proforma has a qualifying integer of one.)

The <subjob type> is defined in 3.4.3.3. It identifies the type of the subjob being reported on. It can never have the value "REPORT-MOVEMENT" when used as above, as reports are not generated on such subjobs.

The <event identification> identifies the event category being reported on. Event categories are defined in 3.3.1.

The <event parameters> are supplied as shown in table 1 for each type of event.

The ERROR-DIAGNOSTIC, ABNORMAL-TERMINATION, NO-PROGRESS and WARNING-REPORT reports are the only ones carrying defined diagnostics (see 3.3.3 for the form of defined diagnostics). The <text> strings are used in other cases to carry diagnostic information, and are not standardised.

The text-strings for the USER-MESSAGE *event are obtained from the* J-MESSAGE *primitive* (see 3.6.5).

The length of each <text> string is restricted to 40 characters.

The <target list> specifies the <target> and <relays> of the new work specification; it is defined in 3.4.3. The <number spawned> specifies the number of work specifications spawned from the old work specification.

The <security data> is defined in 3.3.3.1, and provides an identification of the work specification **causing** a VIOLATION-ATTEMPT event. (Thus the <subjob identification> in it differs from that in the head of the <report>, which is the work specification on which the violation attemmpt was made.) The <security data> contains a copy of the audit trace to permit detection of masquerade.

The <hold state> is HELD if one or more elements in the hold list prevent further progress, and is NOT-HELD otherwise.

The <stop signal> is STOPPED if the modification included a STOP operation, and MODIFIED otherwise.

NOTE — A manipulation producing KILL generates a MANIPULATION-TERMINATION event, and not a MODIFICATION event.

3.3.3 Diagnostic information parameter

<diagnostic information>::= <generator details>

	<time stamp>	(see 3.3.2)
	<CCR diagnostics>	(see below)
<generator details>::=	(<source details> | <se details> | <recipient = N> | PROVIDER)	

NOTE — In the naming of datatypes "se" has been used as a mnemonic for "sink or execution".

<source details>::=	<source identification>	(see 3.4.4.1.3)
	<document source reference>	(see 3.4.4.1.3)
<se details>::=	<se identification>	(see 3.4.4.1.1)
	<document se reference>	(see 3.4.4.1.2)

The <diagnostic information> is generated by the JTM service provider when it is a commitment master and has <CCR diagnostics> (see below) to process. The <CCR diagnostics> are returned from sources, sinks, or transfer attempts and may have been generated by several open systems.

The <source details> identify the CCR subordinate and some parameters of J-GIVE when this produced the J-ROLLBACK. The <se details> identify the CCR subordinate and some parameters of J-DISPOSE. The <recipient> identifies another open system when a transfer attempt fails. PROVIDER identifies the JTM application-entity as itself the generator of the <diagnostic information>.

The <CCR diagnostic> is initially carried in a J-READY or a C-READY (for a diagnostic of WARNING only), or in a J-ROLLBACK or C-ROLLBACK for RETRY-LATER or NO-RETRY.

<CCR diagnostics> of WARNING are embedded in <diagnostic information> for the generation of WARNING REPORTS.

<CCR diagnostics> of NO-RETRY are embedded in <diagnostic information> for abnormal-termination, no-progress and error-diagnostic reports.

<CCR diagnostics> of RETRY-LATER are never embedded in <diagnostic information> unless repeated rollbacks with retry-later occur.

<CCR diagnostics>::=	([Warning]*C | [<Retry later>]*C | [<No retry>]*C)
<Warning>::=	<generator = N> <code> (<reason> | <FTAM outcome>)
<Retry later>::=	<generator = N> <code> (<reason> | <FTAM outcome>) <retry timer = I>
<No retry>::=	<generator = N> <code> (<reason> | <FTAM outcome>)
<FTAM outcome>::=	<"Write transfer outcome" or "Read transfer outcome" as defined in annex D of ISO 8571-3>

The <generator> identifies the open system producing the CCR diagnostic, and the <code> is one of the values listed in the JTM protocol specification. <reason> is human-readable text (see below).

The <FTAM outcome> is a datastructure capable of carrying all the information specified in annex D of ISO 8571-3 which results from a transfer attempt using FTAM.

The <retry timer> gives the suggested time delay before retrying the atomic action. The time is 2**I seconds, where I is the value of <retry timer>.

The <reason> consists of one or more <text> strings. Each <text> string consists of from zero to 40 characters.

NOTE — This datatype is used to carry the human-readable portion of a JTM diagnostic. It has been chosen so that display of the diagnostic using a two-dimensional medium of width forty characters is possible. The generator of a diagnostic should note this intended model of a display.

3.3.3.1 Security data parameter

The <security data> is copied from the work specification attempting the modification which caused the VIOLATION-ATTEMPT.

<security data>::=	<audit trace>	(see 3.4.2)
	<subjob identification>	(see 3.3.2)

The <audit trace> identifies the source of the work specification attempting the unauthorised modification or display. The use of the <audit trace> for this purpose gives protection against masquerade in the use of other fields. The <subjob identification> is copied from the offending work specification and provides information enabling it to be identified.

3.3.3.2 Accounting information parameter

<accounting information>::=	[<charging information>]*C	
<charging information>::=	<identification>	(see 3.3.2)
	<resource-name = S>	
	<charging-unit = S>	
	<charge = I>	

The <accounting information> becomes available locally or as the result of a C-READY or C-ROLLBACK in a transfer attempt.

It is returned to the commitment master using CCR mechanisms, and then to the JTM monitor points using the reporting mechanisms.

The <identification> identifies the account which is being charged. The <resource-name> and <charging unit> are names issued by the authority in the <identification> for the purposes of charging. The same names may also be used for authorisation.

The <identification> identifies a unit of accountability, the <resource-name> identifies a resource which has been consumed, and the <charging-unit> and <charge> contain the charges levied.

3.4 Fields of the conceptual data structures

All JTM activity is governed by conceptual data structures called work specifications.

A J-INITIATE primitive causes a work specification to be created using a combination of parameters from the J-INITIATE primitive and data supplied by the JTM service provider. As a result of processing this work specification at the local system, it can undergo some changes (for example, by the addition of "already checked" authorisation elements, or by the inclusion of documents obtained by J-GIVE).

Further processing can cause the semantics of the work specification to be passed to another open system, using the syntax of a JTM Transfer Element (the JTM PDU). Some semantic changes are made to the datastructure when it is transferred, notably the addition of information about where it came from.

A work specification contains a detailed specification of an OSI subjob, together with proformas. Proformas in a work specification are handled as "black boxes" until they are used for spawning or are the subject of manipulation. At the time of spawning they are "opened up", and work begins on the subjob they specify. When proformas spawn new subjobs, a new work specification is set up from them, with certain fields (the OSI job parameters) copied from the parent work specification.

The complexity of the work specification conceptual datastructure makes it difficult to work with for selection and modification. To aid this task, another (simpler) conceptual datastructure is defined. This is a simple list of the main fields of the work specification, without the structuring information. This simplified conceptual datastructure is called a **header list**, with one "header" for the subjob defined by the work specification, and one for each proforma (nesting ignored) that it contains. The headers for the proformas contain one field (not modifiable) which contains all proforma names leading to the proforma. From this, the nesting structure of proformas can be deduced.

This clause defines the fields of the work specification, the header list, and the transfer control record conceptual datastructure.

3.4.1 Work specifications

<work specification>::=	<OSI job parameters>	(see 3.4.2)
	<subjob name-list>	(see 3.3.2)
	(<subjob specification> | NULL)	
	<proforma list>	
	<local fields>	
	<document list>	

where

<subjob specification>::=	<OSI subjob parameters>	(see 3.4.3)
	<JTM action parameters>	(see 3.4.4)
	<error action>	(see 3.4.3.4)
<proforma list>::=	[<proforma>]*L	(see 3.4.5)
<local fields> ::=	<agency activity parameters>	
	<time waiting = I>	
	<estimated size = I>	
<document list>::=	[<document = D>]*L	

The <OSI job parameters> provide the identification and authorisation for the entire OSI job, and define the reporting which is required.

The <subjob name-list> identifies each of the subjobs of the complete OSI job. It is unambiguous within the scope of the OSI job. Following spawning from the initial subjob, many subjobs can be in existence (and being processed) concurrently.

The Value NULL only replaces a <subjob specification> when the <work specification> is held waiting for a J-END-SIGNAL from a sink or execution agency. A <work specification> is never transferred in this form.

The <OSI subjob parameters> define the open systems which are to do the work, the urgency of the subjob, a possible set of initial holds, and the type of the subjob.

The <JTM action parameters> depend on the type of the subjob, and define in detail the actions of the open systems which are to do the work.

The <error action> controls whether the work specification is held for modification or discarded when errors are detected during its processing or transfer.

The <proforma>s define the further work (if any) which is to be initiated following the processing of the <JTM action parameters>. This further work is defined by further <OSI subjob parameter>s and <JTM action parameters> in the proforma, together with the OSI job parameters for the entire OSI job. Each <proforma> contains an indication of when the subjob it defines is to be initiated and can contain further proformas. The range of activity which can be described by nested proformas is not bounded (nor is it necessarily finite), so that the definition of fields is necessarily recursive.

The <agency activity parameters> are present in the conceptual datastructure only when a sink or execution agency has an activity in progress in relation to it. They identify the activity in the agency for subsequent J-KILL, J-HOLD, J-RELEASE, or J-STATUS primitives. In the Basic Class, there is at most one agency activity associated with a work specification.

The form of <agency activity parameters> is not defined, as they have only local significance. They are used between a J-DISPOSE and a corresponding J-END-SIGNAL to identify activity in an agency associated with the work specification. They are supplied by the agency on a J-DISPOSE response, and used in all subsequent interactions concerning the activity. They do not appear in proformas.

The <time-waiting = I> contains the time in minutes since the work specification entered the state of waiting for processing by the JTM service provider. It can be waiting for transfer, or for the issue of J-GIVE or J-DISPOSE primitives. It is zero if the work specification is not waiting for the attention of the service provider, such as if it is in the hold state, or awaiting a J-END-SIGNAL.

The <estimated size> is zero unless the work specification is awaiting transfer to another open system, when it is the estimated size in kilo octets of the JTM Transfer Element (including any embedded documents) which would be used, using the Basic Encoding Rules for ASN.1 for the Transfer Element and the appropriate minimum implementation required transfer syntax for the embedded documents.

The <document>s (if any) form part or all of the material which is to be passed to sink agencies or to execution agencies during the performance of this OSI job. Each <document> is referred to from the <JTM action parameters> by its position in the list. Its <document type> is also recorded in the <JTM action parameters>.

3.4.2 OSI job parameters

<OSI job parameters>::=	<OSI job submission system = N>	
	<initiating identification>	(see 3.3.2)
	<initiating time>	(see 3.3.2)
	<OSI job name = S>	(see 3.3.2)
	<OSI job local reference = S>	(see 3.3.2)
	<audit trace>	(see below)
	<primary monitoring specification>	(see below)
	<secondary monitoring specifications>	(see below)
	<authorisations>	(see below)
	<permissions>	(see below)
	<accounts>	(see below)
	<security parameters = R>	
	<CCR parameters>	(see below)

The <OSI job submission system> is inserted by the JTM service provider (implementations of JTM are required to be configurable with this value). This open system name equals the first element in the audit trace once a successful transfer has occurred.

The <initiating identification> is provided by the JTM user and can be used in selecting a work specification by a later manipulation, and is present in displays and reports. The form of this identification is defined in 3.3.2. This identification is not subject to authentication.

The <initiating time> is inserted by the JTM service provider to identify the time of the submission.

The <OSI job name> is supplied by the JTM user to identify the job. The <OSI job local reference> is supplied by the JTM service provider, and is unambiguous within the scope of the <OSI job submission system>.

The <audit trace> is used to determine the source of a communication:

<audit trace>::=	[<audit element>]*L

The <audit trace> is initially empty. Whenever the work specification is passed (using a transfer element) to another open system, an element is added to the end of the <audit trace> to identify the sender:

<audit element>::=	<open system name = N>
	(UNKNOWN | KNOWN | AUTHENTICATED)

A sender inserts his own open system name with the literal "UNKNOWN". If the receiver is able to verify this name by use of directory mechanisms and trust in lower layer calling addresses, UNKNOWN is changed to KNOWN. If positive authentication has been performed by the lower layers, then UNKNOWN is changed to AUTHENTICATED.

NOTES

1 The OSI Security Architecture is still under development, but is likely to provide authentication services in the lower layers.

2 The management of public networks normally provide some assurance of the correctness of calling addresses, sufficient for normal commercial work and category KNOWN. On local area networks, such assurance can be missing.

See 3.4.2.1 for the use made of the audit trace in establishing authorisation.

The <primary monitoring specification> is inserted by the JTM service provider. The <secondary monitoring specification>s (if any) are obtained from J-INITIATE.

<primary monitoring specification>::=
<monitoring specification>

<secondary monitoring specifications>::=
[<monitoring specification>]*L

<monitoring specification>::= <monitor specification> (see 3.4.2.3)
<report selector> (see 3.4.2.2)

The <monitor specification> identifies the monitor point, and (optionally) a sink, and (optionally) a relay. The report selector identifies the categories of report to be sent to this monitor point via the relay (if any). (See 3.4.2.2 and 3.4.2.3.)

The <authorisations>, <permissions>, and <accounts> are

<authorisations>::= [<authorisation element>]*C (see 3.4.2.1)

<permissions>::= [<permission element>]*C (see 3.4.2.1)

<accounts>::= [<account element>]*C (see 3.4.2.1)

These elements are supplied by J-INITIATE, but the JTM service provider can, from time to time, add additional elements derived from elements already present (see 3.4.2.1). In the Basic Class the <accounts> is empty.

The <security parameters> determines the use made by JTM of lower layer security mechanisms when transferring the work specification. It also determines the action taken by JTM on repeated denial of service. In the current standard it has only the value "NONE", meaning that no lower layer mechanisms are used to prevent disclosure or to detect tampering or to use only secure routes.

NOTE — Following completion of work on the OSI Security Architecture, this clause is expected to be modified by an addendum.

The <CCR parameters> field is used when a work specification is, at the end of an atomic action, secured and retained by the JTM service provider. In the first atomic action of the OSI job, the CCR parameters on J-BEGIN are determined by the initiation agency. The value of the "diagnostic code indicator" parameters, used by the JTM service provider for subsequent atomic actions, are the same as the value on the J-INITIATE commitment group. The <CCR parameters> field records this value. (Note that it is never explicitly passed in a Transfer Element, being conveyed by C-BEGIN.)

<CCR parameters>::= <diagnostic code indicator = R>

The diagnostic code indicator controls the character set (and hence language) used in CCR (and hence in JTM) diagnostics. Non-standardised text messages (see 3.3.2) are also required to conform to the character sets specified in this parameter.

3.4.2.1 Authorisation and accounting

<authorisation element>::= <identification> (see 3.3.2)
<validation field>

<account element>::= <identification> (see 3.3.2)
<validation field>

<permission element>::= <identification> (see 3.3.2)

<validation field>::= (<audit index = I> | <secret data>)

<secret data>::= (UNSET | <graphic password = S> | <binary password = OS>)

Each <authorisation element> identifies either a user (using an <authority name> and one of the identifications it issued), or the management of an open system or identification authority (using the open system name or authority name).

An open system in general requires an authenticated identification known to it in order to authorise some activity. Such an identification is obtained by use of an authorisation element, possibly in conjunction with the audit trace.

An identification with <secret data> is authenticated (for use by a specific open system at a specific time) if the open system can access the necessary database to check the <secret data>, and if the check succeeds.

An identification with an <audit index = I> validation is authenticated if all <audit elements> from the point specified by the audit index to the end of the audit trace (inclusive) have a sufficient authentication (UNKNOWN, KNOWN, AUTHENTICATED) and are known to and trusted by the open system. This form of validation field is an assertion by the open system referenced by the audit index that the <identification> has been authenticated by it.

NOTE — In the case of the results of processing returning to the OSI job submission system, the open system referenced by the audit index can be the same as the open system currently seeking an authenticated identifier.

The J-INITIATE primitive can only supply authorisation elements with validation fields containing <secret data>. The JTM service provider adds authorisation elements of the <audit index> variety as follows:

a) whenever a <secret data> field is checked; or

b) if, at the time of J-INITIATE, the authenticated <identification> of the initiator is known to the JTM service provider by a local management function; or

c) if tables are available at an open system showing that identifications issued by one authority are equivalent to those issued by another authority.

The <initiating identification> is the claimed identification of the initiator, supplied on J-INITIATE. A JTM implementation is not necessarily able to detect masquerade in this value, using local mechanisms. For reports of attempted security violations, the audit trace is also reported.

The <user> form of <identification> is the most common. The <authority> and <open system> forms are used to identify management activity (for the purposes of authorisation or accounting). The <authority> form of authorisation permits any JTM activity which would be allowed for **any** of the user identifications issued by that authority. The <open system> form of authorisation is needed to manipulate transfer control records, or to display or manipulate report work specifications. It also provides the right to manipulate any work specification which involves that open system.

The <account element> is used (in the same way as the <authorisation element>) to provide an authenticated account for levying charges. In the Basic Class subset, <account element>s are not present; account identifications, if required, are deduced from the identifications provided for authorisation.

Each <permission element> allows other work manipulation subjobs to make changes to the work specification containing the element, provided only that the manipulation subjob carries a validated <authorisation element> for the <identification> given in the <permission element>, or, in the case of a <user>, for its corresponding management.

3.4.2.2 Report selectors

<report selector>::= [<event identification>]*C

<event identification>::= (NORMAL-TERMINATION | MANIPULATION-TERMINATION |
ABNORMAL-TERMINATION | USER-MESSAGE |
CREATION | TRANSFER | SPAWNING | AGENCY-ACCEPTANCE |
MODIFICATION | ERROR-DIAGNOSTIC | NO-PROGRESS |
ACCOUNTING-DATA | NOT-SUPPORTED-TERMINATION |
VIOLATION-ATTEMPT | WARNING-REPORT)

The <report selector> requires the JTM service provider to generate reports for all the categories of event specified in the collection. Reports are never generated on events related to the generation, transfer, processing or disposal of reports.

3.4.2.3 Monitor specifications

<monitor specification>::=	[<relay=N>]*L <monitor system name = N> <disposal instructions>	(see 3.4.3)
<disposal instructions>::=	(KEEP | <disposal data>)	
<disposal data>::=	<se identification> <document se reference>	(see 3.4.4.1.1) (see 3.4.4.1.2)

The subfields of the <disposal data> are identical to subfields of the <JTM action parameters> for document movement work specifications except that the <se identification> must reference a sink. The range of permitted values is defined in 3.4.4.1.

The <monitor system name> is the open system to which reports are to be delivered. If the <disposal instructions> are KEEP, the open system retains the reports (together with any relevant OSI job parameters) until it receives a suitably authorised report manipulation work specification requiring their display or deletion. If the <disposal data> is present, the report is passed to the specified sink agency using the J-DISPOSE primitive group with parameters derived from the <disposal data>.

NOTE — An implementation can discard reports it is retaining after an implementation dependent time.

3.4.3 OSI subjob parameters

<OSI subjob parameters>::=	<target list>	
	<urgency>	(see 3.4.3.1)
	<holds>	(see 3.4.3.2)
	<subjob type>	(see 3.4.3.3)
<target list>::=	<relays>	
	<target = N>	
<relays>::=	[<relay = N>]*L	

The <target> is the open system which is to perform the JTM actions. The <relays> are open systems at which JTM relaying is to occur on the path to the target. A <relay> is removed from the head of this list by the sender during transfer of the work specification, so that the <relays> are empty on arrival at the target.

The <urgency> is used by open systems to assist with determining the order in which different OSI subjobs will be progressed.

The <holds> field is a collection of <hold element>s and specifies whether the conceptual data structure defining the subjob is to be progressed. An OSI job can progress to completion only if there are no <hold element>s preventing its progress. If a <hold element> prevents progress of an atomic action to the point needed for the specified level of commitment, the atomic action is rolled back by the JTM service provider with a NO-RETRY diagnostic. Subsequent manipulation can either remove <hold element>s or add <hold element>s to a work specification or to its proformas.

The <subjob type> defines the nature of the subjob, and the form of the <JTM action parameters>.

3.4.3.1 Urgency

<urgency>::=	(HIGH | MEDIUM | LOW)	

This field provides information from the originator of a work specification or proforma to open systems progressing the work. It can be used as one input to the scheduling of any activity needed to progress the OSI job. Its use is not standardised. In the absence of any other inputs to the scheduling process, a subjob with HIGH urgency is progressed before one with MEDIUM or LOW urgency (and MEDIUM before LOW).

3.4.3.2 Holds

<holds>::=	[<hold element>]*C	
<hold element>::=	<hold location = N>	
	(<reason = S> |	
	<diagnostic information>)	(see 3.3.3)
	<release permission>	
	<release time>	
<release permission>::=	(<identification> | NULL)	(see 3.3.2)
<release time>::=	(<date-time = S> |	(see 3.3.2)
	<time of day = S> | <time interval = I>)	

A <hold element> is ignored by an open system unless the <hold location> is the name of that open system. If the <hold location> in one or more <hold elements> matches the name of the open system, then a work specification will not be processed by the open system (to transfer it, issue service primitives, or spawn from it). It can be transferred to the <hold location>, or created there by spawning or by J-INITIATE.

A <hold element> is either present initially (set by J-INITIATE), or added later by a work manipulation, or automatically inserted if processing is suspended due to errors (see 2.1.6.2 and 3.4.3.4).

If the <hold element> was inserted by the JTM service provider as a result of detecting an error in it, the <diagnostic information> carries details of the error, there is no <reason>, and the <hold location> is the name of the open system inserting the <hold element>.

If the <hold element> was inserted by a manipulation, the <reason> contains human readable text.

The release permission is NULL if the element was inserted by the JTM service provider, otherwise it can contain an <identification> which requires that a manipulation subjob wishing to release the hold has to contain an authorisation element with the same authenticated identification (in addition to satisfying the <permissions> field). If the <release permission> is NULL, the <permissions> in the OSI job parameters control release of the hold.

The <release time> specifies the time at which the JTM service provider automatically removes the hold element. (Note that this causes a MODIFICATION event to occur.) An implementation can place a limit on the length of time that it is prepared to maintain a hold state for a work specification. The <date-time> specifies that the hold element is to be removed at that absolute time. The <time of day> is of the form hhmm and specifies that the hold element is to be removed when that local time first occurs following spawning of the subjob from a proforma or arrival at that open system. The <time interval> is converted to a <date-time> when it is at the <hold location> in a top-level subjob i.e. when the work specification arrives at the <hold location>, or the proforma containing the hold element is spawned, or when a work manipulation adds the hold element to the top-level subjob. The <time interval> specifies the removal of the hold element when <time interval> minutes have passed from the conversion of the <time interval> to <date-time>.

3.4.3.3 Sub-job type

<subjob type>::= (DOCUMENT-MOVEMENT | WORK-MANIPULATION | REPORT-MOVEMENT | TCR-MANIPULATION | REPORT-MANIPULATION)

The value of this field in the <subjob parameters> of J-INITIATE is fixed (for each J-INITIATE primitive) to the value corresponding to the name of the service primitive. It determines the form of the <JTM action parameters>. The value of this parameter in a proforma is not restricted by the J-INITIATE primitive used to specify the OSI job, but can not take the value REPORT-MOVEMENT.

3.4.3.4 Error action

<error action>::= (HOLD <time interval =I> | TERMINATE)

The <error action> controls the action of the JTM service provider at an open system when acting as a commitment master for processing a work specification. The action is described below.

3.4.3.4.1 Sources of error

The master is initiating one or more branches of the atomic action for

a) access to source agencies (which may be using FTAM);

b) access to sink agencies (which may be using FTAM);

c) transfer of a work specification (possibly as the result of spawning).

One or more of these branches can produce rollback from a subordinate with a CCR diagnostic. The subsequent action by the master is determined by the <error action>.

NOTE — Rollback of a J-GIVE by a source agency might or might not produce an error from that branch of the atomic action, depending on the setting of the <embed diagnostics> parameter in the <document source reference> (see 3.4.4.1.3).

3.4.3.4.2 HOLD action

A value of HOLD <time interval> causes the JTM service provider (when an error occurs) to

a) modify the <error action> to TERMINATE; and

b) add a <hold element> for the current location, with <diagnostic information> derived from the CCR diagnostic(s), a NULL <release permission>, and a <release time> calculated from the current time plus the <time interval>; and

c) generate any required reports for the "NO PROGRESS" event.

3.4.3.4.3 TERMINATE action

A value of TERMINATE causes the JTM service provider to

a) discard the work specification; and

b) generate any required reports for the "ABNORMAL-TERMINATION" event.

3.4.4 JTM action parameters

<JTM action parameters>::=	(<document movement operations> |	(see 3.4.4.1)
	<work manipulation operations> |	(see 3.4.4.2)
	<transfer manipulation operations> |	(see 3.4.4.3)
	<report manipulation operations> |	(see 3.4.4.4)
	<report movement operation>)	(see 3.4.4.5)

If the <subjob type> is DOCUMENT-MOVEMENT, this parameter consists of <document movement operations>.

If the <subjob type> is WORK-MANIPULATION, this parameter consists of <work manipulation operations>.

If the <subjob type> is TCR-MANIPULATION, this parameter consists of <transfer manipulation operations>.

If the <subjob type> is REPORT-MANIPULATION, this parameter consists of <report manipulation operations>.

If the <subjob type> is REPORT-MOVEMENT, this parameter consists of <report movement operations>.

3.4.4.1 Document movement operations

<document movement operations>::=	[<document movement>]*L	
<document movement>::=	<document type = ND>	
	<se agencies>	
	<document block>	
<se agencies>::=	[<se identification>]*L	(see 3.4.4.1.1)

NOTES

1 All <se identifications> in the list are <JTM se>, or all are <FTAM se>. Mixed disposal in one document block is not supported.

2 Where the <multiple form> is used, the <se identifications> are the <JTM se> form.

<document block>::=	(<single form> |	(see 3.4.4.1.2)
	<multiple form>)	(see 3.4.4.1.4)

Each <se identification> identifies a sink or execution agency.

Each <single form> generates (by use of one or more J-GIVE primitives) a single document (possibly with embedded diagnostics – see 3.4.4.1.3) for disposal by the target, possibly as the result of concatenating several documents obtained from source agencies. Each <multiple form> generates zero, one or more documents for disposal, each in a separate <single form> <document block>. All documents moved by a single <document movement> are of the same document type. Thus a single <document movement> provides for either

a) movement with concatenation of a number of documents and possible duplication of the result (the use of <single form>); or

b) movement of a set of documents, possibly with duplication (the use of <multiple form>).

The processing by the target of each <document movement> takes all of the documents described by the <document block> and passes each of them to all of the <se agencies>, with separate J-DISPOSE primitive groups for each document.

Where a document (for disposal) is formed by the concatenation of documents from sources, each document has the same document type, and the result of the concatention is required to be a document of the same type.

The <document movement>s are processed by the target in order, all J-DISPOSE indications for one being given before any J-DISPOSE indication for the next. The order of J-DISPOSE indications in the processing of a single <document movement> is not defined. The results of a document movement become visible to other atomic actions only after commitment. They become visible to other parts of the same atomic action at the time of the J-DISPOSE indication.

3.4.4.1.1 Sink or execution agency identification

<se identification>::=	(<JTM se> | <FTAM sink>)	
<JTM se>::=	<se name = S> <additional authorisations> <agency parameter> <agency activity label = S> <se prefix>	
<FTAM sink>::=	<filestore-name = N>	
<agency parameter>::=	(NULL | STORE | <agency format = S>)	
<se prefix>::=	[<name = S>]*L	
<additional authorisations> ::=	(NULL | <password = S>) (NULL | <account element>) (NULL | <authorisation element>)	 (see 3.4.2.1) (see 3.4.2.1)

There are two forms of <se identification>. The first, <JTM se>, is used for both sinks and execution agencies which dispose of the document either locally, or remotely in a way which is not standardised. The second, <FTAM sink>, is used to identify a local sink agency which disposes of a document (locally or remotely) using ISO 8571-3.

The <JTM se> identifies a sink or execution agency at the target site whose name is given by <se name>. <se name> is a string defined by the target open system to unambiguously identify a sink or execution agency (within that open system) to which a J-DISPOSE is to be issued for disposal of a document.

The <additional authorisations> provide an optional additional password, authorisation element, and account element for this access to this sink or execution agency.

The <agency parameter> is passed in J-DISPOSE, and is used (together with the <document type>) by the sink or execution agency to determine the way in which the document is to be presented, stored or interpreted. If the parameter value STORE is supported by a combined sink and source agency, it stores the document in such a way that any subsequent J-GIVE quoting the same <document type> and <agency parameter> together with an appropriate <document source reference> generates the identical document.

NOTE — The value STORE is not applicable to an execution agency.

If the parameter specifies NULL, the document is stored or presented in a way determined locally. If the parameter specifies an <agency format>, the sink or execution agency uses this to determine the way the material is to be stored, presented or interpreted. The possible values of the <agency format> string are not standardised, and are determined by the agency.

The <agency activity label> allows an unambiguous association to be defined between a J-DISPOSE to an execution agency and a later J-GIVE (from a spawned proforma) to that agency. It is only needed when more than one activity is initiated in an agency by a single work-specification.

The <se prefix> names are placed at the head of the list of <document se reference> names (see below) for each document when forming the parameters passed in the J-DISPOSE primitive. This parameter enables the initial part of a <name-list> in a J-DISPOSE to be different for each of the sink or execution agencies in a <document movement>.

The <filestore-name> in the <FTAM sink> form identifies the (local or remote) FTAM virtual filestore to receive the document. All other information for the disposal is contained in the <document se reference> (see 3.4.4.1.2).

3.4.4.1.2 Single form document blocks

<single form>::= <document se reference>
[<document pointer>]*L

<document se reference>::= (<JTM write-data> | <FTAM write-data>)

<JTM write-data>::= <se access parameter>
<name-list>

<FTAM write-data>::= <"write transfer specification" as defined in annex D of ISO 8571-3>

NOTE — The <FTAM write-data> form is used if and only if the corresponding <se identification>s have the <FTAM sink> form.

<se access parameter>::= (NORMAL | NEW | OLD | APPEND | ADD)

<name-list>::= [<name = S>]*L

<document pointer>::= (EMBEDDED |
<single document reference>) (see 3.4.4.1.3)

This form of <document block> specifies that the documents referred to in the <document pointer>s are to be concatenated (in order of the reference in the document block) to form a single document for disposal by J-DISPOSE.

The <se access parameter> specifies the required action of the agency in relation to any existing material of the same name in the sink or execution agency to which they are passed.

NOTE — This parameter is ignored by an execution agency.

It is passed as a field of a parameter of J-DISPOSE. The meaning of the literals is as follows:

NORMAL overwrite any old documents with the same name, or create a new document at the sink;

NEW create a new document at the sink if possible, but if an old document exists with the same name, rollback with a diagnostic;

OLD overwrite an old document of the same name, but if an old document does not exist, rollback with a diagnostic;

APPEND add to the end of any old document, but if an old document does not exist, rollback with a diagnostic;

ADD add to the end of any old document, and if an old document does not exist, create a new one;

The <name-list> has the <se prefix> (see 3.4.4.1.1) added at its head, and then forms the <name-list> passed in the J-DISPOSE to the sink or execution agency to identify the document.

The EMBEDDED literal references a document in the <document list>. The number of documents in the <document list> equals the total number of occurrences of EMBEDDED in the work specification (including proformas). The correspondence is by position in the <document list> and in the transfer syntax of the work specification.

Where <FTAM write-data> is provided, the sink agency uses the ISO 8571-3(FTAM) services to dispose of the document to a local or remote virtual filestore, as defined in ISO 8571-3.

The <single document reference> requires the inclusion of a document to be obtained by a J-GIVE primitive or by use of FTAM. This is processed by the JTM service provider into an EMBEDDED form, or an error diagnostic is included.

3.4.4.1.3 Single document references and source identification

<single document reference>::= <action open system = N>
<source identification>
<document source reference>
<embedded diagnostics>
<state>

<source identification>::= (PROVIDER | <JTM source> | <FTAM source>)

<JTM source>::= <source name = S>
<additional authorisations>
<agency parameter> (see 3.4.4.1.1)

<FTAM source>::= <filestore-name = N>

<document source reference>::= (<JTM read-data> | <FTAM read-data>)

<JTM read-data>::= <source access parameter>
<name-list>

<FTAM read-data>::= <"Read transfer specification" as defined in annex D of ISO 8571-3>

NOTE — The <FTAM read-data> form is used if and only if the <source identification> has the <FTAM source> form.

<source access parameter>::= (MOVE | COPY)

<embedded diagnostics>::= (EMBED | ERROR)

<state>::= (NOT ATTEMPTED | FAILED <error information>)

<error information>::= <time stamp> (see 3.3.2)
<CCR diagnostics> (see 3.3.3)

The <action open system> specifies the name of the open system which is to issue the J-GIVE indication (which can include use of FTAM to obtain the document). The <action open system> is either the open system creating the work specification (by spawning or as a result of J-INITIATE), the target, or one of the relays.

The <source identification> is PROVIDER only when the reference occurs in a proforma which is spawned as a result of a work manipulation, transfer manipulation, or report manipulation. It is used to reference the documents produced by the service provider as the result of a display command.

The <source identification> has the <JTM source> form for sources which obtain a document locally or by non-standard remote access. The <FTAM source> form identifies a local source agency which obtains a document (locally or remotely) using ISO 8571-3.

The <source name> is a string defined by the <action open system> to unambiguously identify a source agency (within that open system) to which a J-GIVE is to be issued to obtain a document.

The <agency parameter> is used to determine how the document was stored. If this field is inconsistent with any directory information held by the source agency, an error diagnostic results. (See also 3.4.4.1.1.)

The <additional authorisations> perform the same function as they do for access to sink or execution agencies.

The <source access parameter> is passed as one of the fields in the J-GIVE primitive; if it is COPY, the source agency is not affected by the activity. If it is MOVE, the document is marked for deletion from the source on commitment, and left unchanged on rollback. If concurrent activities (part of the same atomic action) are accessing this source, it is deleted (if any of them have committed to move) following commitment or rollback by all of them.

The <name-list> is passed on a J-GIVE, and identifies the document which is referenced. It identifies precisely one document; if it does not, the agency to which the J-GIVE was issued rollsback with a diagnostic.

The <filestore-name> in the <FTAM source> form identifies the (local or remote) FTAM virtual filestore which is to provide the document. All other information needed to obtain the document is provided by the <FTAM read-data>.

Note that in resolving a single document reference using J-GIVE, the <document type> of the <document movement> is also supplied.

The <embedded diagnostics> specifies the action to be taken by the <action open system> if J-GIVE is rolled back by the agency with a diagnostic or the <source identification> is PROVIDER and the referenced document is not available.

A value EMBED requires the <state> to be changed from its initial value of NOT ATTEMPTED to FAILED. The <time stamp> records the time of the unsuccessful J-GIVE, and the J-ROLLBACK diagnostic is placed in the <CCR diagnostics> field. Processing of the document reference proceeds to an offer of commitment — the rollback from the agency is not visible to the master. If the main action proceeds to commitment, the change to <state> commits. If it is rolled back (for other reasons) the change to <state> is rolled back. This action generates the DIAGNOSTICS event, which can be selected for reporting.

A value of ERROR leaves the <state> as NOT ATTEMPTED, and propagates the rollback and diagnostic up the atomic action tree to the master. Subsequent action is determined by the value of the <error action> field for the whole subjob. (See 3.4.3.4.)

3.4.4.1.4 Multiple form document blocks

<multiple form>::=	<action open system = N> <document se skeleton> <embedded diagnostics> <JTM source> <additional authorisations> <document source skeleton> <document selector>	(see 3.4.4.1.3)
<document selector>::=	(NULL | <local string = S>)	
<document se skeleton>::=	<se access parameter> <partial name-list>	(see 3.4.4.1.2)
<document source skeleton>::=	<source access parameter> <partial name-list>	(see 3.4.4.1.3)
<partial name-list>::=	[<name =S>]*L	

The fields <action open system>, <JTM source>, <additional authorisations>, <embedded diagnostics>, <se access parameter>, and <source access parameter>, are defined in 3.4.4.1.2 and 3.4.4.1.3.

The multiple form document block is used to generate one or more single form document blocks, each containing an EMBEDEDDED <document pointer>.

The <document se reference> for the <single form>s are constructed from the fields of the <document se skeleton> by adding names (suffices) to the end of the <partial name-list>.

The documents are obtained by the <action open system> from the source agency specified by the <source identification>. They are all of the same <document type> specified for the <document movement>. The <document source reference> for the J-GIVE to obtain each document is constructed from the fields of the <document source skeleton> by adding names (suffices) to the end of the <partial name-list>. The suffices are supplied by J-ENQUIRE.

The multiple form document block is processed by the <action open system> by first issuing J-ENQUIRE to the source agency, passing the <document type>, <document source skeleton>, and <document selector>. This returns a list of suffices (name-lists), one for each document which is to be accessed. Each of these suffices (name-lists) is added to the <partial name-list> in the <document se skeleton> and <document source skeleton> as defined above. The set of suffices (name-lists) returned by J-ENQUIRE is the maximal set for which the resulting J-GIVEs can produce documents, subject to the use of the <document selector> and <document type>. The semantics of the <local string> are not standardised. The value NULL for the <document selector> means maximum selection.

NOTE — FTAM does not currently provide the facilities needed for J-ENQUIRE, so <multiple form> document blocks are restricted to local access.

The source agency can use any or all of the following information to determine which documents are being selected:

a) the <partial name-list>;

b) the <document type>;

c) the <agency parameter>;

d) the <source access parameter>;

e) the <password> from the <additional authorisations>;

f) the <document selector>.

Resolution of this reference is of particular use when collecting documents following processing by an execution agency, where the names and number of the documents might not be known in advance, and can depend on the degree of success of the processing, or on whether it is interrupted by a STOP manipulation.

3.4.4.2 Work manipulation operations

<work manipulation operations>::=	[<work operation>]*L	
<work operation>::=	(<select operation> | <kill operation> | <stop operation> | <work display operation> | <modify operation>)	
<select operation>::=	SELECT <selector>	(see 3.4.4.2.1)
<kill operation>::=	KILL	
<stop operation>::=	STOP	
<work display operation>::=	DISPLAY <doc name = S> (FULL | BRIEF)	
<modify operation>::=	MODIFY <update>	(see 3.4.4.2.4)

The work manipulation operations cause the target open system to display or modify the definition of a work specification for which it is responsible.

The KILL, STOP and MODIFY operations can cause the JTM service provider to issue J-KILL, J-STOP, J-HOLD or J-RELEASE indications to associated agencies, and the DISPLAY operation causes a J-STATUS indication to associated agencies.

The SELECT operation selects zero, one or more work specifications or proformas for subsequent operations. It does this by testing the value of fields in the work specification, using JTM literals to specify the fields being tested. The selected work specifications or proformas are used by subsequent operations (KILL, STOP, MODIFY and DISPLAY). The selection is effective until the next SELECT operation is issued.

The KILL operation causes MANIPULATION-TERMINATION of subjobs, and prevents spawning at the end of the killed activity. It will produce a J-KILL or J-ROLLBACK to any associated agency, and rollback of any transfer activity, and discard of the associated work specification.

The STOP operation also produces a J-STOP or J-ROLLBACK indication to associated agencies, but does not prevent spawning at the end of the activity, nor cause termination of the work specification. The STOP operation causes rollback of any transfer activity, which will be reattemped. Use of STOP is signalled in any MODIFICATION report which is generated.

The MODIFY operation causes changes to be made to the specification of the selected work specification(s) or proforma(s). The <update> causes changes to specified fields; if the <holds> changes, this can result in a J-HOLD or J-RELEASE indication. Changes to the hold-state are signalled in any MODIFICATION report which is generated.

The DISPLAY operation causes a copy of data from the selected work specification(s) or proforma(s), together with information from J-STATUS, to be produced as part of a <work display document> with name <doc name>. A <work display document> is defined in 3.5.2.

A work specification can only be modified by a work manipulation subjob if the subjob contains an <authorisation element> for an authenticated <identification> which is allowed to modify the work specification.

The following authenticated <identification>s are allowed to modify a work specification:

a) any <identification> which appears in the <permissions> of the work specification;

b) any identification authority, one of whose <identification>s appears in the <permissions> of the work specification;

c) any open system whose name appears as a <target> or <relay> for the work specification or its proformas or whose name appears in the <audit trace> of the work specification.

In addition, the removal of a <hold element> or <permission element> can only be performed by a validated <identification> equal to that appearing in the <hold element> (if any) or the <permission element>, or related to it as in b) above. If the <hold element> does not contain a <release permission> this additional restriction does not apply.

Following a DISPLAY <docname> operation, a document of type <work display document> is available for collection by reference to a <source identification> of PROVIDER and a <name-list> containing only <docname>. Two display operations with the same <docname> produce both work displays in the same document. The display can be a full display or a brief display, as defined in 3.5.2.

3.4.4.2.1 Selectors

<selector>::=	<selector form> <header test>
<selector form>::=	(CONTAINS-HEADER | DOES-NOT-CONTAIN-HEADER | FIRST-HEADER-IS | HEADER-IS)
<header test>::=	(<field test> | <NOT clause> | <AND clause> | <OR clause> | SUCCESS | FAIL)
<NOT clause>::=	NOT <header test>
<AND clause>::=	AND <header test><header test>
<OR clause>::=	OR <header test><header test>
<field test>::=	<field name> <selection operator> <selection value>

A <selector> comprises a series of tests applied to a conceptual data structure called the **header list** which contains fields corresponding to the work specification and its proformas.

The fields of this data structure are defined in 3.4.6.

The header list consists of one or more **headers**. Each **header** comprises fields which are global to the whole OSI job together with fields which are specific to an OSI subjob as specified by the work specification or by one of its proformas (at any level of nesting) (see 3.4.6).

A <field test> is applied to a named field in a <header> and produces a value of SUCCESS or FAIL. The values of the <field test>s are combined by AND, OR, and NOT operators to produce a SUCCESS or FAIL value for the <header test>.

A proforma is selected by the form HEADER-IS if and only if the <header test> produces SUCCESS when applied to the header corresponding to the proforma.

The form CONTAINS-HEADER selects work specifications which contain at least one proforma (at any depth) corresponding to a <header> for which the <header test> succeeds. The form DOES-NOT-CONTAIN-HEADER selects work specifications for which the <header test> fails on all <header>s corresponding to proformas in it. The form FIRST-HEADER-IS selects work specifications for which the <header test> succeeds on the header corresponding to the work specification itself.

NOTE — HEADER-IS selects (identifies) a work specification or proforma. In the latter case, it also implicitly identifies the work specification containing the proforma when used in a <transfer control record>. CONTAINS-HEADER, DOES-NOT-CONTAIN-HEADER and FIRST-HEADER-IS always select a complete work specification.

3.4.4.2.2 Selection operators

<selection operator>::= (EQUALS |
LIST-CONTAINS | FIRST-OF-LIST-IS | LAST-OF-LIST-IS |
GT | GE | LT | LE)

The <selection operator>s of GT, GE, LT and LE are only used with fields which are integer items (TOTAL-SIZE and TIME-WAITING). (See 3.4.6.)

The <selection operator> EQUALS can be applied to any type of field. The test succeeds if and only if the <selection value> is equal in type and value to the field (but see 3.4.4.2.3).

The other three <selection operator>s are applied to lists; LIST-CONTAINS can also be applied to collections. The test succeeds if and only if the <selector value> is the same type as an element of the <header> field, and matches any element, the first element, or the last element of the list respectively.

Subclause 3.4.6 defines the operators which can be applied to each field of a header.

3.4.4.2.3 Selection values

The <selection value> field is of the same type as the <header> field corresponding to the <field name> except for the "list" operators, when its type is the same as an element of the <header> field list or collection.

For every syntactic item (basic or structured datatype) in the <header> field, the literal ANY-ITEM can be used in the <selection value> instead of a syntactic item of the appropriate type. This matches any syntactic item of any type.

3.4.4.2.4 Updates

<update>::= <field name>
<update operator>
<update value>

This updates a single field of the selected proforma or work specification according to the <update operator> and <update value>. If a complete work specification was selected, the <update> applies to the work specification itself, corresponding to the first <header> in the <header list>.

3.4.4.2.5 Update operators

<update operator>::= (SET-TO | REMOVE | ADD)

REMOVE and ADD are only applied to collections. SET-TO can be applied to any type of field. The operators applicable to each field of <header> are given in 3.4.6.

3.4.4.2.6 Update values

This is the same type as the <header> field corresponding to <field-name> except for the operators REMOVE and ADD, when it is the same type as an element of the header field. If an element is added to a collection which already contains an element equal to that value, the collection is unchanged.

3.4.4.2.7 Field names

<field name>::= (OSI-JOB-SUBMISSION-SYSTEM | OSI-JOB-NAME | SUB-JOB-TYPE |
INITIATOR | OSI-JOB-LOCAL-REFERENCE |
PRIMARY-MONITOR | SECONDARY-MONITORS |
SECURITY-PARMS | AUTHORISATION-SET | ACCOUNT-SET | PERMISSION-SET |
SUBJOB-NAME-LIST | RELAYS | TARGET |
SE-LISTS | SE-REF-LISTS | SOURCE-LISTS | SOURCE-REF-LISTS |
URGENCY | HOLDS | ERROR-ACTION |
TIME-WAITING | TOTAL-SIZE)

The type of each header field corresponding to these field-names, the meaning of its contents, and the permissible operations is given in 3.4.6.

3.4.4.3 Transfer manipulation operations

<transfer manipulation operation>::=
(<set operation> | <check operation> | <transfer display operation>)

<set operation>::= SET
<set by = N>
<recipient = N>
<control specification> (see 3.4.7)

<check operation>::= CHECK
<checking by = N> (see 3.4.2.1)
<recipient = N>
<control specification> (see 3.4.7)

<transfer display operation>::= DISPLAY
(<destination> | ALL)
<docname = S>

The SET operation sets a <transfer control record> (see 3.4.7) to the values specified. It is normally invoked by the management of a site which knows that it is the frequent recipient of work specifications from some site, and wishes to control such transfers. It does this by setting a transfer control record at the sending site. The manipulation work specification must have an authority for the <identification> of the open system specified in the <recipient> field and for the <identification> of the open system in the <set by> field. The <set by> field is used only to determine the open system to which a subsequent CHECK should be sent. The <set by> field is the name of the open system whose management has determined that a particular transfer path is be controlled. It will normally be identical to the <recipient> field, but can be distinct. The <recipient> is the name of the open system to which transfers are to be controlled. This operation establishes the <control specification> to control transfers to the <recipient>.

The CHECK operation provides data (a <recipient> and <control specification>) in use at the <checking by> open system, and requires the open system to which the CHECK operation is sent to create a transfer control manipulation work specification if the value of the data is no longer appropriate. It guards against the indefinite use of inappropriate transfer control records.

NOTE — It is recommended that open systems should send a CHECK manipulation to the <set by> open system if they have been disconnected from the open systems environment for some time.

The CHECK operation requires an authority for the <identification> of the <checking by> open system.

The transfer DISPLAY operation produces a <transfer display document> (see 3.5.3) containing a copy of the <control specification> and the <set by> for the specified <recipient>. The use of ALL displays the transfer control records for all <recipients> which are not UNCONTROLLED (see 3.4.7).

3.4.4.4 Report manipulation operations

<report manipulation operations>::=
(<delete operation> | <JTM-report-display operation>)

<delete operation>::= DELETE <report selection>

<JTM-report-display operation>::=
DISPLAY <report selection>
<docname = S>

<report selection>::= <submission constraint>
<initiator constraint>
<reference constraint>
<jobname constraint>
<subjob constraint>
<type constraint>
<event constraint>
<initiating-time constraint>

<submission constraint>::= (NULL | <OSI job submission system>) (see 3.4.4.2.7)

<initiator constraint>::=	(NULL | <initiating identification>)	(see 3.4.2.1)
<reference constraint>::=	(NULL | <OSI job local reference = S>)	(see 3.4.6)
<jobname constraint>::=	(NULL | <OSI job name = S>)	(see 3.4.2)
<subjob constraint>::=	(NULL | <subjob name-list>)	(see 3.4.3)
<type constraint>::=	(NULL | <subjob type>)	(see 3.4.3.3)
<event constraint>::=	(NULL | <report selector>)	(see 3.4.2.2)
<initiating-time constraint>::=	(NULL | <time bounds>)	
<time bounds>::=	<starting time> <finishing time>	
<starting time>::=	(NULL | <date-time=S>)	
<finishing time>::=	(NULL | <date-time=S>)	

The <report selection> identifies all reports from subjobs (or spawned subjobs) which have the given

a) <OSI job submission system> (if supplied); and

b) <OSI job local reference> (if supplied); and

c) <initiating identification> (if supplied); and

d) <OSI job name> (if supplied); and

e) <proforma name> and <qualifying integer>s (if supplied); and

f) <subjob type> (if supplied); and

g) an event type which is in the <report selector> (if supplied) ;

h) an <initiating time> between the <starting time> and the <finishing time> (if an <initiating-time constraint> is supplied.)

The DELETE operation deletes these reports, the DISPLAY operation displays them as a JTM-report-display document (see 3.5.1). The DELETE operation requires an authority for an <identification> matching a <permission element> which appeared in the work specification being reported on. The type of a JTM-report-display document is given in 3.5.1.

A report in which the <initiating time> is NULL is included by an <initiating-time constraint> only if at least one of the <starting time> and the <finishing time> is NULL.

3.4.4.5 Report movement operations

<report movement operations>::=	[<report operation>]*C	
<report operation>::=	[<monitor index = I>]*C <report>	(see 3.3.2)

Each <report operation> causes a single report to be processed (on arrival at the target) as specified by the monitor specification identified by the monitor index. A value of zero identifies the primary monitor, and one, two, and so on identifies the successive secondary monitors in the secondary monitoring list of the work specification.

3.4.5 Proformas

<proforma>::=	<proforma name = S> <spawning control data>	(see 3.4.5.2)

	[<demand spawning handle>]*C (<proforma reference> | <proforma specification>) <number spawned = I>	(see 3.4.5.1)
<proforma specification> ::=	<subjob specification> <proforma list>	(see 3.4.1) (see 3.4.1)

In Basic Class, the contents of a <proforma list> in a <proforma specification> are wholly transparent, and are not restricted by Basic Class restrictions.

The fields of a <proforma> are the same as the top-level fields defined in 3.4.1. They define the parameters for the subjob which is to be created when the proforma is spawned.

The <proforma name> identifies the <proforma> uniquely within a containing <proforma list>. A <proforma> containing a <proforma list> is called the parent of that <proforma list>. A <proforma> which is not contained in any other <proforma> is called a top-level proforma.

The <number spawned> contains the count of work specifications spawned from this proforma. As spawning only occurs at the target, this is implicitly zero during transfer. It is used in forming the <subjob name-list> when spawning a new work specification from the proforma.

The <demand spawning handle> strings specify whether the proforma is to be spawned by a J-SPAWN service primitive group. The proforma is spawned if the <demand spawn handle> parameter of the J-SPAWN request primitive is equal to one of the strings in the collection of <demand spawning handle> of the proforma. If the collection of <demand spawning handle> is empty or absent, the proforma cannot be demand-spawned.

3.4.5.1 Proforma references

<proforma reference>::=	<proforma name = S>	(see 3.4.5)

This field is used to reference the <subjob specification> in the named <proforma>. The named <proforma> is required to be in the same <proforma list> as the parent of the referencing <proforma>; it can be the parent. When spawning occurs, top-level proformas in the new work specification have the <proforma reference> replaced by the <subjob specification> in the named <proforma>. (Note that this can include the proforma containing the <proforma reference>.)

3.4.5.2 Spawning control data

<spawning control data>::=	(DEMAND-ONLY | ACCEPTANCE | COMPLETION | CONDITIONAL)

This field specifies when spawning is to take place from the <proforma>.

If it specifies DEMAND-ONLY, spawning only occurs on the issue of a J-SPAWN service primitive group from activity in an execution agency produced by the parent subjob.

If it specifies ACCEPTANCE, spawning occurs when all the sink or execution agencies of the parent subjob have given either ACCEPTANCE or COMPLETION commitment.

If it specifies COMPLETION, the subjob is created only when all the sink or execution agencies of the parent subjob have either given COMPLETION commitment, or have made a J-END-SIGNAL request.

If it specifies CONDITIONAL, the subjob is created as for COMPLETION, but only if the source reference resolution following spawning collects at least one document from an execution or source agency at that open system.

NOTE — This facility is provided to support the case where processing might produce several outputs for different destinations but might also produce a lesser number. The use of CONDITIONAL ensures that proformas are only spawned for those destinations for which output has actually been produced.

All types of spawning can occur either as part of the initial atomic action, or as part of a later atomic action.

The commitment level for demand spawning is specified in the J-SPAWN request. For all other spawning, PROVIDER-ACCEPTANCE level is used if the spawning is not part of the initial atomic action.

When several <proforma>s are to be spawned as the result of a single event (e.g. a J-END-SIGNAL request), they are spawned in the order of their appearance in the <proforma list>.

Any J-GIVE primitives to be issued at the local site to the agency producing spawning proceed to an offer of commitment before the next proforma is spawned. (This ensures that MOVE access by some proformas to collect named documents, and CONDITIONAL spawning of a later proforma to collect all remaining documents, proceeds correctly.)

3.4.6 Header lists

<header list>::=	[<header>]*L	
<header>::=	<OSI job submission system = N>	
	<initiating identification>	(see 3.3.2)
	<OSI job name = S>	(see 3.3.2)
	<OSI job local reference = S>	(see 3.3.2)
	<*audit trace*>	(see 3.4.2)
	<primary monitoring specification>	(see 3.4.2)
	<secondary monitoring specifications>	(see 3.4.2)
	<authorisations>	(see 3.4.2)
	<permissions>	(see 3.4.2)
	<accounts>	(see 3.4.2)
	<security parameters = R>	(see 3.4.2)
	<subjob name-list>	(see 3.4.3)
	<relays>	(see 3.4.3)
	<target = N>	(see 3.4.3)
	<urgency>	(see 3.4.3.1)
	<holds>	(see 3.4.3.2)
	<subjob type>	(see 3.4.3.3)
	<se lists>	
	<se-ref lists>	
	<source-lists>	
	<source-ref lists>	
	<error action>	(see 3.4.3.4)
	<time waiting = I>	(see 3.4.1)
	<estimated size = I>	(see 3.4.1)
<*se lists*>::=	[<*se identification*>]*L	(see 3.4.4.1.1)
<se-ref lists>::=	[<document se reference>]*L	(see 3.4.4.1.2)
<source-lists>::=	[<source identification>]*L	(see 3.4.4.1.3)
<source-ref lists>::=	[<document source reference>]*L	(see 3.4.4.1.3)

These fields correspond to fields in a work specification or proforma. Changing the fields in <header list> changes the corresponding work specification or proforma.

The first <header> contains fields corresponding to the <OSI job parameters> and the <subjob specification> in the work specification itself. Subsequent headers contain fields corresponding to the <OSI job parameters> and to the <subjob specification> of a proforma, with one header for each proforma (at any depth).

NOTE — The order of subsequent headers is not specified, and does not affect the results of any operation.

The <password> in a <document se reference>, and the <secret data> in the <validation field> in an <authorisation element> or <account element> always appear as the item NULL in a <header>.

The semantics of the fields of <header> and the operations on them are defined below. Those marked with an * have the same value in all the <header>s of a particular <header list>, that is, they relate to the whole OSI job. These fields can be modified by selecting any of the headers in the header list, the changed values appearing in all headers.

*<OSI job submission system>
This is defined in 3.3.2. Only EQUALS can be applied.

*<initiating identifications>
This is defined in 3.3.2. Only EQUALS can be applied.

*<OSI job name> This is defined in 3.3.2. Only EQUALS can be applied.

*<OSI job local reference>
This is defined in 3.3.2. Only EQUALS can be applied.

*<audit trace> This field is defined in 3.4.2. Operations are EQUALS, LIST-CONTAINS, FIRST-OF-LIST-IS, LAST-OF-LIST-IS.

*<primary monitoring specification>
This is defined in 3.4.2. Operations are EQUALS and SET-TO. SET-TO requires an authorisation for the <identification> of the OSI job submission system.

*<secondary monitoring specifications>
These are defined in 3.4.2. Operations are EQUALS, LIST-CONTAINS, SET-TO, REMOVE, and ADD. In the case of report work specifications, the only permitted operations are EQUALS and LIST-CONTAINS.

*<authorisations> This is defined in 3.4.2. (Note that <secret data> in the <validation field>s all appear as NULL in <header>s.) Operations are EQUALS, LIST-CONTAINS and ADD. (A <validation> containing a checked index can not appear in an ADD operation.)

*<permissions> This is defined in 3.4.2. Operations are EQUALS, LIST-CONTAINS, ADD and REMOVE. REMOVE requires an authorisation for the identity being removed.

*<accounts> This is defined in 3.4.2. (Note that <secret data> in the <validation field>s all appear as NULL in <header>s.) Operations are EQUALS, LIST-CONTAINS and ADD. (A <validation> containing a checked index can not appear in an ADD operation.)

*<security parameters> This is defined in 3.4.2. Only EQUALS can be applied.

<subjob name-list> This list is defined in 3.4.3 for the first header. In the header for a proforma, this field has a series of subjob name components, one for each proforma name used to reach it. The first component is a J-INITIATE request top-level proforma name, the last is the name of this proforma. The value of the <qualifying integer> is zero if the subjob specification is still in the form of a proforma. Only EQUALS, and FIRST-OF-LIST-IS can be applied.

<relays> This field is defined in 3.4.3. Operations are EQUALS, LIST-CONTAINS, FIRST-OF-LIST-IS, LAST-OF-LIST-IS, SET-TO.

<target> This is defined in 3.4.3. Operations are EQUALS and SET-TO.

<urgency> This is defined in 3.4.3.1. Operations are EQUALS, SET-TO. SET-TO requires an authenticated <identification> equal to the OSI job submission system.

<holds> This is defined in 3.4.3.2. Operations are EQUALS, LIST-CONTAINS, SET-TO, REMOVE, ADD. REMOVE requires an authority for the <identification> appearing in the <hold element>, or for its <authority>, or for an <open system> named in one of the <target>s or <relays>.

<subjob type> This is defined in 3.4.3.3. The only operation is EQUALS.

<se lists> This list contains all the <se identification>s appearing in the subjob specification, in the order of their appearance. Any change is required to retain the same number of <se identification>s. Operations are EQUALS, LIST-CONTAINS, SET-TO.

<se-ref lists> This list contains all the <document se reference>s appearing in the <document block>s in the subjob specification, in the order of their appearance. Any change is required to retain the same number of <document se reference>s. Note that the <password> field always appears as NULL. Operations are EQUALS, LIST-CONTAINS, SET-TO.

<source lists>	This list contains all the <source identification>s appearing in the subjob specification, in the order of their appearance. Any change is required to retain the same number of <source identification>s. Operations are EQUALS, LIST-CONTAINS, SET-TO.
<source-ref lists>	This list contains all the <document source reference>s and <document source skeleton>s appearing in the <document block>s in the subjob specification, in the order of their appearance. Any change is required to retain the same number of elements in the list. Note that the <password> field always appears as NULL. Operations are EQUALS, LIST-CONTAINS, SET-TO.
<error action>	This is defined in 3.4.3.4. Operations are EQUALS and SET-TO.
*<time waiting>	This is defined in 3.4.1. Only EQUALS, GT, GE, LT, LE are allowed.
*<total size>	This is defined in 3.4.1. It is zero unless the work specification is waiting for transfer. Only EQUALS, GT, GE, LT, LE are allowed.

3.4.7 Transfer control records

An open system contains zero one or more transfer control records which control the way in which that open system transfers work specifications to other open systems.

Each transfer control record at an open system is set to control the transfer of work specifications to a specified open system. It enables the receiver of work specifications to influence concurrency and priority for the transfer of work specifications by the sender.

<transfer control record>::=	<set by =N> <time stamp> <recipient =N> <control specification>	(see 3.3.2)
<control specification>::=	(UNCONTROLLED | <selection>)	
<selection>::=	[<selector>]*L	(see 3.4.4.2.1)

Initially, all transfer control records held by an open system are set to UNCONTROLLED. In this case the open system will attempt transfers with a concurrency determined locally. It can adopt any algorithm for selecting work specifications for transfer. (Algorithms will normally use some combination of oldest first, smallest first, and highest <urgency> first).

If the value is UNCONTROLLED, <set by> and <time stamp> is not recorded. The <time stamp> is the time of the SET operation.

If the value is not UNCONTROLLED, then the <selection> is used not only to determine those work specifications which can cause transfers to be attempted, but also to control the degree of concurrency. If the <selection> list is empty, only transfer manipulation work specifications cause transfers. These are always permitted, but they are not allowed to cause the concurrency established by the <selection> to be exceeded by more than one.

An attempt to transfer a work specification to a <recipient> is not attempted unless each of work specification (if any) already being transferred to that recipient, and the work specification to be transferred are each selected by a different <selector> in the <selection> list.

NOTE — This means that each selector permits the transfer of at most one work specification at any time. Several identical <selector>s can occur in the <selection>. Where more than one work specification satisfies a <selector>, the choice of work specification for transfer is a local implementation matter. (See also 3.4.3.1.)

3.5 JTM documents

This clause defines the semantics of the documents carrying JTM report displays, JTM work displays, and JTM-TCR-displays. A transfer syntax for these documents is specified in the JTM protocol specification.

The documents can form part of a work specification and parameters of J-GIVE and J-DISPOSE.

3.5.1 JTM report-display document

<JTM-report-display document>::=	[<JTM-report-display>]*L	
<JTM-report-display>::=	<OSI job monitor system = N> <time stamp> [<report>]*L	 (see 3.3.2) (see 3.3.2)

When produced by JTM, these documents contain only a single <JTM-report-display>. Concatenation of these documents produces a list of JTM-report-displays.

When such documents are produced as the result of a report manipulation, the selected reports are listed in the order they were received by the OSI job monitor system, to provide the <report> list for the <JTM-report-display>. The <time stamp> gives the date and time the display was produced.

When such documents are produced by a monitor point for use of J-DISPOSE to dispose of a <report> in accordance with the <disposal data>, the <time stamp> is the time the document was constructed by the monitor point, and there is a single <JTM-report-display> containing all the <report>s that were present in the REPORT-MOVEMENT subjob.

This document type is named by an ASN.1 ObjectDescriptor value (see ISO/IEC 8824) of

"JTM report-display document"

and by an ASN.1 OBJECT IDENTIFIER value which is specified in ISO/IEC 8832.

3.5.2 JTM work-display document

<work display document>::=	[<work display>]*L	
<work display>::=	<displaying system = N> <time stamp> [<ws display>]*L	 (see 3.3.2)
<ws display>::=	(<full display> | <brief display>)	
<full display>::=	<header list> [<destination status>]*C	(see 3.4.6)
<destination status>::=	<destination> <status> <time stamp> [<text = S>]*L <CCR diagnostics>	 (see 3.3.2)
<destination>::=	(<recipient = N> | <source agency> | <se agency>)	
<source agency>::=	(<source name = S> | <filestore-name =N>)	(see 3.4.4.1.3)
<se agency>::=	(<se name = S> | <filestore-name =N>)	(see 3.4.4.1.1)
<status>::=	(IN-PROGRESS | ACCEPTED | WAITING)	
<brief display>::=	<subjob identification> [<destination status>]*C	(see 3.3.2)

The <time stamp> in the <work display> gives the time the display was produced.

The <header list> in the <full display> contains the <header>s for a selected subjob, and for all the proformas it contains.

The <destination status> provides information about any agencies or transfers associated with the work specification. The <time stamp> in the <destination status> gives the time that the state was entered.

The <CCR diagnostics> are those of the most recently received retry-later result and are absent unless the <status> is WAITING.

The <filestore-name> form is used for agencies which are obtaining or disposing of documents using FTAM. The <source name> and <se name> forms are used otherwise.

IN-PROGRESS means that there is an atomic action currently in progress. The JTM service provider generates the <text>, which contains human-readable, non-standardised information.

ACCEPTED means that acceptance commitment has been given by an agency. The text list is obtained by the J-STATUS primitive, which returns human-readable, non-standardised information.

WAITING means that resources are not available, a retry-later result has been received, or that a transfer control record prevents progress. The JTM service provider generates the text list, which contains human-readable, non-standardised information.

Concatenation of these documents produces a new list of work displays.

This document type is named by an ASN.1 ObjectDescriptor value (see ISO/IEC 8824) of

"JTM work-display document"

and by an ASN.1 OBJECT IDENTIFIER value which is specified in ISO/IEC 8832.

3.5.3 JTM TCR-display document

<transfer display document>::=	[<transfer display>]*L	
<transfer display>::=	<displaying system = N>	
	<time stamp>	(see 3.3.2)
	[<transfer control record>]*L	(see 3.4.7)

The <time stamp> is the time that the display was produced.

Concatenation of these documents produces a new list of <transfer display>s.

This document type is named by an ASN.1 ObjectDescriptor value (see ISO/IEC 8824) of

"JTM tcr-display document"

and by an ASN.1 OBJECT IDENTIFIER value which is specified in the extended JTM protocol specification.

3.6 Parameters of JTM service primitives

3.6.1 J-INITIATE request and confirm primitives

The J-INITIATE request carries the following fields of a work specification:

<initiating identification>	(see 3.3.2)
<OSI jobname>	(see 3.3.2)
[<secondary monitoring specification>]*L	(see 3.4.2)
<authorisations>	(see 3.4.2)
<permissions>	(see 3.4.2)
<accounts>	(see 3.4.2)
<security parameters>	(see 3.4.2)
<subjob specification>	(see 3.4.1)

<proforma list>	(see 3.4.1)
<document list>	(see 3.4.1)

This completely defines the work to be performed.

The J-INITIATE confirm carries only the single parameter

<OSI job local reference = S>

This string is generated at the OSI job submission system to unambiguously identify this use of J-INITIATE within that open system. A local representation of the date and time of the issue of the primitive would be suitable, but any other unambiguous string can be used.

The commitment level and diagnostic code indicator CCR parameters (carried by J-BEGIN) are chosen by the issuer of the J-INITIATE request.

CCR diagnostics and accounting information can be returned to the user on the other CCR-related parameters.

3.6.2 J-DISPOSE service primitives

3.6.2.1 J-DISPOSE indication primitive

The J-DISPOSE indication is issued by the <target> of a work specification to a sink or execution agency identified by <se name = S> for the <JTM se> form and to the local sink agency performing FTAM access for the <FTAM sink> form.

The J-DISPOSE indication carries the following parameters:

	<provider activity id = S>	
	<user authority>	
	<user account>	
	<additional authorisations>	(see 3.4.4.1.1)
	(<filestore-name = N> |	
	<agency parameter>)	(see 3.4.4.1.1)
	<document se reference>	(see 3.4.4.1.2)
	<errors>	
	<disposal document>	
<errors>::=	[<diagnostic information>]*L	(see 3.3.3)
<user authority>::=	(NOT-KNOWN |	
	<identification> |	(see 3.3.2)
	<authorisation element>)	(see 3.4.2.1)
<user account>::=	(NOT-KNOWN |	
	<identification> |	(see 3.3.2)
	<account element>)	(see 3.4.2.1)
<disposal document>::=	<document type = ND>	
	<document = D>	

The <provider activity id> represents the local information needed to identify the activity for future primitive groups issued by the agency (such as J-END-SIGNAL, J-MESSAGE, etc). Its form is not standardised.

NOTE — Where the agency and the provider are part of the same open system, the <provider activity id> is not visible in the OSI environment.

The <user authority> and <user account> are obtained from an <authorisation element> and an <account element>. They are validated by the JTM service provider if only <identification> is provided, otherwise the complete <authorisation element> or <account element> is supplied. The choice of which <authorisation element> to use for a J-DISPOSE is performed by a local directory function, using the sink or execution agency name and the user identification authority name in the <authorisation element>. The value NOT-KNOWN indicates that no suitable authorisation element could be found, and would often be a reason for rollback by the agency. In some cases, however, the <additional authorisations>, or data in the document itself, can suffice to allow the activity to take place.

One J-DISPOSE primitive group is issued for each <se identification> in the <se agencies> field for each document produced by a <document block>. The <filestore-name> or <agency parameter> are obtained from the <se identification>.

A J-DISPOSE primitive group is also issued to the sink agency specified in the <disposal data> (if any) for an OSI job monitor for each report which reaches the OSI job monitor.

The <document se reference> is obtained from the <se prefix>, <document block> or <disposal data>.

The <errors> consist of a list of <diagnostics information> formed from the <source identification> and <error information> for any <single document reference> with <state> FAILED (see 3.4.4.1.3). The <errors> (if any) are supplied to the sink agency together with the <disposal document>. The disposal of the <errors> is not standardised.

The <disposal document> contains either a <JTM-report-display> or a document of any other (standard or non-standard) document type supported by the implementation.

NOTE — Implementations are required to state the document types they support (for J-GIVE, J-DISPOSE, and transfer). Implementations claiming to be general-purpose are expected to support at least the first three document types defined in annex C.

Disposal of reports to sink agencies is done by J-DISPOSE after converting the <report> into a <JTM-report-display>.

3.6.2.2 J-DISPOSE response primitive

The J-DISPOSE response carries the single parameter

<agency activity id = S>

This represents the local information needed to identify the activity for future primitive groups issued by the JTM service provider (J-KILL, J-STATUS, etc). Its form is not standardised.

3.6.3 *J-GIVE service primitives*

3.6.3.1 *J-GIVE indication primitive*

The J-GIVE indication is issued to a source or execution agency as a result of a <single document reference>, or as the result of a <document source skeleton> and a <name-list> following a J-ENQUIRE. Refer to 3.4.4.1.3, 3.4.4.1.4 and 3.6.4 for details of these fields.

The J-GIVE indication is issued by the

<action open system = N>

to the source or execution agency identified by <source name = S> for the <JTM se> form and to the local source agency performing FTAM access for the <FTAM source> form.

The J-GIVE indication carries the following parameters

<user authority>	(see 3.6.2)
<user account>	(see 3.6.2)
<additional authorisations>	(see 3.4.4.1.1)
<document source reference>	(see 3.4.4.1.3)
(<filestore-name = N> | <agency parameter = S>)	(see 3.4.4.1.3)
<document type = ND>	
<document group>	

where

<document group>::=	(PERMANENT | <activity end group> | <proforma group>)	
<activity end group>::=	<agency activity id>	(see 3.6.2)
<proforma group>::=	<agency activity id>	(see 3.6.2)

<demand spawn handle = S>

All parameters and their meaning have been defined in earlier clauses except <document group>. This is PERMANENT unless the J-GIVE is issued to an execution agency, when it is <activity end group> or <proforma group>. In this case it indicates that the J-GIVE indication is for documents available at the end of an activity or from spawning any proformas that have a <document spawning handle> that is equal to the <demand spawn handle>, respectively.

Where the <source name> is an execution agency, the activity to which the J-GIVE relates is identified by the <agency activity id>. The set of documents being accessed is those available at the end of an activity for the <activity end group>, and those available when J-SPAWN was issued for the <proforma group>. In the latter case, the <demand spawn handle> is that which appeared in the J-SPAWN request.

3.6.3.2 J-GIVE response primitive

The J-GIVE response carries a single <document> parameter. The <document>is of the <document type> specified in the J-GIVE indication. Note that if no suitable document can be found, a J-ROLLBACK request is issued instead of a J-GIVE response, and the C-service diagnostic is then used by the JTM service provider for a JTM diagnostic.

An agency which is both a source and a sink permits immediate J-GIVE access to a document written by a J-DISPOSE indication, provided the same atomic action identifier is used.

The agency permits several J-GIVE accesses by the same atomic action, even if MOVE is specified by the last access. The source document is deleted when and only when all J-GIVE primitive groups have committed or backed-up, with at least one committing to MOVE.

Concurrent activity with a different atomic action identifier follows the normal rules of CCR.

3.6.4 J-ENQUIRE indication and response primitives

The J-ENQUIRE indication is issued to a source or execution agency as a result of a <multiple form> document block in the OSI subjob being processed.

The source or execution agency to which it is issued is obtained as for the J-GIVE indication. It carries the following parameters:

<user authority>	(see 3.6.2)
<user account>	(see 3.6.2)
<additional authorisations>	(see 3.4.4.1.1)
<agency parameter = S>	(see 3.4.4.1.1)
<document source skeleton>	(see 3.4.4.1.4)
<document selector = S>	(see 3.4.4.1.4)
<document type = ND>	
<document group>	(see 3.6.3.1)

The J-ENQUIRE response carries only

[<name-list>]*L

The J-ENQUIRE is used to obtain the names of all documents whose <name-list>s begin with the <partial name-list> in the <document source skeleton>, and which are accessible by a J-GIVE using the quoted parameters (see 3.4.4.1.4).

The returned <name-list>s provide the suffices to be added to the <partial name-list> for the issue of J-GIVE primitives.

If no document can be found, a J-ROLLBACK request is issued instead of a J-ENQUIRE response, and the C-service diagnostic is then used by the JTM service provider for a JTM diagnostic.

Note that it is not required that the J-ENQUIRE response should remain valid after the corresponding J-COMMIT indication. The JTM service provider issues all necessary J-GIVE groups (with the same atomic action identifier) prior to committing to J-ENQUIRE.

3.6.5 J-MESSAGE request primitive

The J-MESSAGE request carries the parameters:

<provider activity id>
<message>

where

<message>::= [<text=S>]*L

The <message> is used by the JTM service provider to form the text list of a report which is delivered to an OSI job monitor if reporting of the USER-MESSAGE event type is selected. (If reporting of this event type is not selected for any monitor point, the <message> is discarded.)

The <commitment level> parameter is set to PROVIDER-ACCEPTANCE, and the atomic action always proceeds to commitment. The diagnostic code indicator is identical to that on the corresponding J-DISPOSE, and the <message> conforms to the character sets required by the diagnostic code indicator. Each line of <text> in the <message> is restricted to a length of 40 characters.

3.6.6 J-SPAWN request primitive

The J-SPAWN request is used to cause the creation of new work specifications from one or more of the top-level proformas in the work specification associated with an activity. The parameters are

<provider activity id>
<demand spawn handle>

Any top-level proforma with a <demand spawning handle> that is equal to the <demand spawn handle> of the J-SPAWN request is spawned.

Following the corresponding J-READY indication, any documents to be used in spawning can be deleted (collection by J-GIVE is complete).

The <commitment level> parameter can take any defined value. The atomic action can be rolled back by a J-ROLLBACK.

3.6.7 J-END-SIGNAL request primitive

The J-END-SIGNAL request implicitly refers to an earlier J-DISPOSE which gave ACCEPTANCE commitment. It contains only:

<provider activity id>

Where there is only one activity involved, it causes all top-level proformas marked for COMPLETION or CONDITIONAL spawning to be spawned, in the order of their appearance in the <proforma list>. See also 3.4.5.2. Where more than one activity is involved, the spawning commits only when all J-END-SIGNAL primitive groups commit.

The commitment level is 1, and the atomic action always proceeds to commitment.

3.6.8 J-STATUS indication and response primitives

The J-STATUS indication is issued by the JTM service provider to a sink or execution agency which has given ACCEPTANCE commitment to a J-DISPOSE, but has not yet issued a J-END-SIGNAL request. It carries only the <agency activity id> parameter. The response carries the single parameter

<status message>

where

<status message>::= [<text = S>]*L

The indication is issued by the JTM service provider as a result of a work manipulation display operation on the work specification associated with the activity. The <status message> is carried in the display. Its form is not defined, but it should give as much information as possible about the state of the activity in the agency. Where the agency wants the report to carry the <agency activity id>, it includes it in the <status message>.

This atomic action always proceeds to commitment. The character set in the status message conforms to the requirements of the diagnostic code indicator. Each line of <text> is limited in length to 40 characters.

3.6.9 J-HOLD, J-RELEASE, J-KILL, J-STOP indication primitives

These primitives carry only the <agency activity id> parameter.

The J-HOLD indication informs the sink or execution agency that the JTM service provider wishes the activity to be suspended. No J-MESSAGE, J-SPAWN or J-END-SIGNAL request is issued until a J-RELEASE indication is received.

The J-RELEASE indication allows J-MESSAGE, J-SPAWN and J-END-SIGNAL requests, and is used to cancel a J-HOLD.

The J-KILL indication tells the sink or execution agency to terminate the activity produced by a J-DISPOSE, and to generate a J-END-SIGNAL request. The J-END-SIGNAL cannot provide any documents. (The JTM service provider does not issue J-ENQUIRE or J-GIVE groups.)

The J-STOP indication has the same effect as J-KILL, but documents are allowed on the resulting J-END-SIGNAL. (The JTM service provider issues J-ENQUIRE and J-GIVE groups.)

A subordinate does not issue a J-ROLLBACK for these atomic actions, but the superior may require rollback.

Table 2 — Primitives potentially used in the processing of a singlesubjob

Document-movement subjob	
J-ENQUIRE	if a multiple form document block is present
J-GIVE	if a single document reference is present, or following a J-ENQUIRE
J-DISPOSE	this either completes, or allows J-END-SIGNAL, J-SPAWN and J-MESSAGE
Report-movement subjob	
J-DISPOSE	
Work-manipulation subjob	
J-KILL	
J-STOP	
J-HOLD	
J-RELEASE	
J-STATUS	
TCR-manipulation subjob	
None	
Report-manipulation subjob	
None	

3.6.10 Summary

A summary of the JTM service primitives appears as table 3. An indication of their use appears in table 2.

A J-INITIATE primitive group can cause further service primitives to be issued as a direct result of processing the first subjob. If the first subjob spawns further subjobs, either by J-SPAWN or by J-END-SIGNAL or on completion of a J-DISPOSE or following a manipulation, then further primitive groups can occur. The primitive groups caused by the original J-INITIATE primitive groups can occur concurrently with that primitive group, or can occur after it, depending on the level of commitment. As a proforma can consist of any type of subjob (except REPORT-MOVEMENT), any type of J-INITIATE primitive group can give rise to all forms of service primitive, depending on its parameters. In addition, reporting of events, including reporting caused by a J-MESSAGE group, can produce a J-DISPOSE. Table 2 gives details of primitives potentially used in processing a single subjob.

Table 3 — Summary of JTM Service Primitives and Parameters

Primitives	Request	Indication	Response	Confirm
J-INITIATE-WORK				
document movement specification	x			-
OSI job local reference	-			x
J-INITIATE-WORK-MAN				
work manipulation specification	x			-
OSI job local reference	-			x
J-INITIATE-TCR-MAN				
transfer manipulation specification	x			-
OSI job local reference	-			x
J-INITIATE-REPORT-MAN				
report manipulation specification	x			-
OSI job local reference	-			x
J-DISPOSE				
provider activity id		x	-	
user authority		x	-	
user account		x	-	
agency parameter | filestore-name		x	-	
document se reference		x	-	
additional authorisations		x	-	
errors		x	-	
document and document type		x	-	
agency activity id		-	x	
J-GIVE				
user authority		x	-	
user account		x	-	
agency parameter | filestore-name		x	-	
document source reference		x	-	
additional authorisations		x	-	
document type		x	-	
document group		x	-	
document and document type		-	x	
J-ENQUIRE				
user authority		x	-	
user account		x	-	
agency parameter		x	-	
document source skeleton		x	-	
additional authorisations		x	-	
document type		x	-	
document selector		x	-	
document group		x	-	
list of name lists		-	x	
J-SPAWN				
provider activity id	x			
proforma name	x			

Primitives	Request	Indication	Response	Confirm
J-MESSAGE				
provider activity id	x			
message		x		
J-END-SIGNAL				
provider activity id	x			
J-STATUS				
agency activity id		x	-	
status message		-	x	
J-HOLD				
agency activity id		x		
J-RELEASE				
agency activity id		x		
J-KILL				
agency activity id		x		
J-STOP				
agency activity id		x		

Section 4 : Basic Class

4.1 Primitive groups and document types for Basic Class

4.1.1 Service primitive groups

The following groups of primitives are available:

J-INITIATE-WORK
J-INITIATE-WORK-MAN
J-DISPOSE
J-GIVE
J-END-SIGNAL
J-STATUS
J-KILL
J-STOP
J-MESSAGE

A Basic Class implementation can support any combination of the following sets of service primitives:

a) J-INITIATE-WORK

b) J-INITIATE-WORK-MAN

c) J-DISPOSE (to a sink or execution agency), with completion commitment on the J-READY

d) J-GIVE (for the activity end group)

e) J-DISPOSE (to a sink or execution agency), with agency acceptance commitment on the J-READY
J-END-SIGNAL
J-STATUS
J-MESSAGE
J-KILL
J-STOP

A Basic Class system does not support the value COMPLETION for the commitment level on a C-BEGIN or J-BEGIN. It may, however, return such a value or accept such a value on a C-READY or a J-READY.

4.1.2 Document types

Basic Class systems do not necessarily support a full range of document types. ISO/IEC 8832 defines three document types named by ASN.1 ObjectDescriptor values (see ISO/IEC 8824) of

"Simple ISO text document"
"Simple ISO print document"
"Simple ISO binary document"

For brevity, these are referred to in this section only as text (1), print (2), and binary (3) respectively.

In addition, the following document types (see 3.5) are used in Basic Class:

"JTM work-display document"
"JTM report-display document"

For brevity, these are referred to in this section only as work-display (4) and report-display (5) (but see also 4.2.1).

The following document types are supported by Basic Class implementations claiming to be general purpose:

in J-INITIATE-WORK
text (1)
print (2)

in the proforma in J-INITIATE-WORK
print (2)
in the proforma in J-INITIATE-WORK-MAN
work-display (4) (Brief only)

in J-GIVE
print (2)
in J-DISPOSE
text (1)
print (2)
report-display (5)
work-display (4) (Brief only)

Note that other document types are not necessarily supported by a JTM Basic Class implementation.

4.2 Conceptual datastructures for Basic Class

This clause defines the subset of the datastructures (defined in clauses 3.3, 3.4, and 3.5) which are used in the Basic Class. Clause 4.3 defines additional Basic Class restrictions related to the parameters of service primitives.

NOTE — Where the parameters of the Basic Class J-INITIATE request (defined in clause 4.3) are more restricted than the corresponding fields of the Basic Class conceptual datastructure defined in this clause, the general case defined here only occurs when a Basic Class implementation is interworking with an implementation posessing functionality exceeding that required for Basic Class.

4.2.1 Reports

Basic Class reports are generated by Basic Class implementations. A Basic Class monitor point can only handle Basic Class reports.

These reports are a subset of the reports defined in 3.3.2, as follows:

<report>::= <name of reporter>
<time stamp>
<subjob identification>
<event identification>
<event parameters>

<subjob identification> has the full range of values specified in 3.3.2.

<event identification>::= (NORMAL-TERMINATION | MANIPULATION-TERMINATION |
ABNORMAL-TERMINATION | USER-MESSAGE)

Other events are not recognised in Basic Class. Reports of other events are not handled by Basic Class Monitor Points. Requests (in a <report selector>) for reporting of other events cause a work specification to be refused by a Basic Class system.

The only <event parameters> supported are <number spawned>, <text>, and <diagnostic>.

4.2.1.1 Diagnostic information

The <diagnostic information> in reports received by Basic Class monitor points can have the full form defined in 3.3.3 with the following exceptions.

The <source details> and <se details> fields do not refer to FTAM activity, and the <CCR diagnostics> do not contain an <FTAM outcome>.

4.2.1.2 Accounting information

Accounting information is not generated by Basic Class systems. If it is passed to such systems on a C-READY or C-REFUSE it is ignored.

4.2.2 Work specifications

A Basic Class implementation supports a restricted form of work specification. Work specifications are created on a Basic Class system as the result of an incoming transfer, or (if an initiation agency is supported) a J-INITIATE (see clause 4.3). Such work specifications contain only the information defined below. An attempt to transfer a more complex work specification to a Basic Class system results in refusal and a CCR diagnostic.

<work specification>::= <OSI job parameters>
<subjob specification>
<proforma list>
<local fields>
<document list>

<local fields> are as in 3.4.1, except that <time waiting> and <estimated size> are not present, and the <agency activity parameters> identify at most one agency activity.

<subjob specification>::= <OSI subjob parameters>
<JTM action parameters>
<error action>

<error action>::= TERMINATE

<proforma list>::= [<proforma>]*L

<document list>::= [<document>]*L

The <proforma list> for any work specification with a Basic Class <target> contains at most a single top-level proforma. Any proforma lists within this proforma are handled transparently. Work specifications produced at a Basic Class system by spawning, and targeted at some other system, contain a transparent proforma list.

NOTE — Proforma lists only occur within a proforma if the original work specification was created on an implementation with a functionality exceeding that required for Basic Class.

A Basic Class work specification contains at most a single <document>, referenced by the top-level subjob.

4.2.2.1 OSI job parameters

Basic Class restrictions on these parameters apply to the whole OSI job. Any OSI job which is to make use of a Basic Class system for the processing of any of its OSI subjobs is subject to the restrictions of this clause.

<OSI job parameters>::= <OSI job submission system>
<initiating identification>
<initiating time>
<OSI job name>
<OSI job local reference>
<audit trace>
<primary monitoring specification>
<authorisations>
<permissions>
<CCR parameters>

Note that <secondary monitoring specification>s, <accounts> and <security parameters> are not present.

NOTE — Interworking between an extended implementation requiring <accounts> and a Basic Class implementation is not possible. For this reason, extended implementations are required to be able to operate without <accounts>.

The fields of the <OSI job parameters> have the full form and meaning defined in 3.4.2, with the exceptions given below.

A Basic Class system can choose to ignore the <authorisations> and to perform authorisation using data contained in the <documents> being sent to sink or execution agencies.

NOTE — Basic Class systems are not required to support source agencies. Authorisation for J-GIVE is therefore not required, as access is only to an execution agency activity established by an ealier J-DISPOSE. The linkage between the two primitives is the agency activity id.

The <primary monitoring specification> does not contain any <relay>, and the <disposal instructions> do not have the value KEEP.

The <report selector> contains only the following event categories:

NORMAL-TERMINATION
MANIPULATION-TERMINATION
ABNORMAL-TERMINATION
USER-MESSAGE

4.2.2.2 OSI subjob parameters

<OSI subjob parameters>::= <subjob name-list>
<target list>
<urgency>
<subjob type>

The <target list> contains only <target> for any work specification transferred to a Basic Class system or generated there by spawning or by J-INITIATE. The <holds> is also absent from such subjobs.

These restrictions do not apply in proformas which remain transparent to the Basic Class system (are not spawned) (see 4.2.7).

The <subjob type> is one of

REPORT-MOVEMENT
DOCUMENT-MOVEMENT
WORK-MANIPULATION

Other <subjob type> values appear only in proformas which remain transparent.

4.2.3 Document movement operations

The following restrictions apply to any subjob whose <target> is a Basic Class system. They do not apply where a proforma is handled transparently (see 4.2.7).

There is a single <document movement>, and a single <se identification>. The <document block> is a <single form>.

The <se identification> is not the <FTAM se> form. It has no <additional authorisations>, the <agency parameter> is NULL, and the <se prefix> is empty.

The <document se reference> is not the <FTAM write-data> form, and the <se access parameter> is NORMAL.

There is a single <document pointer> which, on arrival at a Basic Class target system, has the value EMBEDDED, or contains a single <single document reference> with <state> FAILED.

In the latter case, the <source identification> is <JTM source>, there are no <additional authorisations>, and the <document source reference> is <JTM read-data>. The <CCR diagnostics> has the form defined in 4.2.1.1 above.

Where the <document movement operations> are in a top-level subjob produced at a Basic Class system by spawning, a <single document reference> with state NOT-ATTEMPTED can be present. If the <action open system> is the local system, the <source identification> is either PROVIDER or a <JTM source> which is an execution agency, and there is an <agency parameter> of NULL, and a <source access parameter> of MOVE.

4.2.4 Work manipulation operations

Work manipulation operations in a subjob targeted at a Basic Class system have the following restrictions.

The <work specification operations> consist of exactly two <work operation>s. The first <work-operation> is

SELECT <selector> (see below)

The second is one of

KILL
STOP
DISPLAY <doc-name> BRIEF

The <selector> consists of three <field test>s combined by <AND clause>s to give a <header test>. The <selector> is then

FIRST-HEADER-IS
<header test>

The three <field test>s are tests for equality on the three fields

OSI-JOB-SUBMISSION-SYSTEM
OSI-JOB-NAME
SUB-JOB-TYPE

Use of ANY-ITEM is not supported in Basic Class.

The **header list** conceptual data structure thus contains only a single header with only these three fields, in Basic Class systems.

Where a work specification contains embedded proformas, these are wholly transparent, and open system names carried in the <target> or <relay>s of these proformas do not provide implicit permissions for manipulation or display of the work specification.

4.2.5 Report movement operations

A <report movement operation> has the full generality specified in 3.4.4.5 except that

a) the monitor index is always zero (primary monitors only); and

b) the <disposal data> is not KEEP; and

c) the restrictions on the form of (se identification and <document se reference> defined in 4.2.3 apply.

4.2.6 Proformas

The top-level proforma in a Basic Class system has <spawning control data> set to COMPLETION. The proforma name is not used in any proforma reference in any embedded proforma.

4.2.7 Transparency of proformas

A work specification transferred to a Basic Class system has a <target> of that open system.

A work specification created by spawning (or by J-INITIATE) might (the first case) or might not (the second case) have that open system as the <target>.

In the first case, the subjob specified in the work specification has at most one top-level proforma; the main subjob, the top-level proforma, and the subjob specified in the top-level proforma are all subject to the Basic Class restrictions specified in this clause. The proforma-list in the top-level proforma is transparent and unrestricted.

In the second case, the subjob specified in the work specification is subject to the Basic Class restrictions, but the proforma-list in the subjob is transparent and unrestricted.

4.2.8 Transfer control records

Basic Class systems do not recognise the concept of transfer control records, nor do they support TCR-MANIPULATION subjobs.

All transfers made from such systems are scheduled in an implementation-dependent way.

4.2.9 JTM documents

A Basic Class system supports the generation and transfer of JTM work displays containing only <brief display>. It does not support such documents if they contain <full display>.

If the system provides support for generation of J-INITIATE-MAN, it supports the receipt of such documents and their disposal using J-DISPOSE.

If the Basic Class system provides support for a monitor point, it is capable of converting a <report> (but see 4.2.1) into a JTM report display document and disposing of this document using J-DISPOSE to a sink.

JTM TCR display documents are not supported.

A Basic Class system intended for general purpose use supports the document types listed in 4.1.2. It might also support other document types.

4.3 J-INITIATE primitive group parameters for Basic Class

In the Basic Class service, a J-INITIATE primitive group has a subset of the parameters available in the extended JTM service. The following restrictions apply.

The commitment level has only the values PROVIDER-ACCEPTANCE and AGENCY-ACCEPTANCE on J-BEGIN.

There are no <secondary monitoring specification>s. There are no <accounts>. There are no <security parameters>.

Only

J-INITIATE-WORK
J-INITIATE-WORK-MAN

are available.

The <subjob specification> for J-INITIATE-WORK is subject to the restrictions specified in clause 4.2, but **in addition** the following restrictions apply.

4.3.1 Top level parameters

<OSI job parameters>

<OSI subjob parameters>

<JTM action parameters>

<error action> This is TERMINATE

<proforma list> A single proforma or no proformas. (The single proforma has no embedded proformas.)

<document list> A single text (1) or print (2) document for J-INITIATE-WORK. No documents for J-INITIATE-WORK-MAN. If the document is print (2), there are no proformas.

4.3.2 OSI job parameters

<initiating identification> Any value.

<OSI job-name> Any value.

<authorisation data> An <authorisation element> and a <permission element> giving a user identification and password for use at the <target> open system. This is used to authorise JTM activity, such as manipulation, or the return of results. An execution agency can require the same information to be repeated at the head of the document ("JCL") passed on J-DISPOSE. An additional authorisation and permission element may be present to provide for action at the proforma target system. The permission element identification is always equal to the authorisation element identification.

<secondary monitoring specification>
No <secondary monitoring specification>s. The local management sets a <primary monitoring specification> with a <report selector> specifying one or more of the event types of ABNORMAL-TERMINATION, USER-MESSAGE, NORMAL-TERMINATION and MANIPULATION-TERMINATION, and with <disposal data> of their choice. KEEP is not used in basic class.

4.3.3 OSI subjob parameters

<target list> A single <target>; the <relays> are empty.

<urgency> MEDIUM only.

<holds> The <holds> is empty.

<subjob type> DOCUMENT-MOVEMENT or WORK-MANIPULATION only.

4.3.4 JTM action parameters

These are either <document movement operations> or <work manipulation operations>.

4.3.4.1 Document movement operations

There is a single <document movement> consisting of:

<document type> This is a document type of text (1) or print (2).

<se agencies> There is one <se identification> of the <JTM se> form containing

<se name> A value of a sink name (document type print (2)) or an execution agency name (document type text (1)) at the <target>.

<agency parameter>
This is NULL.

<se prefix> This is NULL.

<additional authorisations>
None

<document block> This is a <single form> containing <JTM write-data>:

<se access parameter>
Set to NORMAL.

<name-list> Set to the name of the document.

one <document pointer>
Set to EMBEDDED.

4.3.4.2 Work manipulation operations

There are two <work manipulation operation>s. The first is a <select operation>, and the second is a KILL, STOP or DISPLAY operation. The <selector> is

FIRST-HEADER-IS
AND OSI-JOB-SUBMISSION-SYSTEM
EQUALS <name>
AND OSI-JOB-NAME EQUALS <string>
SUB-JOB-TYPE
EQUALS <subjob type>

The DISPLAY operation requests BRIEF, and the <docname> is the string DISPLAY. The <name> is that of the open system where the J-INITIATE-MAN was issued.

4.3.5 Proformas

The single proforma, if present, consists of

<proforma name> Set to "OUTPUT".

A proforma body with:

<spawning control data>
Set to COMPLETION.

<OSI subjob parameters>
As in 4.3.3, but the subjob type is DOCUMENT-MOVEMENT.

<JTM action parameters>
See below

<proforma list> Proformas do not appear in proformas in Basic Class J-INITIATE.

The JTM action parameters in the Basic Class proforma consist of a single document movement operation with:

<document type> A minimal conforming OSI job submission system is capable of setting this field to print (2) (for J-INITIATE-WORK) and to work-display (4) (for J-INITIATE-WORK-MAN), and is not required to support the setting of any other values. Minimal conforming receivers of work specifications are not required to support any value other than those listed above.

<se agency> There is one <se identification> of the <JTM se form>:

<se name> An appropriate name for a sink at the proforma target.

<agency parameter>
This is NULL.

<se prefix> This is NULL.

<document block> This is a <single form> containing:

<se access parameter>
Set to NORMAL.

<name-list> A single name.

The <document block> also contains a single <document pointer> which is a <single document reference> giving

<action open system> This equals the initial <target>.

<source identification> This is PROVIDER in a proforma in a manipulation, otherwise it is a <JTM source>:

<source name> This equals the initial <se name>.

<agency parameter> This is NULL.

<document source reference>
This is <JTM read-data>:

<source access parameter>
This is MOVE.

<name-list> This is a suitable value for collecting the document from the target.

<embedded diagnostics>
This is EMBED

4.3.6 Summary of information contained in a Basic Class J-INITIATE request

4.3.6.1 J-INITIATE-WORK

The following information is passed to the JTM service provider in this primitive:

a) an initiating identification;

b) an OSI job-name;

c) the name of the target;

d) the name of a sink or execution agency at the target;

e) a user identification and password for use at the target;

f) a text (1) or print (2) document to pass to the execution or sink agency (respectively), and its name;

g) the name of the proforma target (if an execution agency is used);

h) a user identification and password for use at the proforma target if this is required;

i) the name of the resulting print (2) document produced at the initial target;

j) the name of a sink agency at the proforma target for the resulting print (2) document;

k) a name for the resulting print (2) document when passing it to the sink at the proforma target.

All other fields have fixed values.

4.3.6.2 J-INITIATE-WORK-MAN

The following information is passed to the JTM service provider in this primitive:

a) an initiating identification;

b) an OSI job-name for the manipulation;

c) an OSI job-name for the OSI job which is to be manipulated, and its subjob type;

d) the name of the target;

e) a user identification and password for use at the target;

f) whether the manipulation is a request to DISPLAY BRIEF, STOP or KILL the OSI job being manipulated;

g) the name of the proforma target;

h) a user identification and password for use at the proforma target if this is required;

i) a sink name for the work-display (4) document at the proforma target;

j) a name for the work-display (4) document when passing it to the sink at the proforma target.

4.4 Other primitive groups in Basic Class

4.4.1 J-DISPOSE

The full set of parameters defined in 3.6.2 are carried, with the following exceptions.

There are no <additional authorisations>, no <user account>, and the <agency parameter> is NULL. The <diagnostic information> does not contain any <FTAM outcome>.

The <disposal document> is either work-display (4) (Brief), report-display (5), text (1) (for an execution agency), or print (2) (for a sink agency).

4.4.2 J-GIVE

The full set of parameters defined in 3.6.3 are carried, with the following exceptions.

There are no <additional authorisations>, no <user account>, and the <agency parameter> is NULL. The <document group> is <activity end group> and the <document type> is print (2).

4.4.3 Other groups

The remaining primitive groups of Basic Class (J-MESSAGE, J-END-SIGNAL, J-STATUS, J-KILL, J-STOP) carry the full range of parameters defined in 3.6.5, 3.6.7, 3.6.8, and 3.6.9.

4.5 Summary of Basic Class primitives and parameters

A summary appears as table 4.

Table 4 — Summary of JTM Basic Class

Primitives	Request	Indication	Response	Confirm
J-INITIATE				
basic document movement specification	x			-
OSI job local reference	-			x
J-INITIATE-WORK-MAN				
basic work manipulation specification	x			-
OSI job local reference	-			x
J-DISPOSE				
provider activity id		x	-	
user authority		x	-	
agency parameter = NULL		x	-	
document se reference		x	-	
errors		x	-	
document type text (1), print (2), work-display (4) or report-display (5)		x	-	
agency activity id		-	x	
J-GIVE				
user authority		x	-	
agency parameter = NULL		x	-	
document source reference		x	-	
document type = print (2)		x	-	
activity end document group		x	-	
document and document type		-	x	
J-END-SIGNAL				
provider activity id	x			
J-STATUS				
agency activity id		x	-	
status message		-	x	
J-KILL				
agency activity id		x		
J-STOP				
agency activity id		x		
J-MESSAGE				
provider activity id	x			
message		x		

Annex A
(normative)

JTM service conventions

A.1 Introduction

ISO/TR 8509 (Service Conventions) is restricted in scope to the two-party operations used in the Network, Transport, Session and Presentation layers. This annex is modelled on ISO/TR 8509, making minimal changes to it, and defines the service conventions used in JTM.

A.2 Scope

This annex establishes definitions of terms and conventions for defining the services provided by JTM.

A.3 References

ISO 7498 : 1984, *Information processing systems – Open Systems Interconnection – Basic Reference Model.*

A.4 Definitions

A.4.1 This annex builds on the concepts developed in ISO 7498 and makes use of the following terms defined in that standard:

a) (N)-service;

b) (N)-facility;

c) (N)-layer.

A.4.2 For the purpose of this annex the following definitions also apply:

A.4.2.1 service-user: An abstract representation of an entity in a single system that makes use of a service; any one service user can be initiating a service, or can be responding to initiatives from the service provider as a result of earlier initiatives by some other service user or users.

A.4.2.2 service-provider: An abstract machine which models the behaviour of the totality of the entities providing the service, as viewed by a service-user.

A.4.2.3 layer service: A service provided by a layer of the Reference Model.

A.4.2.4 service primitive; primitive: An abstract, implementation-independent element of an interaction between a service-user and the provider.

A.4.2.5 request (primitive): A primitive issued by a service user to invoke a procedure (for a confirmed or non-confirmed service) by the service provider.

A.4.2.6 indication (primitive): A primitive issued by a service provider to invoke some procedure (for a confirmed or non-confirmed service) by a service user.

A.4.2.7 response (primitive): A primitive issued by a service user to complete the procedures associated with a confirmed service.

A.4.2.8 confirm (primitive): A primitive issued by a service-provider to complete the procedures associated with a confirmed service.

A.4.2.9 service element access point: The conceptual point at which interactions occur between the JTM service provider and a JTM agency.

A.5 Model for the JTM service

The JTM service is defined in terms of an abstract model (see figure A.1) having the following elements:

a) JTM-service-users;

b) JTM-service-provider.

Note that the use of the term "service users" for JTM does not imply the existence of a higher layer. It merely provides a convenient notation for describing the interactions with JTM agencies.

Each service-user interacts with the service provider by issuing or receiving service primitives. The service defines relations between interactions with one service user and consequential interactions with one or more other service-users.

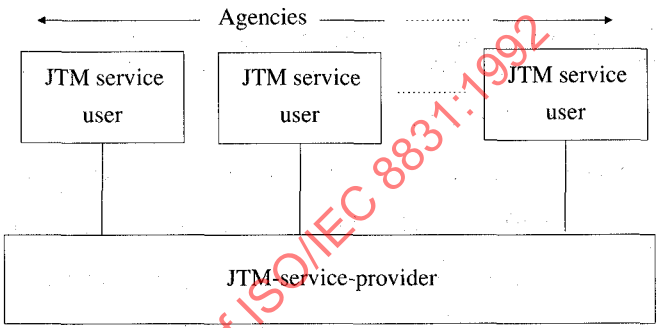

Figure A.1 — Service model

A.6 Service primitives

The use of primitives does not preclude any specific implementation of a service in terms of interface primitives. The following comments apply to this definition technique based on service primitives:

a) Service primitives are conceptual and are neither directly related to protocol elements, nor seen as macro calls of an access method to the service;

b) There are other equivalent sets of service primitives which can describe the same service;

c) Only service primitives which correspond to some element of the service involving other service-users are considered. The primitives which are only related to local conventions between the service user and provider do not relate to this description technique. For example, strictly local functions could be provided in some implementations. As they do not involve other users, such functions are not visible outside the local system.

A.6.1 Types of service

Two types of service are identified:

a) a non-confirmed service involving either

1) a single request primitive (see A.4.2.5) and one or more indication primitives (see A.4.2.6) at different service element access points; or

2) one or more indication primitives at different service element access points;

b) a confirmed service involving

1) a single request primitive (see A.4.2.5); and

2) one or more indication primitives (see A.4.2.6) and one or more response primitives (see A.4.2.2) both at different service element access points from 1) above; and

3) a single confirm primitive (see A.4.2.8) at the same service element access point as 1) above.

A.6.2 Properties of primitives

The occurrence of a service primitive is a logically separate and indivisible event. The event occurs at a logically separate instant, which cannot be interrupted by another event.

A service primitive has a direction which is either

a) from a service-user to the service-provider; or

b) from the service-provider to a service-user.

One or more parameters can be associated with a service primitive and each of these parameters has a defined range of values. Parameter values associated with a service primitive are passed in the direction of the service primitive.

A.6.3 Primitive names

The name of each service primitive contains three elements:

a) an initial (or initials) indicating the service (see A.8.1);

b) a generic name which indicates the primitive group (see A.8.2);

c) a specific name which indicates the type of primitive (see A.8.3).

A.6.4 Groups of primitives

The occurrence of a group of service primitives is not a logically instantaneous and indivisible event. The intervals between the constituent service primitives can be non-disruptively interspersed with other service primitives.

A group of primitives which includes primitives mapping to the CCR primitives is called a commitment group, but is otherwise named as for a primitive, e.g.

J-INITIATE commitment-group
J-DISPOSE commitment-group

A.7 Time sequence diagrams

Time sequence diagrams indicate

a) the sequence of interactions with a service-user;

b) where appropriate, the sequence of events between two or more service-users.

Each diagram contains one or more vertical lines representing the interaction between the service provider and a service user (a service element access point); for the initiating or master (commitment) user, the left of the line represents the service user, and the right of the line the service provider; in the other cases the service provider is on the left of the line, and the service user on the right.

Arrows, placed in the areas representing the service-user, indicate the direction of propagation of the primitives. The angle of the arrow in the fields has no significance.

Necessary sequence relations between the interactions are indicated by a horizontal dashed line. Where no time-relationships exists, there will be no dashed line. In this case all the primitives for one interaction can occur following or before all those for some other interaction, or interleaved with them.

The dashed line can contain an arrow tail or an arrow head at each service element access point. All interactions above all arrow tails necessarily occur before any interactions beneath any arrow head.

A.7.1 Non-confirmed service examples

The sequences in figure A.2 are all possible.

A.7.2 Confirmed service examples

The sequences in figure A.3 are all possible; note that in the second case the service-provider takes responsibility for completion of the service, and verifies that some error reporting mechanism is available, typically involving a further indication to the same or a different service-user.

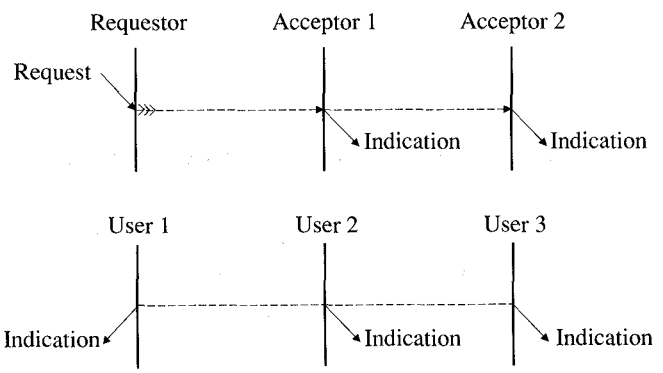

Figure A.2 — Non-confirmed services

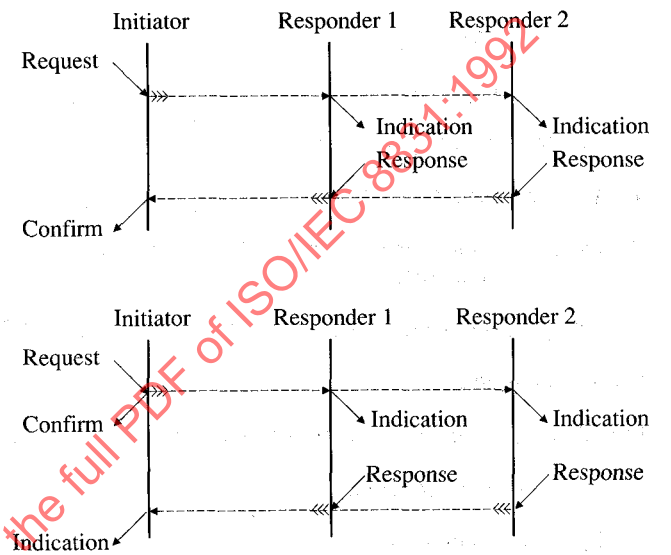

Figure A.3 — Confirmed services

A.8 Conventions for naming service primitives

A.8.1 Initials

The following initials are used to indicate services provided by the layers of the OSI model:

F	FTAM service;
J	JTM service;
V	VT service;
C	CCR service;
A	Other application-services;
P	Presentation service;
S	Session service;
T	Transport service;
N	Network service;
DL	Data Link service;
Ph	Physical service;

NOTE — The use of N to signify the Network service is not to be confused with the use of (N) – to signify a particular but unspecified layer of the model.

A.8.2 Generic name

A single word or a hyphenated word (usually consisting of the infinitive form of a verb) is used for the generic name e.g. INITIATE, DISPOSE, END-SIGNAL.

A.8.3 Specific name

The specific name consists of one of the following (indicating the type of the primitive or a primitive group):

a) request;

b) indication;

c) response;

d) confirm;

e) commitment group.

A.8.4 Representation

The initial is represented in the form given in A.8.1. The generic name is written in capital letters and the specific name is written in lower case letters.

The initial and the generic name are separated by a hyphen. The generic and specific names are separated by a space.

A.8.5 Examples

The following are examples of parameter names which use these conventions:

a) J-INITIATE request;

b) J-KILL indication;

c) J-DISPOSE commitment-group.

Annex B
(normative)

Registers of document types

B.1 Introduction

An enterprise can wish to establish a document type register for private use, or to register a document type in the ISO Register of Document Types.

This annex defines the information which is required by JTM for a Register Entry supporting document types carried by JTM.

B.2 Purpose of the entry

The purpose of the entry is to provide

a) an identification for the document type; and

b) the definition (possibly by reference) of the semantics and abstract syntax of the document which are standardised for this type; and

c) the definition (possibly by reference) of one or more transfer syntaxes for the document type; and

d) the definition of concatenation rules for the document type.

B.3 Identification

The document type is identified by an ASN.1 OBJECT DENTIFIER and by an ASN.1 ObjectDescriptor assigned according to the rules of ISO/IEC 8824.

B.4 Document semantics

The semantics which are standardised can be very complete, as is the case of document types defined to support JTM displays. For other document types, the semantics can be only partly standardised. An example of this is the "Simple ISO text document" document type (see ISO/IEC 8832) which is standardised only to the extent of saying that it consists of lines of graphic characters.

Document types with highly defined semantics can form a special case of documents with a lesser definition. In this case they could share the same transfer syntax.

The abstract syntax of a datatype supporting the transfer of the document type semantics is defined and named in accordance with the requirements of ISO 8822 for naming abstract syntaxes.

B.5 Transfer syntax

If communication is to be possible, there has to be agreement on the transfer syntax to be used for each type of document. This agreement can be bilateral, enterprise-specific, or expressed as an International Standard.

The register entry defines, either directly or by reference to encoding rules, the precise set of bits which are to be used to transfer the single presentation service data value (see ISO 8822) which is to carry the document semantics (for Basic Class transfers) and the precise sets of bits which are to be used to transfer the series of presentation service data values when extended implementations provide transfer.

NOTE — Neither JTM nor the presentation service restricts the form of specification or encoding for abstract and transfer syntaxes, except that in Basic Class only a single set of bits are transferred for each document.

The transfer syntax for the document type is named in accordance with the requirements of ISO 8822for naming transfer syntaxes.

NOTE — The above text has assumed a single abstract syntax name for all the data values used in forming a document type, and a single transfer syntax for that abstract syntax. This restriction is recommended, but is not required by JTM. The full generality of the presentation layer for using different abstract syntaxes for each data value, and for having embedded data values from different abstract syntaxes is available, as is the ability to select from a number of transfer syntaxes supporting each abstract syntax.

B.6 Relation of syntax and semantics

It is possible (but unlikely) for two totally different abstract syntaxes, and associated with totally different semantics, to produce identical bit patterns for transfer through the use of different encoding rules.

It is also possible (and more likely) for two totally different document types, with different semantics, to share a common abstract syntax.

Thus it is not possible, in general, to deduce the abstract syntax from the transfer syntax, nor to deduce the document semantics (the document type) from the abstract or transfer syntax.

Given a document type, however, it is possible to deduce (from the register entry), one or more possible abstract syntaxes, and from these to deduce one or more transfer syntaxes.

Correct interpretation of a received string of bits (transfer syntax) requires knowledge of the encoding rules which were used to generate the bits. Correct interpretation of an abstract syntax similarly requires knowledge of the semantics being carried.

The JTM protocol explicitly carries the document type name with each document, thus defining all standardised aspects of the semantics. The use of a particular transfer syntax, if there is more than one available, is negotiated using the Presentation Service.

B.7 Syntactic considerations

The JTM protocol is defined using the abstract syntax notation ASN.1, but such a restriction is not imposed on document types.

Each document is carried as a separate presentation data value, and all problems of delimiting documents and octet alignment are resolved by the Presentation Service.

B.8 Contents of the register

Each Register Entry contains the information defined in the following sub-clauses.

B.8.1 Document type identifier

This is an ASN.1 OBJECT IDENTIFIER value and an ASN.1 ObjectDescriptor value.

B.8.2 Document semantics

The definition (either directly or by reference to another standard) of those parts of the total semantics which are standardised for this document type.

B.8.3 Document syntaxes

The definition (either directly or by reference to another standard) of one or more transfer syntaxes (possibly via one or more abstract syntaxes and encoding rules) which will support the document semantics.

Where multiple transfer syntaxes are specified then one of them is selected as a minimum implementation requirement for conforming systems which claim to support this document type.

B.8.4 Syntax names

Abstract and transfer syntax names to support the transfer of the document as one (for JTM Basic Class) or more (extended) presentation service data values (see ISO 8822).

B.8.5 Concatenation

A statement of which document types can be involved in the concatenation operation:

A concatenate B to give C

and the definition of the operation in each case.

NOTE — The common case will be where A, B, and C are all the same document type.

B.8.6 Additional information

Additional information can be needed if the document type is to be used in other protocols, but is not required by JTM.

Annex C
(informative)

Tutorial material

C.1 Introduction

The Job Transfer and Manipulation work was originally designed to support remote off-line processing. It does not, however, set out to in any way standardise "JCL", nor to model in any detail what constitutes "processing".

It recognises the requirement to transmit (transparent) documents to a system for "processing", together with information which will enable that system to correctly dispose of documents resulting from the "processing".

The documents could be traditional JCL, program and data, and the "processing" could involve queueing for a "job-mill" style of activity. On the other hand, the documents could be a letter with enclosures, and the "processing" could be by a human being. A third possibility is trade documents or requests for resupply, and "processing" could involve the immediate scheduling or despatch of a lorry, with the output documents being invoices or management statements or notification of expected arrival times.

JTM is concerned with so-called **OSI jobs**. These should not be confused with traditional "jobs", processed by a single machine.

The work JTM undertakes, the OSI job, is the movement of documents (files) between open systems, a pause for those documents to be processed by the open systems (transparently to JTM), then the movement of documents resulting from that processing, and so on.

The processing which is being carried out is of no interest to JTM. Its concern is purely with the documents it delivers, and in due course with the delivery of further output documents. The processing can be carried out by a traditional job-mill (in which case one of the documents that JTM delivers will be some form of "JCL"), or by some application-specific process, such as one handling orders and despatch of goods. The processing could even be entirely manual, with a human signal to JTM that output documents are available, and that the work of the OSI job can proceed.

C.2 Architecture

The JTM "architecture" involves the concept of a so-called "initiation agency" – typically, but not necessarily – representing a human user. This agency determines the possible patterns of document movements which are to occur, the open systems to be used, and where the initial documents are to come from. This is all specified in what is called "a work specification". Understanding JTM is almost entirely a question of understanding the very wide range of activity which can be specified in a work specification.

The work specification is created by the JTM service provider from the parameters of the J-INITIATE service primitive. Thereafter, the JTM service takes full responsibility for progressing the activity. A key feature of the JTM protocol design is that responsibility for progressing a work specification passes from open system to open system. The open system on which J-INITIATE was issued need have no memory of the activity once the first part of it is completed and responsibility passed.

Typically, protocol elements (called Transfer Elements) containing some or all of the information in the work specification are passed around a number of open systems, each of which performs some part of the requested activity – providing a document, accepting a document for printing or filing or display to a user, processing one or more documents, and so on.

C.2.1 Use of CCR

All JTM activity (service primitives and protocol transfers) is performed within one or more CCR atomic actions. This gives complete protection against application failures and communications failures, resulting in a "no loss, no duplication" guarantee for the requested work.

C.2.2 JTM agencies

The JTM Standards recognise not only an initiation agency (defining the work by use of J-INITIATE), but also of agencies accessed by the JTM service (through further service primitives) to enable the work to be carried out. These are described in the following paragraphs.

C.2.2.1 Source agencies

The JTM "source agency" is accessed by the JTM service provider through the J-GIVE service primitive. This is issued (as a result of requirements in a work specification) to request a named document from the source agency. The source agency **typically** models a conventional local filestore, but it can also model some running program, an operator being requested to load a magnetic tape, or an FTAM handler obtaining a document remotely. Note that there is no restriction placed by JTM on the open systems where J-GIVE primitives can be issued. Open system A can issue J-INITIATE, intending to process some documents on open system B, but the J-GIVE primitives to obtain the initial documents can be issued on open system A, open system B, or on entirely separate open systems C, D, E, ... The JTM protocol arranges for the necessary requests and documents to be transferred between systems.

C.2.2.2 Sink agencies

The JTM "sink agency" is accessed by the JTM service provider through the J-DISPOSE service primitive. This is issued to request the agency to dispose of a document supplied by the service (typically obtained by an earlier J-GIVE). This disposal can (depending on the agency) be to a file, to a line-printer, to a plotter, a short message to a user mailbox, an operator message, a signal to a running program, etc. etc. As for J-GIVE, there are no restrictions (subject to authorisation, discussed below) on where J-DISPOSE can be performed.

C.2.2.3 Execution agencies

The JTM "execution agency" is also accessed by the J-DISPOSE service primitive. The difference between this and a sink agency is that the execution agency is able to generate new documents for collection (using J-GIVE), as part of the activity stimulated by J-DISPOSE. This forms the "processing" element within the JTM model.

C.2.3 Commitment level

JTM makes use of the so-called "commitment level" mechanism in CCR primitives. The atomic action containing a J-DISPOSE to a sink or execution agency can either

— complete with COMPLETION commitment level; in this case all requested activity (printing or processing the document) is completed prior to commitment;

— complete with AGENCY ACCEPTANCE commitment level; in this case the JTM service provider simply stores (securely, on disc) the remaining requests in the work specification and awaits a signal from the agency (the J-END-SIGNAL service primitives) that the activity is now complete.

If AGENCY ACCEPTANCE occurs, then the agency will at some later time (possibly as the result of a human interaction saying, for example, that a lorry has been despatched), issue a J-END-SIGNAL request to inform the JTM service provider that the work is completed. If the agency was an execution agency, J-GIVE primitives can now be issued to collect documents from it, and the work specified by the work specification continues.

C.2.4 Disposal of new documents

The documents produced by an execution agency will in due course be passed to other sink **or execution** agencies in further J-DISPOSE primitives. Thus JTM supports an arbitrary (unbounded, and potentially infinite) sequence of J-GIVE/J-DISPOSE, J-GIVE/J-DISPOSE activities. (See figure C.1) It should be noted that this is a branching process, with each branch proceeding asynchronously.

C.2.5 On-line processing support

The activity requested on J-INITIATE can be performed "off-line" as described above. If, however, the initiation agency demands a CCR commitment level of COMPLETION on the J-INITIATE, then the entire activity is carried out immediately (or not at all) with a suitable response to the J-INITIATE.

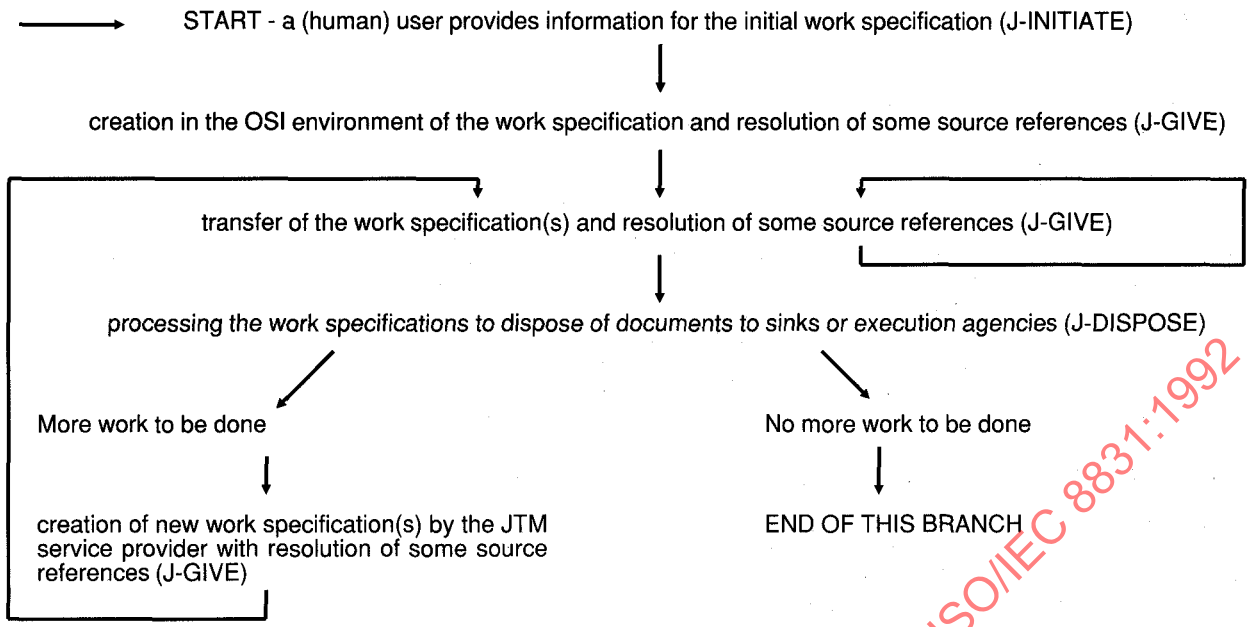

Figure C.1 — An illustration of work specification processing

C.2.6 Sequences of primitives

The activity requested by the initiation agency in general causes a tree-structure of J-service primitives to be issued, possibly *in parallel, possibly sequentially, on many different systems. These primitives are* issued as part of one or more atomic actions (depending on the requested commitment level). Some examples appear in clause C.16.

C.3 Reporting

As part of the work specification defined by J-INITIATE, the initiation agency identifies one or more **monitor points**, and one or more **reporting categories**. The JTM service generates reports (documents containing JTM specified, formatted, data) when any event in these categories occurs. The categories of reporting include the completion of parts of the work, transfer between systems of responsibility for progressing the work, accounting information, and, most importantly, errors in the specification *which prevent further processing. (Such errors can either cause the rest of the work to be abandoned or,* depending on what the work specification says, simply suspended for corrective action.)

C.3.1 Delivery and disposal

All requested reports are delivered (using a protocol almost identical to that used for progressing the main work) to the open systems containing the specified monitor points. Two options now exist; either

— the report is turned into a document and a J-DISPOSE is issued to a sink agency at the monitor point, typically resulting in printing or display of the report document;

— the report is held (securely, on disc) by the JTM service provider; it can be interrogated from any other open system by a subsequent (human) initiative as described below.

C.3.2 Responsibility for reporting

For each category of report, the responsibility for generating a work specification for delivery and disposal of the report rests with one particular open system in the CCR atomic action tree. This is specified in the JTM protocol specification, and described in the body of this text.

Some reports (for example, those reporting creation of a work specification, transfer of responsibility for it, normal termination) are carried out as part of the main CCR atomic action progressing the work. They are committed only if this action commits, and rolled-back otherwise. In CCR terms, they are part of the bound data of the action.

Other reports (for example, reports of accounting information, reports of attempted security violations, or reports – by the open system containing the commitment master – of the failure of an atomic action) are issued as separate atomic actions. These reports are committed even if the main atomic action is rolled back.

C.4 Manipulation

Following a J-INITIATE, it is possible for the work to progress with no further human interaction. More typically, however, further human interaction will occur in order to

— display the current status of work specifications submitted earlier;

— modify work specifications, possibly to correct errors reported to monitor points;

— interrogate monitor points which are storing reports as described in C.3.1;

— provide operator control over the scheduling of the various transfers the JTM service is performing to carry out the work specifications submitted to it.

These interactions are also performed by an initiation agency (any initiation agency). The agency again supplies information for a work specification, but the requested work is now one of the operations listed above. The agency again has the choice of setting up "off-line" activity, or (more typically in these cases) of demanding COMPLETION commitment level on the J-INITIATE.

C.5 J-INITIATE primitives

The service primitive submitting a normal work specification is called J-INITIATE-WORK. That requesting display or modification of work specifications is called J-INITIATE-WORK-MAN ("MAN" for MANipulation). That interrogating monitor points is called J-INITIATE-REPORT-MAN. That performing control of transfers is called J-INITIATE-TCR-MAN. ("TCR" stands for "transfer control record"; transfer control is described in C.11.6.)

In all cases of J-INITIATE, the requested activity is described by (some type of) work specification. All types of work specification contain a number of global fields concerned with authorisation, identification, and security. These are formed from J-INITIATE parameters supplemented by information provided by the JTM service provider to form the so-called "OSI job parameters". These parameters are carried in the protocol for all JTM transfers concerned with this work specification, (including report movement) and serve to coordinate the entire activity. These fields of a work specification are described in clause C.9, after introducing some of the concepts.

C.6 Naming

JTM recognises two sorts of objects which require (world-wide) unambiguous names to be allocated. One of these identifies a JTM implementation. Such names are expected to be (JTM) application-entity-titles as described in the Addressing Addendum to the Basic Reference Model. The second is what JTM calls "user-identification-authority names". These use the same name-space as application-entity-titles, and represent a source of user identifications. Frequently, such user identifications are issued for use within and by a single computer system (open system). In this case a single application-entity-title could cover both roles. In other cases, a single computer system (open system) might have more than one set of user identifications, or, perhaps more commonly, a single set of user identifications might be issued to cover several computer systems (open systems). The world-wide unambiguity of application-entity-titles is ensured by the aims of the work on naming and addressing. World-wide unambiguity of user identifications and of work specification identifications follows from this by incorporating such names into these identifications.

C.7 Authentication

Unambiguous naming mechanisms do not solve the problem of masquerade - a system or user deliberately using the name assigned to some other system or user. Masquerade is prevented by suitable **authentication** mechanisms. JTM recognises three mechanisms which can be applied to achieve authentication.