DISTRIBUTED TRANSACTION PROCESSING STANDARDS AND THEIR APPLICATIONS

GRAHAM CHEN

***Abstract*—The OSI Standards committee (ISO SC21 Working Group 8) has defined the distributed transaction processing (DTP) standards that consist of a reference model, service definitions and protocol specifications. The reference model can be used for defining and building distributed transaction processing systems and applications.**

X/Open is an industrial standards organisation that has also defined a reference model for OSI DTP. The purpose of this model is to provide industry with a functional decomposition that captures the functionality of OSI DTP, and that can be architected, designed and implemented. In addition to the reference model, X/Open has also defined application program interfaces and system interfaces between the system components defined in the model. The interfaces provide a standard way for applications and resource management technology to work in a DTP environment.

In this paper, OSI DTP concepts, model, services are introduced and their application to some other OSI technologies is discussed. Also, the X/Open DTP reference model and its application program and system interfaces are discussed. Finally, the applications of the model and interfaces in building distributed applications are discussed.

***Keywords*—Distributed transaction processing; X/Open; RDA; systems management.**

***Source of Publication*—This paper was published in the Computer Standards and Interfaces 17 (1995) pp 363-373.**

1 INTRODUCTION

The International Standards Organisation (ISO) work on OSI Distributed Transaction Processing (DTP) has stabilised at *International Standards* (IS) status. The standards consist of three main parts: the DTP reference model [1], the DTP service definition [2] and the DTP protocol specification [3].

The OSI DTP is designed to support the synchronisation and co-ordination of distributed applications to ensure that a unit of work (that is, a transaction) is reliably completed on multiple open systems.

In the OSI DTP model, a transaction is defined as a unit of work performed by a set of transaction partners. A transaction is characterised by four attributes:

- *Atomicity*—The unit of work is either performed entirely or not performed at all.
- *Consistency*—The unit of work is performed accurately, correctly and with validity.
- *Isolation*—Partial results of the unit of work are not accessible.
- *Durability*—All effects of the unit of work are not affected by any kind of failure.

These are known as the *ACID* properties of the transaction. A distributed transaction is defined as a transaction that may span more than one open system.

The X/Open company has defined a reference model for distributed transaction processing. The model defines a functional decomposition of the OSI DTP services. It also defines the Application Program Interfaces (APIs) and the system interfaces between the functional components. The model defines how an application program, a transaction manager, and one or more resource managers cooperate to achieve application portability and component interchangeability in a distributed transaction processing environment. The reference model illustrates how application programs can achieve these goals via a set of published interface specifications.

This paper introduces the basic concepts of distributed transaction processing and the reference models defined by the OSI committee and by the X/Open company. In Section 2, the OSI DTP model and service definitions are discussed. In Section 3, the X/Open DTP model and interface APIs are presented. Section 4 discusses the application of the OSI and X/Open DTP models in some areas for building large distributed applications.

2 OSI DTP REFERENCE MODEL

2.1 CONCEPTS OF OSI DTP

This section informally introduces some basic concepts of OSI DTP. All these concepts are defined precisely in the relevant standards documents which are included in the reference list [1,2,3].

ASSOCIATION

Two open systems participating in a distributed processing first need to set up connections for exchanging information. This connection is called an *association*. The establishment of the TP associations is supported by the ACSE [14] that supports the DTP.

DIALOGUE

Two TP service user invocations communicate in a peer-to-peer relationship. This relationship is called a *dialogue*. Once a TP dialogue is established, the two participants may communicate to transfer data, notify errors, initiate and complete a transaction, terminate a dialogue, or synchronise activities. A dialogue has an initiator acting as a *superior* and a responder as a *subordinate*. TP dialogue is a logical concept and is independent from the underlying association, although a TP dialogue is supported by one or more TP associations.

DIALOGUE TREE

A *dialogue tree* is a directed tree with TP service user invocations as nodes and dialogues as arcs between nodes. A dialogue tree represents the communication topology of the open systems participating in the distributed processing.

TRANSACTION BRANCHES

A portion of a distributed transaction performed by a pair of TP service user invocations sharing a dialogue is called a *transaction branch*. A transaction branch has an initiator acting as a transaction superior and a responder as a transaction subordinate.

TRANSACTION TREE

A *transaction tree* is a directed tree with TP service user invocations as nodes and transaction branches as arcs between nodes. A transaction tree represents the topology of the transaction among the open systems participating in the distributed processing. The root of the transaction tree is the TP service user invocation that does not have a superior node and a leaf is a node without any subordinate node. An intermediate node of the tree is a subordinate with respect to its superior but it creates a number of transaction branches according to the application semantics. It acts as the superior with respect to the transaction branches it created.

SEMANTICS OF TRANSACTION COMPLETION

The semantics of the transaction completion are defined in terms of transaction *bound data*. The bound data is some specific TP service user data as determined by the requirements of the distributed application. The TP service modifies this bound data from its initial state to the final state according to the application's semantics. These modifications are indivisible, that is, only two results can be returned: the initial state of the bound data or the final state of the bound data.

To maintain the properties of a transaction is, in essence, to maintain the indivisibility of the bound data of the transaction. The property of atomicity is manifested by the fact that only the initial state or the final state of the bound data is regarded as the result of the operation. The property of isolation is manifested by the fact that any intermediate modification to the bound data is invisible to the concurrently executed operations outside of the transaction. Therefore, the semantics of commit are to release the final state of the bound data and make the update to the resources permanent. The semantics of rollback are to release the bound data in its initial state and not to update the resources at all.

2.2 OSI DTP SERVICES

OSI DTP defines a number of services among which is the support for the establishment of TP dialogue trees and TP transaction trees. However, the most important services OSI DTP support are those to coordinate shared resources to achieve the successful completion of transactions and to recover from the failure of computer systems or communication.

2.2.1 COORDINATION OF RESOURCES

The TP service provider coordinates the resources participating in the transaction in order to achieve consistent commitment semantics by using the two-phase commitment protocol. If the commitment of a transaction is not possible, the transaction is rolled back.

After a transaction begins, the root node sends the application service requests. These requests access and update certain parts of the resources involved in a transaction. These parts form a transaction's bound data. The application requests transform the initial state of the bound data to the final state of the bound data. At some later stage, the application may request the root node to complete the transaction (either by commit or rollback) using two-phase commit protocol.

When the root requires the completion of the transaction, the TP service provider achieves this completion in the following two phases:

COMMITMENT PHASE 1—PREPARE

At phase 1 of the commitment, each subordinate node is informed by its superior that the completion of transaction phase is entered. The node understands that no more application services will be required from the superior and that each node should prepare for the commitment of the transaction. If the node agrees to proceed, it will make the bound data, within its subtree, ready for commitment. It then informs all its subordinates (if any) to prepare for the commitment, recursively.

If any non-root node is not able to make bound data ready for commitment, it initiates a request to roll back the transaction within its subtree and returns a rollback response, instead of a ready response, to its superior.

COMMITMENT PHASE 2—COMMIT

During phase 1 of the commitment, if all the subordinates within the transaction tree are ready for commitment and have responded to their superior's request positively, the completion process of the transaction enters its phase 2. The root then orders its subordinates to commit—to release the bound data in its final state and each node in turn acts as a superior to inform its subordinates (if any) to commit, recursively.

COMMITMENT PHASE 2—ROLLBACK

If any node in the transaction is not able to release the bound data in its final state, it issues a rollback transaction request within its subtree and releases the bound data in its initial state. It also sends a rollback response to its superior. Its superior, upon receiving the response, orders its subordinates to rollback the transaction, recursively.

After these two steps are complete, the commitment of the transaction is complete.

2.2.2 RECOVERY

OSI DTP adopts a *presumed rollback* recovery mechanism. During recovery after a failure, if the transaction coordinator (superior) ordered the commitment of the transaction, then it is responsible for remembering its subordinates and for informing them of the final outcome of the transaction. If a subordinate has offered *ready* to its superior and suffered a failure, then it is the subordinate's responsibility to obtain from the superior the outcome of the transaction. If the superior has no knowledge of the transaction, the subordinate presumes that the transaction should roll back. For the purpose of recovery, the subordinate is required to record the bound data after it offered *ready* to the superior.

The transaction coordinator (superior) is required to record the state of its subordinates when their *ready* responses have been received and the transaction can commit. All nodes, except for the root, must record the state of their immediate superior when they have received *ready* responses from all subordinates and are themselves *ready*, and before responding *ready* to the superior.

3 X/OPEN DTP REFERENCE MODEL

The X/Open reference model is not a design for a specific system or product. The model provides a framework on which a standard set of application programming interfaces (APIs) and integration interfaces may be specified. Based on this framework and standard APIs, the real DTP systems can be built to achieve interoperability between heterogeneous TP systems.

The main objectives of the X/Open DTP model are to provide portability of transaction processing applications. These objectives are realised in the reference model by an architecture whereby an application program can access several resource managers under the coordination of a single transaction manager.

3.1 REFERENCE MODEL

Figure 1 shows the principal components of the X/Open DTP reference model [[7]]

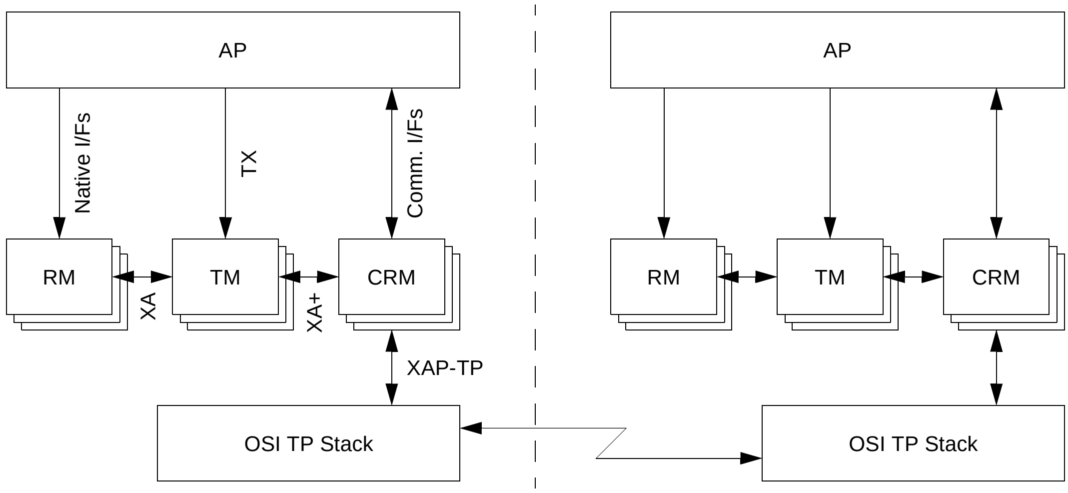

Figure 1: X/Open Reference Model and Interfaces

Five principal components are defined in the X/Open DTP reference model:

- *Application Program (AP)*—the component that implements end-users' semantics, specifies actions that constitute a global transaction and defines global transaction boundaries.
  X/Open DTP standards do not use the term *distributed transaction*; instead they use *global transaction* for this concept. A global transaction collectively describes all of the work done by participating resource managers anywhere in the network for a single unit of work. An AP defines the start and end of a global transaction. A single, coordinating transaction manager manages its initiation and completion.
- *Transaction Manager (TM)*—the component that assigns identifiers to transactions, monitors their progress, and takes responsibility for transaction completion and for coordinating failure recovery.
- *Resource Manager (RM)*—the component that provides access to resources. Typically, these can be database systems.
  Many RM products structure their own work into transactions. An RM-internal transaction is a recoverable unit of work owned by a single RM. In the X/Open DTP model, a global transaction consists of one or more RM-internal transactions. The TM coordinates the start and completion of the RM-internal transactions of each participating RM.
- *Communication Resource Manager (CRM)*—the component that controls communication between distributed applications.
- *Communication Protocol Stack*—the component that provides the underlying communication services used by distributed applications and supported by CRMs.

The X/Open DTP model does not insist that an AP performs all work within the scope of a global transaction. Work performed when no global transaction is defined is termed a *nonglobal transaction*. In nonglobal transaction processing, one or more RMs are associated to the AP, but not to the TM. RM-specific work may be committed using the appropriate native interface commands, such as SQL COMMIT. As far as individual RMs are concerned, the work performed is transactional and the ACID transaction properties are obeyed, but if two or more RMs are involved, then there is no coordinated commitment between them.

An AP may choose to run nonglobal and global transactions with the same TM domain. The two approaches for such a combination are:

- The AP runs nonglobal and global transactions in parallel.
- The AP switches between nonglobal and global transactions, but never runs both types at the same time.

In either case, the coordination of the global and nonglobal transactions is an additional issue of application design. Most applications should not need to mix nonglobal and global transactions.

3.2 THE INTERFACES

The reference model defines the following six interfaces between the components:

- AP-RM *interface (Native Interface)*
  This interface enables the application program to access shared resources. The X/Open DTP AP-RM interface allows the existing X/Open interfaces such as SQL to be used to provide the portability. In this case the AP may be portable to other RMs that use the same interface. The RM may, on the other hand, offer a proprietary interface specific to its services.
- TM-RM *interface*
  The TM-RM interface (the XA interface [[9]]) enables the TM to structure the work of RMs into global transactions and coordinate the global transaction's completion and recovery (in case of failure).
- AP-TM *interface (Global Transaction Demarcation)*
  The AP-TM interface (the TX interface [[8]]) allows the application to delimit the global transactions. The TM organises transactions based on the requests from the application program and coordinates the two-phase commit protocol among participating RMs and other remote TMs.
- AP-CRM *interface*
  X/Open has specified a number of portable APIs for DTP communication. The specifications correspond to the following communication paradigms:
  – peer-to-peer communication [10],
  – client/server communication [11], and
  – remote procedure call (RPC) [12].
- TM-CRM *interface*
  This interface (the XA+ interface [13]) supports global transaction information flow across TM domains. A TM uses the XA+ interface to communicate with remote TMs participating in global transactions.
- CRM-OSI TP
  This interface (XAP-TP) provides a programming interface to OSI DTP services. The use of this interface is not mandatory since the use of the OSI DTP protocol for communication is not mandatory.

3.3 COORDINATING RESOURCES UNDER THE X/OPEN MODEL

X/Open reference model uses the *two-phase-commit with presumed rollback* defined in ISO OSI DTP standards as its transaction completion protocol.

This example explains how the X/Open DTP model works. The X/Open DTP environment is used to coordinate a local database access and a remote one, and ISO Remote Database Access (RDA) is used as the communication stack.

Firstly, the application program (AP) issues a call to its communication resource manager (CRM), in this case through an RDA client, to set up the communication channel with the remote AP through its CRM.

The AP then issues through TX interface a *tx_begin*() command to begin the global transaction. The TM will then propagate this *begin transaction* command by calling XA interface function *xa_start*() to tell the local database RM, and then CRM to tell the remote CRM, which in turn tells the remote TM and RM.

After the global transaction has been successfully initiated, the AP then sends SQL operations to the local RM (using a native interface such as SQL CLI) and to the remote CRM via RDA client. The remote RDA server then passes the SQL operations to the RM (using a native interface such as SQL CLI). The TMs in both the local and remote domains have no knowledge of these interactions.

When the AP decides to commit the global transaction, it issues a *tx_commit*() call to its local TM. The TM is responsible for coordinating the two-phase commit protocol. The local TM issues a *prepare* protocol via the XA function call (first phase). This call notifies the local RM to prepare for commitment. It also notifies the remote RM to prepare. This is achieved by the local CRM that propagates the call to the remote TM, and the remote TM in turn notifies the remote RM (RDA server). If the RMs have successfully prepared to commit, the local TM issues the *commit* protocols to the local and the remote RMs using the same mechanism (second phase).

If the transaction is completed, the TM returns the *tx_commit*() call back to the AP, indicating the end of the global transaction.

The following figure illustrates the flow of calls and protocols between the initiating AP and the rest of the system components:

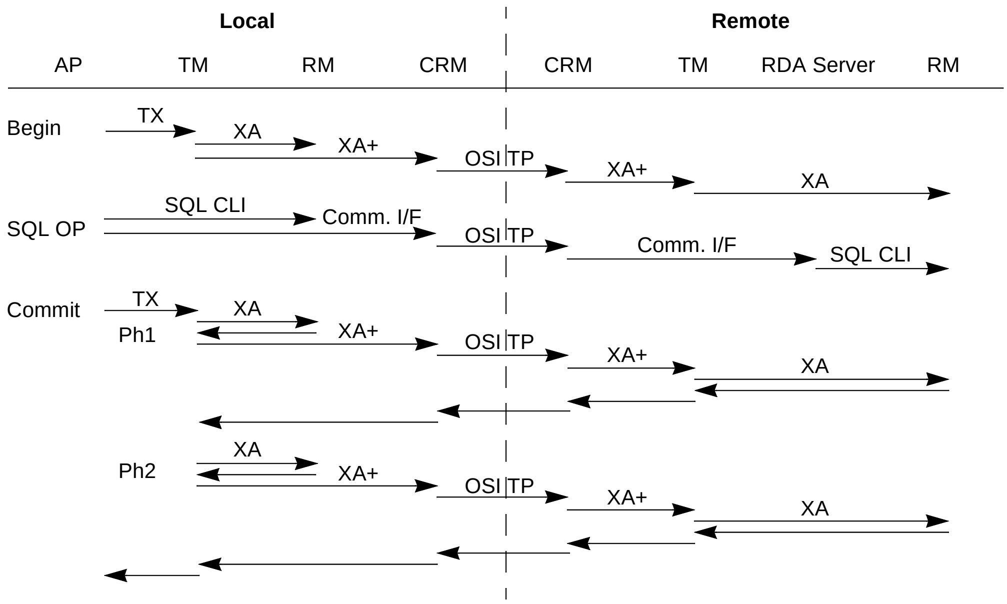

Figure 2: Flow of Function Calls and Protocols in the X/Open DTP Model

4 APPLICATIONS OF OSI AND X/OPEN DTP TECHNOLOGY

This section discusses the application of the OSI DTP and X/Open TP reference models. The following areas have been investigated:

- Relational databases participating in DTP as shared resource managers;
- ISO remote database access (RDA) facilitating communication for access remote database systems;
- ISO systems management services facilitating communication for performing distributed management functions.

4.1 DATABASE SYSTEMS AS RESOURCE MANAGERS

A relational database management system is consistent with the definition of an RM. When participating in a distributed transaction processing environment, it controls the shared database resource and it operates under the control of the TM.

4.1.1 REQUIREMENTS OF AN RM

The reference model of X/Open DTP specifically allows for nonglobal transactions and the use of the native interface facility to complete these. However, in RMs such as SQL RDBMSs where the native interface defines transactions, an AP must not use these services in a DTP context when a global transaction is initiated, since TMs have no knowledge of an RM's transaction.

4.1.2 GLOBAL VS NONGLOBAL TRANSACTIONS IN SQL

The support in SQL database standards for an X/Open style of global distributed transaction processing is very important. The ability to work with X/Open DTP architecture enables SQL database systems to participate in global transactions involving other resource management systems. A typical example lies in the system and network management area. With managed networks and systems growing larger and larger, more and more management systems are using SQL databases as system repositories, and the ability for SQL database systems to participate in the global management application is essential.

On the other hand, the transaction facilities in SQL standards have some unique features that are not considered in X/Open's definition of transactions. These include the isolation level and access control. Furthermore, with the client/server architecture and the transaction management facilities in the SQL standards, a large range of industrial requirements in transaction processing can be satisfied. It is also the intention of X/Open to include more resource management systems into the global transaction processing environment, including those with their own transaction management facilities, such as RDBMSs.

4.1.3 SPECIALISATION OF THE XA INTERFACE

When an SQL database is treated as an RM participating in global distributed transactions, the XA interface should be supported. Although X/Open has specified the interface definition for XA, it is a general specification for any type of shared resources. It appears that an SQL specialisation of such an interface is required, particularly in the following areas:

- Language mapping
  The mappings from XA functions to SQL statements need to be specified.
- The two-phase commit protocol
  The definition of the two-phase commit protocol in an SQL database should be defined. In particular, the semantics of *prepare* and *recovery* responsibility need to be specified.
- Relationship between global and nonglobal transactions
- Specific SQL features

This specification will give sufficient guidelines for the industry to implement the XA interface so that the X/Open DTP model can be used to coordinate heterogeneous SQL databases.

4.2 OSI REMOTE DATABASE ACCESS WITH TP APPLICATION CONTEXT

ISO Remote Database Access (RDA) [5,6] is a set of international standards that enables access to remote databases and defines the database languages used to manipulate the server. Its model provides the capability for access to any type of database, but currently only SQL databases are supported.

The following benefits are observed in using the RDA technology:

- It makes database applications more portable.
- It separates the application development tools from the back-end database.
- It allows database applications to make use of more than one vendor's database server concurrently.

The RDA standard defines the database language (DBL) that must be supported, the set of RDA services available, and the order in which the RDA services can be requested. It also specifies the means of exchanging DBL statements, input data, results, and errors arising in executing DBL statements.

RDA does not define a programmatic interface. Therefore access to the RDA service is through an SQL agent not exposed to the application. The use of RDA provides a communication function that is transparent from the database applications.

4.2.1 REQUIREMENTS FOR TRANSACTION PROCESSING

RDA does not support distributed databases as such. However, it is important for RDA to be used in a distributed environment for accessing remote data resources. Many distributed applications require the use of RDA as the communication component. In order to meet this requirement, RDA has defined an application context with transaction processing. This application context allows RDA to operate within a DTP environment.

4.2.2 RDA AS COMMUNICATION RESOURCE MANAGER

The application context fits into the X/Open reference model in such a way that RDA behaves like a CRM by providing communication with remote database resources on behalf of the application program.

Figure 3 depicts how RDA fits into the X/Open DTP environment.

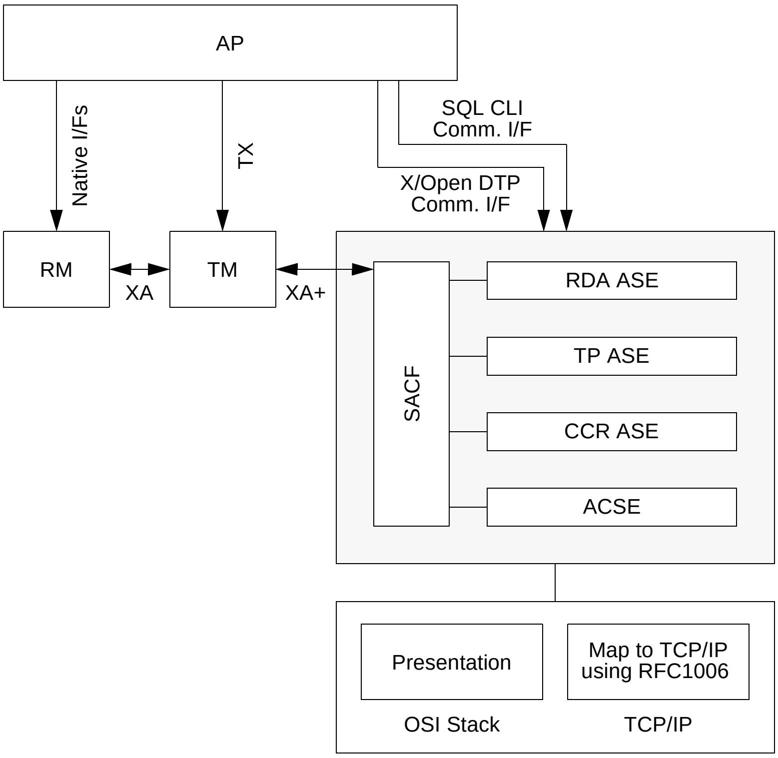

Figure 3: RDA as the X/Open Communication Resource Manager

In this application context, RDA and TP jointly provide services. The RDA application service element provides resource management, operation control and DBL command management services. The TP application service element provides dialogue management and transaction management services.

Figure 4 depicts a possible implementation where an application program uses the X/Open DTP model to coordinate local and remote database access through RDA.

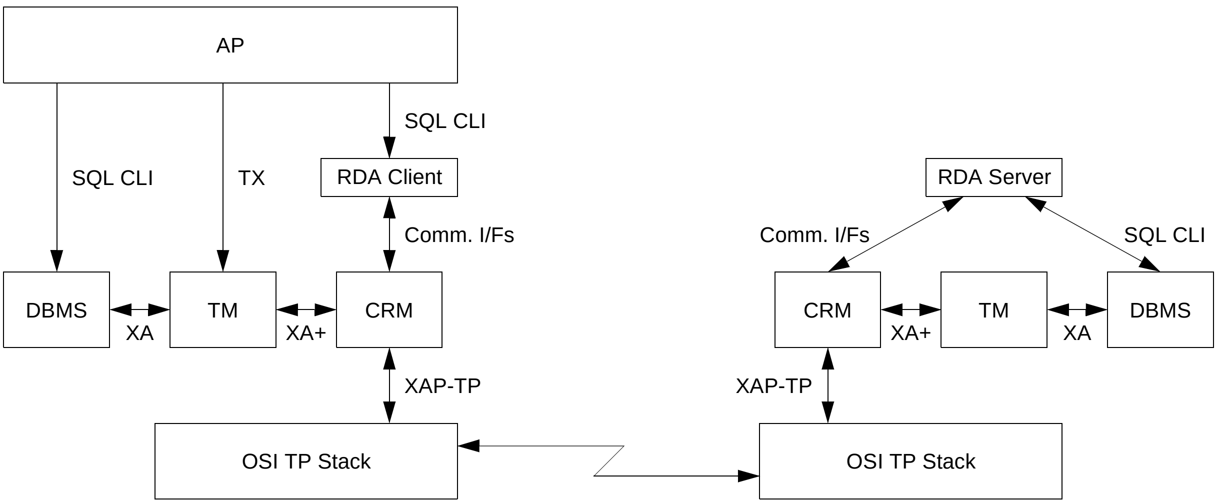

Figure 4: RDA with the X/Open DTP Model

4.3 OSI SYSTEMS MANAGEMENT WITH TP APPLICATION CONTEXT

OSI System Management [15,16,17] is a set of ISO standards that enable a standard way of defining management objects in the system and network management area. It also defines a standard set of services to define systems management functions. The standards in this area contain the following components:

- System Management Overview
  This overview defines the systems management model and different aspects of system management. It also defines the application context for system management.
- Systems Management
  This set of standards define a group of system management functions. The functions include:
  - Object Management Function
  - State Management Function
  - Relationship Management Function
  - Alarm Reporting Function
  - Event Report Management Function
  - Log Control Function
  - Security Alarm Reporting Function
- Structure of Management Information
  This set of standards provides the basic management information model. It defines what the managed objects are, what they are composed of, what they can do, what can be done to them, how they relate to one another, and how they are named. A set of general managed objects are defined in the standards. An abstract language, GDMO, is also provided for people to define their own objects in an unambiguous way.
- Common Management Information Service Element and Protocol Specifications
  These standards (CMISE [18], CMIP [19]) define the basic services to manipulate objects and provide a vehicle to convey messages passed between system management applications.

4.3.1 REQUIREMENT FOR TRANSACTION PROCESSING

The ISO Standards work relating to System Management with TP Application Context is conducted within the SC21 working group 4 and it has so far progressed to the committee draft status.

The main requirements for adding a TP application context as identified in the committee draft are:

- support for grouping CMISE service requests so that the consistency constraints can be satisfied without requiring provisions for rollback or recovery
- support for atomic synchronisation of a set of CMIS service requests with provisions for rollback and recovery so that either all the CMIS service requests are satisfactorily performed or none are performed.

4.3.2 CMISE AS COMMUNICATION RESOURCE MANAGER

As a communication facility for the systems management services, CMISE fits into the X/Open model as a CRM. Figure 5 depicts a common scenario where a global transaction involves access to a local database and to a remote management agent. A global transaction is initiated by an AP acting as a manager. The AP sends

CMISE requests to a remote AP acting as an agent. The CMISE operations are within a transaction involving operations on a local database.

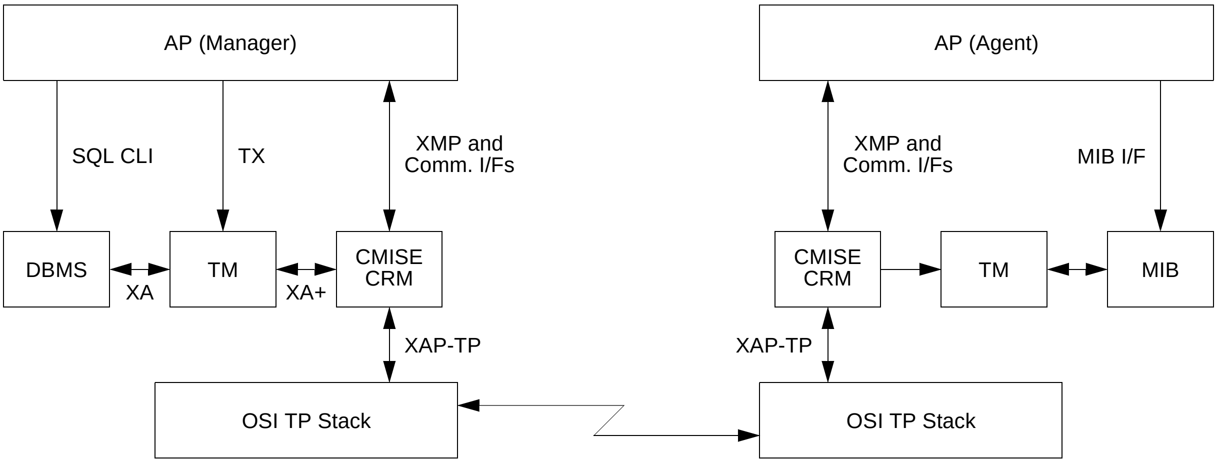

Figure 5: CMISE as a Communication Resource Manager

5 CONCLUSION

The standards work in both OSI DTP and X/Open TP provide very useful models for structuring distributed computing applications. It also provide standard service and communication protocol definitions and environments for architecting commercial distributed transaction processing technologies and platforms. Some commercial products are available based on the general framework set by the OSI and X/Open standards.

The OSI DTP committee is continuing work on some aspects of the standards. Notably, it is working on the two-phase commit protocol optimisation, nested transactions and the message queue facilities for asynchronous events produced during the span of the transaction.

Both the OSI DTP model and the X/Open DTP model do not address the issue of security. There is no mechanism for a TP service user to specify authentication and access control of a transaction branch. This is clearly an area for some future work.

6 ACKNOWLEDGEMENTS

The earlier version of this paper was submitted and accepted as an expert contribution to the ISO/IEC JTC1/SC21 WG3 DBL and RDA rapporteur groups. The author thanks the committee members for their comments and suggestions which led to this version.

REFERENCE

[1] ISO/IEC 10026-1:1992, *Information technology—Open Systems Interconnection*—Distributed Transaction Processing: Part 1 OSI TP Model.
[2] ISO/IEC 10026-2:1992, *Information technology—Open Systems Interconnection*—Distributed Transaction Processing: Part 2 OSI TP Service Definition.
[3] ISO/IEC 10026-3:1992, *Information technology—Open Systems Interconnection*—Distributed Transaction Processing: Part 3 OSI TP Protocol Specification.
[4] ISO/IEC SC21/WG3 Working Draft: *Database Language SQL* (*SQL3*), Jim Melton (ed.), August 1993.
[5] ISO/IEC 9579-1: 1992—*Remote Database Access*—Part 1 Generic model, Service and Protocol.
[6] ISO/IEC 9579-2: 1992—*Remote Database Access*—Part 2 SQL Specialisation.
[7] X/Open Guide G307: *Distributed Transaction Processing Reference Model* (V2), X/Open Company Ltd., November 1993.
[8] X/Open Preliminary Specification P209: Distributed Transaction Processing—*The TX (Transaction Demarcation) Specification*, X/Open Company Ltd., October 1992; ISBN: 1-872630-65-0.

[9] X/Open CAE Specification C193: *Distributed Transaction Processing—The XA Specification*, X/Open Company Ltd., December 1991; ISBN: 1 872630 24 3.

[10] X/Open Snapshot S214: *Distributed Transaction Processing—The Peer-to-Peer Specification*, X/Open Company Ltd., December 1992; ISBN: 1-872630-79-0.

[11] X/Open Snapshot S216: *Distributed Transaction Processing—The XATMI Specification*, X/Open Company Ltd., December 1992; ISBN: 1-872630-80-4.

[12] X/Open Snapshot S218: *Distributed Transaction Processing—The TxRPC Specification*, X/Open Company Ltd., December 1992; ISBN: 1-872630-81-2.

[13] X/Open Snapshot S201: *Distributed Transaction Processing—The XA+ Specification*, X/Open Company Ltd., March 1993; ISBN: 1-872630-93-6.

[14] ISO/IEC 8649: *Information Processing Systems—Open Systems Interconnection*—Service Specification for the Association Control Service Element.

[15] ISO/IEC 10140: *Information Processing Systems—Open Systems Interconnection*—System Management Overview.

[16] ISO/IEC 10164: *Information Processing Systems—Open Systems Interconnection*—Systems Management.

[17] ISO/IEC 10165: *Information Processing Systems—Open Systems Interconnection*—Structure of Management Information.

[18] ISO/IEC 9595: *Information Processing Systems—Open Systems Interconnection*—Common Management Information Service Definition.

[19] ISO/IEC 9596-1: *Information Processing Systems—Open Systems Interconnection*—Common Management Information Protocol Specification.

[20] ISO/IEC CD 11587: *Information technology—Open Systems Interconnection*— Application Context for Systems Management with Transaction Processing.

AUTHOR

Dr. Graham Chen has a BSc degree in computer science from Nanjing University, China in 1982. He earned his PhD in computer science from Heriot-Watt University in the U.K. in 1986. From 1986, he worked as research team leaders for two Alvey and Esprit projects in the U.K. for four years before he took a lectureship in the School of Computing and Information Technology at Griffith University. In 1991, Dr. Chen joined CiTR. Since then he has worked in many research and development projects in distributed information systems and telecommunication network management. Dr Chen is a senior consultant on commercial and R&D projects.

Dr. Chen keeps close contact with the academic and industrial research community. He also represents Australia in International Standards Organisations in Open Systems and Distributed Systems areas.